New Interfaces for Musical Expression

# NIME-03

## Proceedings

Montréal, May 22-24, 2003

Faculty of Music, McGill University
555, Sherbrooke Street West, H3A 1E3
Montréal, Québec, Canada

Proceedings edited by François Thibault
Cover graphical design by Catherine Motuz
Cover picture by Nicole Rivelli

**NIME-03 was organized by:**
Music Technology Area, Faculty of Music,
McGill University, Montréal, Québec, Canada

**NIME-03 was supported by:**
Fonds Québécois de la Recherche sur la Nature et les Technologies
The Social Sciences Humanities Research Council of Canada
Centre for Interdisciplinary Research in Music Media and Technology
Faculty of Music, McGill University

**Published by:**

McGill University, Faculty of Music
**ISBN: 0-7717-0604-9**


**Copies may be ordered from:**

Music Technology Area
Faculty of Music
McGill University
Prof. Marcelo M. Wanderley
email: nime03@music.mcgill.ca

**Homepage of NIME:**

http://www.nime.org/

## NIME03 Committees:

### Local organizing committee

Marcelo M. Wanderley (*conference chair*)
Philippe Depalle
Robert Ferguson
Sean Ferguson
Ichiro Fujinaga
D'Arcy Philip Gray
Wes Hatch
Richard McKenzie
Neil Middleton
Louise Ostiguy
Eileen Tencate
François Thibault
Adam Tindale
Philippe Zaborowski

### Artistic Committee

Butch Rovan (*coordinator*)
Teresa Marrin Nakra
Atau Tanaka
Todd Winkler

### Paper Committee

Philippe Depalle (*coordinator*)
Andy Hunt
Kia Ng
Nicola Orio

### Industry Liaison

Michael J. Lyons

### NIME Steering Committee

Tina Blaine
Sidney Fels
Michael J. Lyons
Sile O'Modhrain
Joe Paradiso
Ivan Poupyrev
Atau Tanaka
Marcelo M. Wanderley

### NIME 04 Organizer

Yoichi Nagashima

### Program Committee

Michael Alcorn, *Ireland*
Daniel Arfib, *France*
Curtis Bahn, *USA*
Marc Battier, *France*
Johannes Birringer, *USA*
Tina Blaine, *USA*
Bert Bongers, *Spain*
Richard Boulanger, *USA*
Bill Buxton, *Canada*
Antonio Camurri, *Italy*
Peter Castine, *Germany*
Joel Chadabe, *USA*
Insook Choi, *Canada*
Perry Cook, *USA*
Jeremy Cooperstock, *Canada*
Gideon D'Arcangelo, *USA*
Philippe Depalle, *Canada*
Peter Driessen, *Canada*
Stuart Favilla, *Australia*
Sidney Fels, *Canada*
Sean Ferguson, *Canada*
Mikael Fernstrom, *Ireland*
Emmanuel Flety, *France*
Ichiro Fujinaga, *Canada*
Suguru Goto, *France*
D'Arcy Philip Gray, *Canada*
Tomie Hahn, *USA*
Vincent Hayward, *Canada*
Andy Hunt, *UK*
Fernando Iazzetta, *Brazil*
Sergi Jorda, *Spain*
Daniel Levitin, *Canada*
Michael J. Lyons, *Japan*
Jonatas Manzolli, *Brazil*
Teresa Marrin Nakra, *USA*
Geoff Martin, *Denmark*
Axel Mulder, *Canada*
Sile O'Modhrain, *Ireland*
Charles Nichols, *USA*
Kia Ng, *UK*
Nicola Orio, *Italy*
Garth Paine, *UK*
Joe Paradiso, *USA*
Bruce Pennycook, *Canada*
Ivan Poupyrev, *Japan*
Miller Puckette, *USA*
Andrew Schloss, *Canada*
Laetitia Sonami, *USA*
Atau Tanaka, *France*
Dan Trueman, *USA*
Bill Verplank, *USA*
Marcelo Wanderley, *Canada*
David Wessel, *USA*

# Word from the hosting institution

Dear NIME participants:

It gives me great pleasure to welcome the 2003 NIME Conference to McGill University. McGill participants in the conference need little introduction to you all. But I would be remiss if I did not take this opportunity to thank on behalf of the Faculty of Music the work of the Prof. Marcelo Wanderley and the whole organizing committee.

Music at McGill has a long tradition, 100 years old in fact in 2004. Our relation to innovation in musical creation and performance is long established. In particular, the activities of the Electronic Music Studio since the 1960s, the Graduate Program in Sound Recording since the 70s, and the Music Technology program in the 1980s, have recently moved to a new level of dynamic interdisciplinary potential with the establishment of CIRMMT (the Centre for Interdisciplinary Research in Music Media and Technology). CIRMMT has become a driving force in the technological developments of key elements in the New Music Building Project that is now underway. A $50M project, the new building will include ECHO (the Electronic Concert Hall and Opera) facility -- a unique combination of symphonic-theatre scoring stage and acoustic research space with integrated control facilities, a 200-seat lecture-recital hall, three floors of music library, staff and administrative offices, conference rooms, and CIRMMT research labs. Yet these valuable spaces will still not be sufficient to contain much important music technology and interdisciplinary work that must be redistributed in our existing building or which goes on in satellite facilities in other faculties and institutions. McGill's Faculty of Music moves forward to the future informed by the strength of its past achievements and inspired by the potential of its newest students and staff. We hope to maintain and enhance our beautiful balance of musical creation, performance, and research. And we shall continue seek for ourselves and with others to develop "new interfaces of musical expression".

Welcome to Montreal, welcome to McGill University, *Bonne conférence*!

Don Mclean
Dean
Faculty of Music
McGill University

# Word from the conference chair

Welcome to the 2003 International Conference on New Interfaces for Musical Expression – NIME03!

NIME03 is the third event in the NIME series, after the initial NIME Workshop during CHI 2001 and the first international conference last year in Dublin (NIME02).

The main goal of the NIME03 conference is to blend high-level scientific and technological research on the development of new interfaces for musical expression and high-level artistic performances using such interfaces.

For this purpose, one of our main concerns in designing NIME03 was to insist on exploring the importance of *previous* works on interfaces for musical expression. In fact, although NIME can be considered as showcasing a brand new domain, several artists, researchers and engineers have been producing groundbreaking work in this area for decades.

This was the main force behind the choice of the invited speakers for this year's conference: the three keynote speakers are internationally known in this domain and each has substantially contributed to the development in this area: *Michel Waisvisz*, from STEIM, the Netherlands, *Claude Cadoz*, from ACROE-ICA, France, and *Joe Paradiso*, from the MIT Media Laboratory, USA. We will have the unique opportunity to learn from their experience in this area during NIME03.

On the scientific side, this year's NIME had an impressive list of reviewers among the most important experts in this area worldwide. Actually, NIME03 reviewers came from North and South America, Europe, Japan and Australia and reflect the various trends in this domain. The paper selection process was headed by Philippe Depalle (McGill University), who ensured the quality of the final publication.

Concerts will be presented at the end of each conference day, where the state-of-the art on new interfaces will be shown in a musical setting. The first concert will feature an invited performance of the *Wireless Duo* performing their score on the screening of the silent movie masterpiece *Faust*, by F. W. Murnau (1926). *Mark Goldstein* and *Dennis James* will use alternate controllers such as the *Buchla Lightning* and the *Theremin* to create the sounds that accompany the movie. The concert will take place in McGill's Redpath Hall, a former chapel whose architecture will help create the atmosphere for Murnau's Faust!

The second and third concerts will feature selected performances submitted to the NIME03 artistic committee, headed by Joseph Butch Rovan (University of North Texas) and will be performed in McGill's Pollack Hall.

Apart from the paper sessions and concerts, NIME03 will innovate on the format of the conference. We have designed this year's conference to optimize cross-fertilization, so that formal paper and report sessions will end early in the afternoons, leaving space for posters, demonstrations and workshops when there will be more flexibility for discussion. It is our goal that after the formal sessions (always single-track), delegates and artists will have the opportunity to discuss their works.

Another innovation is reflected through the various guest presenters to NIME03. They include some of the most representative artists and researchers in this area that will be presenting and discussing their works during the conference:

- *Max Mathews* presenting the new *Radio Baton* design, *Jana Saleh* and *Richard Boulanger* performing two of Boulanger's recent real-time multi-media pieces "*StarDust*" and "*DarkMatter*";
- *Tomie Hahn* and *Curtis Bahn* presenting "*Pikapik*a";
- *Alcides Lanza*, director of McGill Electronic Music Studio (EMS), giving a special talk on *Hugh Le Caine* and on some of the electronic instruments he developed in the 50's and 60's. In fact, *Le Caine* worked at McGill's EMS in the 60's and *Prof. lanza* had the opportunity to use some of his instruments on his own compositions.
- *Garth Paine* presenting his performance "Organic Serendipity".

Finally, the Dutch Institute STEIM will be the guest institution in NIME03. Apart from *Michel's* keynote address closing the conference on Saturday, STEIM will be presenting two workshops: the first one on various sensors, the new *SensorLab* interface and the software *LiSa* by *Frank Baldé* and *Joel Ryan* (Thursday and Friday afternoons) and the second one, "Ensemble", with 7 active garments, by *Kristina Andersen*.

In short, NIME03 will consist of:

- Three invited speakers: *Michel Waisvisz* (STEIM, NL), *Claude Cadoz* (ACROE- ICA, France), and *Joe Paradiso* (MIT Media Laboratory, USA).
- Research and development papers and reports in 9 single-track sessions, plus one poster session.
- Demonstrations of controllers, software and technologies for musical expression on Friday and Saturday afternoon.
- Three evening concerts featuring pieces for new interfaces.
- Invited demonstrations, installations and short concerts by outstanding guests.
- Workshops given by STEIM, the Dutch Institution that has been in the forefront of developments related to gestural controllers for the last three decades.

I am sure that NIME03 will be a unique opportunity to learn the latest developments on the area of new interfaces for musical expression and to exchange information.

Thanks to all of you – speakers, performers, guests, delegates – for contributing to the success of NIME03!

Have a nice conference and see you in NIME04 in Hamamatsu!

**Acknowledgements:**

Thanks to the Faculty of Music, McGill University and the Centre for Interdisciplinary Research on Music Media and Technology (CIRMMT) for support and funding, and the Social Sciences and Humanities Research Council of Canada (SSHRC) – aid to occasional research conferences and international congresses in Canada, the Fonds Québecois de la Recherche sur la Nature et les Technologies (FQRNT) – programme stratégique de professeurs-chercheurs for partially funding NIME03.

I am very grateful to all the NIME03 local organization team that was instrumental in putting this conference together. Thanks to their efforts and competence, NIME03 is a reality. My warmest thanks to:

Philippe Depalle, Sean Ferguson, Ichiro Fujinaga, D'Arcy Philip Gray, Richard McKenzie, Louise Ostiguy, and the students Robert Ferguson, Wesley Hatch, Neil Middleton, Eileen Tencate, François Thibault, Adam Tindale, Caroline Traube, and Philippe Zaborowski.

Thanks finally to the support team from McGill and friends who have been invaluable in helping the local committee to prepare NIME03: Katherine Simons, Jacqueline Gauthier and all staff at the Concerts and Publicity Office, Alain Terriault, Patrick Waegeli, Bruce Minorgan, Jeremy Cooperstock, Ian Knopke, and Anne-Marie Burns.

Marcelo M. Wanderley
NIME03 Conference Chair
Music Technology Area, Faculty of Music
McGill University

# Word from the paper committee

During these three days of NIME03, we will have the opportunity to attend roughly fifty presentations including the contributions of our invited speakers Joe Paradiso, Claude Cadoz, and Michel Waisvisz.

As you will see these presentations are organized into three paper sessions, six report sessions, one poster session and two demos sessions. We have broadly classified these presentations by subject, but decided not to give sessions topic names, as it might appear somewhat restrictive.

In order to take into account the various activities in the discipline, we have introduced a new submission category, namely *report*, on top of the usual submission categories (papers, posters, and demos). The idea is to split up the usual paper format into two categories depending on the nature of the contribution in the field: papers are primarily intended for new and original contributions, while reports are more for applied research on the design and usability of new instruments.

Each submission was anonymously (i.e. authors' name removed) reviewed by three program committee members, who ranked it in terms of relevance, originality, quality of presentation, appropriateness of the chosen length and category, as well as in terms of an overall appreciation. Reviewers also provided comments, and advices, which were then sent to the authors in order to help them finalizing their document. According to the advices of reviewers, and of the paper committee, several submissions were transferred into other categories (mainly papers into reports, but also reports into posters or demos). As regards numbers and statistics, we received 67 submissions (28 papers, 25 reports, 2 posters, and 14 demos) and accepted 47 submissions (9 papers, 23 reports, 6 posters, and 9 demos).

We have decided to complete the review process by sending the anonymous review comments on each paper to the two other reviewers. We believe this is an effective way to increase our awareness of each other's point of view.

Finally, I would like to thank the paper committee and the fifty three reviewers for their thorough evaluations of the submissions. I would also like to thank Wesley Hatch (general secretary), Robert Ferguson (web programming), and Francois Thibault (proceedings editing) for their invaluable assistance.

And last but not least, I would like to thank all the authors for submitting their work to NIME03! A dynamic and active research community remains the best guaranty for the success of both the conference and the development of this exciting field of New Interfaces for Musical Expression.

Have a great time and enjoy your stay in Montreal!

Philippe Depalle
Paper Committee Chair
Music Technology Area, Faculty of Music
McGill University

# Word from the artistic committee

The artistic committee for NIME 2003 included Teresa Marrin Nakra, Butch Rovan, Todd Winkler, and Atau Tanaka. Forty-five submissions were received in response to the call for works that highlighted the use of musical controllers, novel interface concepts, and/or new mapping systems appropriate to a concert environment. The materials submitted ranged from text-only proposals to online video and/or audio, and represented artists from all corners of the world.

To help facilitate the committee's work, an online interface to all of the submitted materials was created at CEMI (Center for Experimental Music and Intermedia, www.music.unt.edu/cemi/). CDROM versions of the website and index were also sent to all committee members. Each committee member judged all of the entries, and then an extensive online debate / discussion followed to reconcile the differences. Needless to say, it was a very difficult decision, as there were many interesting submissions and so few spots on the two concerts. In the end, the committee focused on those works that addressed the core concerns of NIME, as highlighted in the call for works, while showcasing a diversity of approaches.

The committee feels that the two programmed concerts represent an extremely interesting cross-section of the work being done today with new interfaces. We hope you enjoy the music!

Butch Rovan
Artistic Committee Chair
Center for Experimental Music and Intermedia
University of North Texas

## Table of Contents

**Report Session IV**



**Paper Session III**



**Report Session V**



**Report Session VI**



**Additional Documents**

**Poster Session**



**Demo I**



**Demo II**



**Invited Papers**

# Papers and Reports

# EpipE: Exploration of the Uilleann Pipes as a Potential Controller for Computer-based Music

Cormac Cannon
Media Lab Europe
Sugarhouse Lane
Bellevue, Dublin 8
+353 1 4742836
cormac@media.mit.edu

Stephen Hughes
Media Lab Europe
Sugarhouse Lane
Bellevue, Dublin 8
+353 1 4742864
steveh@mle.media.mit.edu

Síle Ó Modhráin
Media Lab Europe
Sugarhouse Lane
Bellevue, Dublin 8
+353 1 4742844
sile@media.mit.edu

## ABSTRACT
In this paper we present a design for the EpipE, a new expressive electronic music controller based on the Irish Uilleann Pipes, a 7-note polyphonic reeded woodwind. The core of this proposed controller design is a continuous electronic tonehole-sensing arrangement, equally applicable to other woodwind interfaces like those of the flute, recorder or Japanese shakuhachi. The controller will initially be used to drive a physically-based synthesis model, with the eventual goal being the development of a mapping layer allowing the EpipE interface to operate as a MIDI-like controller of arbitrary synthesis models.

## Keywords
Controllers, continuous woodwind tonehole sensor, uilleann pipes, Irish bagpipe, physical modelling, double reed, conical bore, tonehole.

## 1. INTRODUCTION
A musician's performance is intimately bound up with their physical means of expression. This relationship is equal parts struggle and dialogue – the instrument constrains, but its unique character colours and informs the performance. Until now, much of the focus in electronic musical interface design has concentrated on the keyboard, leaving many instrumental musicians at a loss when faced with a computer. The wind and string communities have been particularly poorly served – the more 'organic' nature of their interaction with their instrument making it much harder to measure meaningfully.

A number of flute and clarinet-style interfaces have been developed in the past, Akai's EWI and Yamaha's WX series of controllers being notable examples, but these fail to provide continuous tonehole control, a feature particularly important for players of open-holed folk woodwinds like the pipes, tin-whistle or traditional flute. Performers of Irish, Scottish, Breton and other folk musics make heavy use of sliding and half-holing techniques, and a controller that doesn't allow for this drastically restricts their range of expressive possibilities. The tonehole problem has been considered by some investigators, in the context of keyed woodwinds at least. Gary Scavone of CCRMA built the *Holey Controller[3]*, modifying a Yamaha WX11 to give a degree of continuous control by adding force sensing resistors under the keys, but this was primarily to provide him with a means of controlling his multiple-tonehole physical model of a clarinet. For a real instrument, the parameter of interest is the degree to which the player's finger is covering the tonehole rather than the force with which it is applied.

The Irish Uilleann Pipes, a polyphonic reeded woodwind (described more fully in the next section of this paper), provides an interesting basis for a new electronic interface. Like the Scottish Pipes, air is continually provided to the instrument and it is therefore completely finger-articulated. Unlike the Scottish pipes, the sound of the instrument may be stopped during play, allowing for staccato as well as legato playing and a wide range of tone colours and dynamic variation, greatly increasing the range and complexity of instrumental gesture. This combination of factors makes it an ideal test-bed for a new tonehole-sensor. Another feature of the instrument unusual in a member of the woodwind family is that it allows the player to provide their own chordal accompaniment using the regulators, a set of keyed pipes played with the wrist.

## 2. THE UILLEANN PIPES
A brief description of the Uilleann Pipes is provided here to render this document more intelligible to those unfamiliar with the instrument.

### 2.1 Physical Design
A set of Uilleann Pipes (see Figure 1) generally consists of:

- Bag
- Bellows
- Chanter
- Drones (3)
- Regulators (typically 3, though as many as 5 have been known)

The *bag* or air reservoir on the Uilleann Pipes is inflated by means of a *bellows*. The bellows are attached by a belt to the player's waist and by a length of flexible tubing to the bag.

The *Chanter*, attached to the neck of the bag, is the main melody instrument and has a range of two octaves, very unusual among bagpipes. Its tonehole arrangement is similar to that of the recorder, though the fingering is very different and the bottom of the chanter effectively serves as an extra hole, played by raising and lowering it from the knee.

**Figure 1: Set of D Uilleann Pipes made by the Taylor brothers, Chicago circa 1890.**

The *Drones* supply a continuous ‘drone’ accompaniment to the melody. Each drone is tuned by means of a slide – the tenor to the bottom note of the chanter, the baritone an octave lower and the bass an octave lower again.

The *Regulators* are keyed chanters and are used to provide a chordal accompaniment. The end of each regulator is sealed, so the regulator remains silent until a key is pressed. The three most usual regulators (tenor, baritone and bass) lie side-by-side across the player’s knee with the bass on the outside, near the chanter, the tenor on the inside, and the baritone between them. The bass and baritone each have four keys and the tenor five.

Both Drones and Regulators are mounted on a common stock which is connected to the body of the bag and rests across the player’s knee.

## 2.2 Playing Technique

The pipes are usually played sitting down so as to allow the regulators and chanter to rest on the knee. They are operated by the elbow of one arm while the bag is held under that of the other and used to maintain a steady supply of air to the rest of the instrument. The main factor determining the range of bag pressure to be applied is the chanter reed – those of the drones and regulators are of more straightforward construction and can normally be adjusted to match it. The chanter reed plays in quite a narrow range of bag pressure: too little and the reed will not sound; too much and it will blow closed. The drones, along with the regulators, lie across the player’s knee on the bellows-side, and may be activated or deactivated using a wrist-operated toggle switch.

### *2.2.1 The Chanter*

The Uilleann pipe chanter, though tuned differently, has an identical tonehole arrangement to that of the Scottish bagpipe chanter. Of the seven toneholes on the front of the instrument, the top three are covered with the 1st through 3rd fingers of one hand (usually the left) and the remaining 4 with the 1st through 4th fingers of the other. The topmost tonehole is on the back of the chanter and is covered with the thumb of the upper hand.

The Uilleann pipe chanter differs from that of other bagpipes in three major respects:

1. Its reed, like that of the oboe, is a double reed which can be overblown into the second octave, effectively doubling its range.
2. The open end of the chanter serves as an extra hole. The chanter is usually played resting on the knee, allowing the player to stop the sound abruptly between notes by covering all the toneholes. This enables the performer to play “tight” staccato passages as well as the legato stream of notes to which players of other bagpipe instruments are restricted. Notes of the scale are played with just one or two of the toneholes uncovered, facilitating rapid staccato runs and allowing the performer to modulate the tone by uncovering lower toneholes and/or raising the chanter from the knee. Vibrato may be achieved by rapidly uncovering and re-covering the lower toneholes and, optionally, raising the chanter from the knee to produce a more strident tone.
3. The bottom note of the chanter can be played in two different pressure regions, one producing a tone similar to that of the other notes of the lower octave, the other a much stronger tone, known as the “hard” bottom D which can be used to great ornamental effect.

As with other open-holed folk woodwinds, much use is made of the technique of “sliding” from a lower note to the note above and of rapid grace notes.

**Figure 2: Principal author playing a set of pipes pitched in D**

### *2.2.2 The Regulators*

The Regulators lie across the player’s knee on the same side as the player’s lower chanter hand and are usually played using the wrist of the this hand, though while playing certain notes the hand may be removed from the chanter to finger more complex chords.

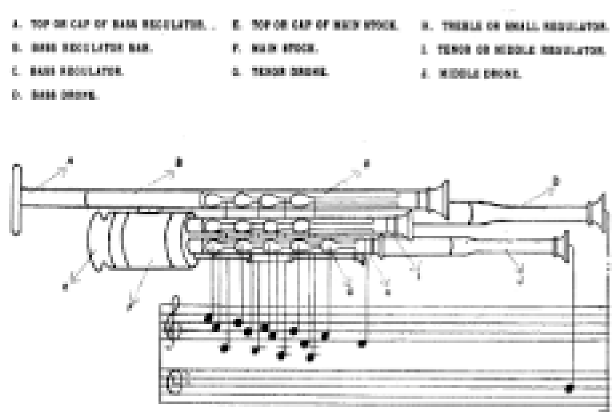


**Figure 3: Regulator Tuning Diagram [2]**

When played with the wrist, the upper four rows of regulator keys may be struck singly or in chords of two or three notes across a row. The 5th note on the tenor regulator is usually played in combination with the lowest note on the baritone. These notes and chords may be sustained or used for rhythmic effect.

## 3. DESIGN OF THE EPIPE INTERFACE

There are two parallel strands to the development of the EpipE interface: the physical design of the controller and the construction of an appropriate synthesis model with which to test the design.

### 3.1 Physical Controller Design

During the initial phase of development, we considered the instrument itself and the way it is played in an effort to identify the parameters to be measured and determine appropriate ranges, resolutions and sampling rates for them. Starting with these observations, we have sketched a design outline for the interface prototype and have made substantial progress in the process of designing and constructing the required electronic circuitry.

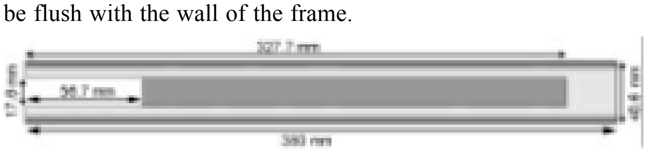
#### *3.1.1 The Bag and Bellows*

The air pressure in the bag is a function of the force applied by the player's elbow. One means of quantifying this parameter would be to attach a pressure sensor to a sealed bladder. The other obvious solution is to measure the applied force directly using a force-sensing resistor. We have chosen the second approach for reasons of cost and ease-of-integration.

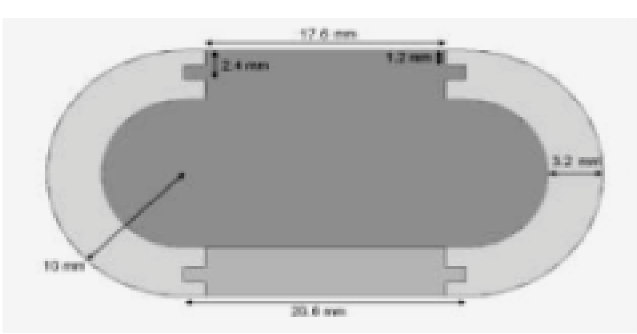
The Bellows serve the purely mechanical purpose of filling the bag with air to drive the reeds. As no air flow is required through our electronic interface we have opted to discard the bellows, thereby easing the task of playing the regulators.

#### *3.1.2 The Chanter*

The EpipE chanter consists of a 3D-printed tube-like frame housing two printed circuit boards and a momentary switch. The frame is a flattened oval in cross-section, approximately 38 cm in length, and has slots front and back for the circuit boards. The front board is a four layer PCB 17.6 mm across and 328 mm in length. It has seven holes along its length, whose spacing and dimensions are based on measurements taken from a number of instruments in the principal author's possession. The back board is of identical cross section, 57 mm in length, and contains the thumb-hole and its supporting circuitry. The perimeters of both PCBs are routed to a depth of 1.2 mm (half the board thickness), allowing them to be mounted securely in their respective slots in such a way as to be flush with the wall of the frame.



*Plan View*



*Cross Section*

**Figure 4: EpipE Chanter**

##### *3.1.2.1 The Tonehole Sensor*

The selection of an appropriate technology for the chanter tonehole sensor has proved to be the biggest challenge encountered to date in the development process. The length of the air column, which determines the pitch produced, is predominantly governed by the position of the first opening in the bore of the instrument; therefore the particular parameter we want to measure is degree of tonehole coverage. More specifically, we must measure the degree of tonehole edge coverage, as the player's finger may begin to cover or uncover the hole from a range of different angles. The tactile feedback provided by the feel of the tonehole edges is an important aid to the player of the real instrument, and for this reason a flat plate-style sensor would be unsuitable for our purposes.

Numerous approaches have been considered, most involving optical or capacitive technologies, or a combination of the two. The idea of using capacitive sensing in its analogue range was discarded due to its extreme sensitivity to variations in humidity, skin-conductivity and other environmental variables. Likewise, the optical solutions considered would have been susceptible to variations in ambient light and, in some cases, to variations in the levels of infrared radiation emitted by the performer. An instrument built using any of these approaches would need to be calibrated to each individual user and set of conditions, and would not provide the clean open, closed and partially covered readings required for a robust electronic interface.

We eventually decided to experiment with a number of capacitive touch sensors spaced evenly around the rim of the tonehole. The advantage of this approach is its reliability – each individual switch is either definitely on or definitely off – but the drawback is that a relatively high number of switches is required to approximate an analogue range. The issue is not the frequency resolution required for held notes – over the semitone to full-tone intervals involved, a relatively small number of sensors would suffice – but rather the resolution required while sliding between notes. Some of the technologies used in fingerprint sensors would be ideal for the task, as would those used in some laptop pointing devices. Regrettably, these are usually implemented as custom-designed microelectronic devices and none are available in the unusual form factor we require.

Our first tonehole prototype was built using discrete analogue components and comprised six individual capacitive touch-sensor circuits connected to electrodes mounted around the edge of a hole. The number is of no particular significance, but worked out conveniently in terms of the electronic components involved and was deemed sufficient to test the reliability of our solution. These sensors were connected as inputs to a PIC microcontroller and used to drive a single-hole clarinet model provided with Perry R Cook's Signal Processing Toolkit[4] and some single-hole flute models of our own, with the expected results. Even with the limited resolution of our prototype, it was possible to consistently produce sounds corresponding to fully open and fully closed tonehole states as well as a satisfactory semitone. Attempts to slide, however, produced stepped tones rather than smooth legato, highlighting the need to increase the sensor density to the point where each touch electrode corresponds to a change in frequency of less than the Just Noticeable Difference discernible by the human ear. The JND varies with frequency, but can be taken to be about $1/12^{th}$ of a semitone in the frequency range of interest in our work [5]. Filtering the composite sensor signal to conceal its stepped characteristic is not appropriate in the context of a performance instrument, as this would introduce an unacceptable delay of somewhere around 30ms (the shortest measured duration for a complete finger gesture in a sample of fast, staccato piping).

For the second prototype we increased the number of electrodes-per-hole to sixteen. In this denser system, dedicating a touch-sensor circuit to each electrode would have entailed a prohibitive component count. Instead, we chose to use a single capacitive touch-switch circuit and to multiplex the tonehole sensor-pads using some new analogue switches with ESD protection manufactured by Maxim. This arrangement, detailed below, provides a satisfactory sliding effect between notes and forms the basis of our multi-tonehole circuit. Multiplexing the tonehole-sensor pads has enabled us to reduce the circuitry considerably and poll all 128 pads at a rate of 100 Hz using just two touch-plate sensor sub-circuits.

Figure 5 is a simplified representation of the sensing circuitry for a single electronic tonehole. A high frequency oscillator feeds the top node of an RC (Resistor-Capacitor) potential divider. A capacitance, the parallel combination of parasitic capacitances to ground and the capacitance of a tone-hole pad to ground, forms the lower half of the divider. When the player's finger is in contact with the pad this capacitance typically increases several fold, thereby causing the voltage at the centre node of the potential divider to fall in magnitude. This drop is detected by comparing its peak value to a reference voltage. By multiplexing all the tone-hole pads to the centre node of the potential divider, one peak-detect / comparator circuit can service a much larger number of tonehole pads. The microcontroller manages the multiplexer addressing and converts the sensor signals to a custom serial protocol for output.

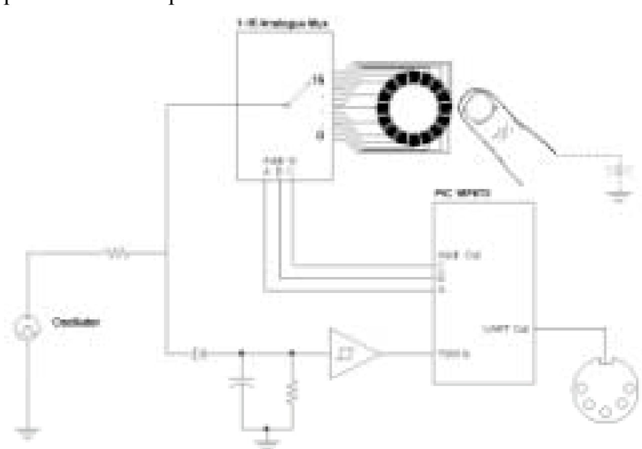


**Figure 5: Block Diagram of Single EpipE Tonehole**

### *3.1.3 Drones and Regulators*

The drones and regulators of the EpipE will be realised as a set of switches which connect directly to a distribution board mounted in the modified stock of a real set of pipes. The switches' states are multiplexed and monitored by the PIC on the chanter PCB. The distribution PCB also houses various housekeeping sub-systems, including an RS-232 driver, an FSR circuit for bag pressure measurement and power supply circuitry.

The interface to the drones is a simple wrist-operated on-off toggle switch. It might also be useful to include another button to cycle through additional pre-programmed drone tunings not achievable on a real instrument; to activate an extra drone tuned a 5th above the baritone for example.

The regulator keys can easily be instrumented, either by mounting simple momentary switches in the holes they cover or by attaching wires directly to the keys themselves and allowing them to act as capacitive touch-plates. An alternative approach would be to use a position sensitive strip, like those used in some MIDI keyboards, for each regulator. This would allow for more flexibility of use – in standard operation the strips could be divided into zones corresponding to the various regulator keys, but an alternative mode could be implemented in software allowing the player to slide between notes or perhaps use regulator vibrato.

## 3.2 The Synthesis Model

In order to properly evaluate our interface, we will need a carefully designed synthesis model to drive with it. The subsections below review the synthesis possibilities that seem most appropriate for the chanter, drones and regulators.

### *3.2.1 Chanter Synthesis*

The complex inter-relationships of the control parameters being measured here suggest a physical modelling approach as being the most appropriate during the initial evaluation of our electronic interface. This will involve tackling a number of difficult physical modelling issues, particularly in relation to the double reed excitation of the chanter. The following section offers a brief review of recent developments in the relevant areas of physical-modelling synthesis of woodwind instruments.

#### *3.2.1.1 The Double Reed*

Although the single reed of the clarinet and saxophone is well understood, the double reed remains something of a mystery. The two halves of the double reed run close to one another for a considerable distance near their tips, making Bernoulli forces become particularly important[6]. The flow resistance in the narrow staple also has a significant effect on its operation[7]. Josep Nebot reports some success with a CSound synthesis of the Grallot, a sort of double-reeded clarinet[8]. Cook (1995) has also conducted some research on the Aulos[9], a double-reed cylindrical-bore woodwind dating from the 5th century BC, concluding that more theoretical and experimental work was required from the acoustics community before an accurate synthesis model of the double reed could be constructed. Guillaume Lemaitre, working in the Université du Maine and IRCAM, recently completed a masters thesis on a physical model of the oboe[10] and published a paper with Christophe Vergez, Xavier Rodet and René Caussé on the influence of bore conicity and the pipeneck downstream of a double reed[11]. They explain that the presence of the narrow pipe (or staple) causes a pressure drop between the top of the staple and opening of the conical bore, invalidating the flow model commonly used for the single reed, and propose a simple model for the pipeneck with the pressure drop represented by a discharge-loss coefficient. The resulting reed model exhibits three different behaviours depending on the discharge-loss coefficient chosen for the pipeneck.

#### *3.2.1.2 The Tonehole Model*

Keefe (1981) [12] presented a study of woodwind tonehole providing distinct models for the open and closed toneholes. These models were translated for efficient digital waveguide implementation by Scavone and Smith[13]. In his doctoral thesis[14], Scavone presents a new tonehole model capable of dynamic state changes from fully-open through fully-closed. This latter solution does have a limitation on the minimum tonehole length (one spatial sampling interval) later addressed by van Walstijn and Scavone using wave-digital filtering techniques[15]. This wave-digital model, though more computationally complex than that presented in Scavone's thesis, is probably the most appropriate for our chanter model given the small diameter of some of the chanter toneholes.

#### *3.2.1.3 The Conical Bore*

The bore of the chanter is approximately a truncated straight sided cone with some subtle but significant deviations[1] which affect the alignment of the air column's modes of oscillation. For synthesis purposes, the chanter bore can be approximated as a serious of conic sections interspersed with tonehole scattering junctions.

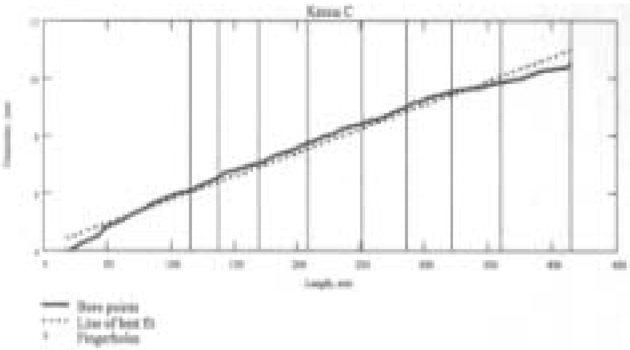


**Figure 6: Bore Diameter vs. Distance for a Kenna Chanter pitched in C [15]**

Various previous studies have demonstrated the difficulty of establishing stable regimes of oscillation in a truncated conical bore. Scavone (2002) [12] presents a flexible model for such a bore and describes its implementation using digital waveguide techniques. In his doctoral thesis [13], on which our most recent work in this area has been based, Maarten Van Walstijn uses wave-digital filtering techniques to address some issues relating to the modelling of toneholes in a conical bore.

#### *3.2.1.4 Sound Radiation*

Given that much of the dynamic variation in piping is achieved by opening and closing chanter toneholes, it will be important for us to accurately model the contribution of each tonehole to the overall sound radiated by the instrument. Scavone has dealt with this area in a paper on the modelling of wind instrument sound radiation in 3D space published in the proceedings of ICMC 1999, approximating each tonehole as an open-pipe discontinuity [14]. Lower frequency components tend to radiate in an almost omni-directional pattern while those at higher frequencies have greater magnitude in front of the pipe opening and along its axis. Scavone uses a frequency-dependent directivity filter for each radiation source to account for the angle between source and pickup-point. For our purposes it should be safe to assume that the observer is at a sufficient distance from the instrument that the angle to each source is approximately equal, allowing us to sum the pressure components contributed by each tonehole directly and avoid the additional computational complexity the inclusion of directivity filters would incur.

### *3.2.2 Drone Synthesis*

A physical drone model would be similar to that of a simple clarinet and could be readily realised using digital waveguide techniques. As the drones are not "played" as such but rather tuned to the chanter at the outset and switched on and off as needed, a number of other synthesis approaches (additive, or even sample-based) could also be used interchangeably, or MIDI signals could be used to drive an external synthesis module.

### *3.2.3 Regulator Synthesis*

The physical construction of the regulators would suggest a modification of the chanter synthesis model as being the ideal regulator synthesis solution. When we attend to the way in which they are played, however, it becomes clear that a much simpler and less computationally intensive solution will suffice. The regulators are key-operated, so a regulator tonehole may be held only in its fully-closed or fully-open state. This suggests that a simplified tonehole model will be adequate to the task. We can further simplify things when we consider that only one tonehole on each regulator will be open at any given time, suggesting that a model consisting of a double reed coupled to a conical, or even cylindrical, bore should be sufficient for our purposes. Indeed, as with the drones, the more mechanical way in which the regulators are played detracts from the benefits to be gained from a physical-modelling synthesis approach in terms of note transitions etc. The pitch corresponding to each regulator key is known, so the additive, sample-based and external synthesis options mentioned in the drone section above remain equally viable in this case.

## 4. CONCLUSIONS AND FUTURE WORK

In the course of the research described here, we have studied the Uilleann Pipes and the way in which they are played in an effort to identify the physical parameters that are manipulated by the player in their interaction with the instrument. On the strength of our findings we have come up with a design for a new electronic interface, at the core of which is a new tonehole-state sensing solution. We have successfully built two tonehole prototypes and written a Java interface allowing us to test them with physical models constructed using Perry Cook's Synthesis Toolkit (STK). Based on the design of the second of these prototypes, we have laid out the PCBs for an 8-tonehole electronic chanter. In preparation for the construction of a physically-based synthesis model with which to test our interface, we have reviewed much of the existing literature on the double reed and other relevant aspects of woodwind acoustics and physical modelling synthesis. Work has been begun on the implementation of the required building blocks within the framework provided by the STK.

Our observations suggest that the Uilleann Pipes have great potential as a basis for a new electronic musical controller. Their combination of expressive woodwind-style melodic control with the harmonic capabilities of the regulators should make them an attractive option in a performance context as well as a novel expressive MIDI controller. Our immediate goal is to complete construction of our physical interface. In parallel, we will be continuing to develop the physical synthesis model of the chanter discussed in section 3.2. Having verified our sensor arrangement using this synthesis model, the eventual goal is the abstraction of our interface through the development of a mapping layer which will allow us to drive arbitrary synthesis models using the EpipE controller.

## 5. ACKNOWLEDGEMENTS

Thanks are due to Perry Cook and Gary Scavone for consultations on the subjects of the double reed and the conical bore respectively.

## 6. REFERENCES

[1] Fischer, C., *Pipe Bores and Pipe Acoustics*. Ceol na hÉireann, 2001. 3: p. 24-37.

[2] Rowsome, L., *Tutor for the Uileann Pipes*. 1936, Dublin: Walton's Musical Instrument Galleries Ltd. 35.

[3] Scavone, G.P., *The Holey Controller*. 1999.

[4] Cook, P.R. and G.P. Scavone. *The Synthesis ToolKit (STK)*. in *ICMC*. 1999. Beijing.

[5] Worrall, D., *The Physics and Psychophysics of Sound & Music*.

[6] Benade, A.H., *Fundamentals of Musical Acoustics*. 2nd ed. 1976: Dover Press. 596.

[7] Fletcher, N.H. and T.D. Rossing, *The Physics of Musical Instruments*. 2nd ed. 1998: Springer-Verlag.

[8] Nebot, J.M.C., *CSound Implementation of Physical Models of Woodwind Instruments*. 1999.

[9] Cook, P.R. *Greek Aulos Project Status Report: Acoustics of Double Reed Cylindrical Bore Instruments*. in *Symposium on Physical Models and Applications in Psychoacoustics*. 1995. Thessaloniki, Greece.

[10] Lemaitre, G., *Synthèse de Hautbois par Modèle Physique*, in *DEA (Acoustique Appliquée)*. 2000, Université du Maine.

[11] Lemaitre, G., et al. *Physical modeling of oboe-like instruments: influence of the bore conicity and of the pipeneck after the double reed*. in *SCI*. 2001.

[12] Scavone, G.P. *Time-Domain Synthesis of Conical Bore Instrument Sounds*. in *ICMC*. 2002. Göteborg, Sweden.

[13] van Walstijn, M., *Discrete-Time Modelling of Brass and Reed Woodwind Instruments with Application to Musical Sound Synthesis*. 2002, University of Edinburgh: Edinburgh. p. 312.

[14] Scavone, G.P. Modeling Wind Instrument Sound Radiation using Digital Waveguides. in ICMC. 1999. Beijing, China.

[15] Fischer, C., *Pipe Bores and Pipe Acoustics*. Ceol na hÉireann, 2001. 3: p. 24-37.

[16] Scavone, G.P. *Time-Domain Synthesis of Conical Bore Instrument Sounds*. In ICMC. 2002. Göteborg, Sweden.

[17] van Walstijn, M., *Discrete-Time Modelling of Brass and Reed Woodwind Instruments with Application to Musical Sound Synthesis. 2002,* University of Edinburgh,Edinburgh. p. 312.

[18] Scavone, G.P. *Modeling Wind Instrument Sound Radiation using Digital Waveguides*. In ICMC. 1999. Beijing, China.

# HyperPuja: A Tibetan Singing Bowl Controller

Diana Young
MIT Media Lab
20 Ames St.
Cambridge, MA 01239
diana@media.mit.edu

Georg Essl
Computer & Information Science & Engineering
University of Florida
Gainesville, FL 32611
gessl@cise.ufl.edu

## ABSTRACT

HyperPuja is a novel controller that closely mimicks the behavior of a Tibetan Singing Bowl rubbed with a "puja" stick. Our design hides the electronics from the performer to maintain the original look and feel of the instrument and the performance. This is achieved by using wireless technology to keep the stick un-tethered as well as burying the electronics inside the the core of the stick. The measured parameters closely resemble the input parameters of a related physical synthesis model allowing for convenient mapping of sensor parameters to synthesis input. The new controller allows for flexible choice of sound synthesis while fully maintaining the characteristics of the physical interaction of the original instrument.

## 1. INTRODUCTION

In this paper we describe a new controller based on an instrument called the Tibetan singing bowl. These bowls have received increased attention in the Computer Music community in recent years [22, 15, 8, 23, 17, 3, 18, 19] for their intriguing sound and performance style.

Tibetan singing bowls are metallic or glass bowls in shapes varying from that of a spherical segment to an almost cylindrical shape. The bowls can be made to ring by striking. The sound of a bowl when struck is related to bell sounds [13]. The sound mainly depends on the shape, material and size of the bowl and the hardness of the striker.

However the more characteristic performance type is based on a rubbing interaction. The bowl is rubbed with a wooden stick called "puja", which may or may not be wrapped in a thin sheet of leather. The performer rubs with the stick around the outer rim of the bowl at various speeds and tangential forces that, if performed correctly, create a sustained ringing sound.

In recent years, research on Tibetan bowls has focused on two main aspects. One is the use of real and virtual bowls in performance through novel interfaces. Atau Tanaka [18] used an array of acoustic Tibetan bowls, which he played while controlling additional electronically created or modified sounds through electromyogram and position sensing technology developed by him and Benjamin Knapp [19]. The technology was used in two ways: The first was used to electronically augment the original sound through the control of the measured gestures. The second does not use original sound, but rather the gestures create independent sound articulations.

Carr Wilkerson and co-workers [22, 23, 17] used the Mutha Rubboard Controller to play physical models of the Tibetan singing bowl. The Mutha Rubboard Controller is a controller motivated by washboard playing as present in Zydeco music. A capacitive sensing technique was used to determine the position of the key. This was used as a contact free excitation mode for the virtual bowl. The performer would use circular up-down hand motions in front of his body to feed energy into the synthesis algorithm.

Research in controllers for musical expression has seen an increased development in theoretical foundations and guiding principles [2; 4; 5; 11; 21, for example]. The mapping problem, that is the relation of control-device output to synthesis algorithm input, has seen both theoretical and experimental advances [9; 10; 14, for example].

The type of friction behavior that is responsible for the oscillatory action of the rubbing stick on a Tibetan bowl is known as stick-slip friction [1], a mechanism also reponsible for the dynamic function of string instrument bows. Bowed string controllers are under ongoing development. One line of research augments the violin bow and maintains the original bow action [24, 20] whereas another line includes haptic feedback in the controller design [12].

In the remainder of the paper we describe the design of a new controller for Tibetan singing bowls by implementing an electronic sensor version of the "puja" stick that we call the "HyperPuja".

## 2. HARDWARE IMPLEMENTATION OF THE BOWL CONTROLLER

In designing the first prototype of the HyperPuja stick, there were several priorities established from the onset. Because the most immediate goal of this research is to allow the performance of the Tibetan bowl physical model using traditional playing technique, we wanted to maintain the ergonomics of the stick above all. Therefore, we wanted to make a system of sensors that was both as small and as light as possible. We also aimed to maintain the natural wireless feature of both the bowl and the stick, and so it was decided that the data transmission would be performed using an RF module.

In this work, we also sought to integrate as much of the electronics within the structure of the HyperPuja stick as possible, so that in all ways the stick would give the appearance and feel of that of its traditional counterpart. Therefore, a design was desired that allowed the various components of the electronics system to be placed almost entirely inside the stick itself.

Because we wanted to create an interface that could not only be used as a means of controlling a physical model, but could also offer the player the use of the acoustic sound of the bowl (so that, for instance, a player might be able to play a duet between real and virtual bowls), another priority of the interface design was that the bowl itself remain as untouched by technology as possible.

With the above priorities in mind, we sought to create a prototype of the HyperPuja interface that would be capable of measuring data relating to the velocity of the moving stick around the bowl, the pressure between the stick and the bowl, and the acceleration of the stick along its trajectory. The first two types of measurements are of obvious interest for this project, as they relate directly to parameters of the basic physical model of the Tibetan bowl. Though acceleration measurement is not crucial to the control of the model, we included it in our design specification for possible use in extended performance control.

The acceleration measurement was performed quite simply using a commercial 2-axis accelerometer.

In order to measure velocity, we began by placing magnets, which are the only visible components of the sensing system associated with the controller, on the inside of the singing bowl . These were temporarily affixed to the side of the bowl using a small amount of easily-removable putty. Two Hall effect sensors were then employed to detect the presence of the magnetic field produced by the magnets. The rate of the appearance of the peaks in the Hall sensor data is then taken to reflect the velocity of the stick in its trajectory around the bowl.

Building the pressure sensor for the HyperPuja was a challenge, as we wanted the model to be able to respond to continuous pressure changes between the stick and the bowl occurring at any contact point on the traditional playing area of a rubbing stick. This criteria concerning area, as well as the added constraints of working with a curved surface and our initial priorities demanding simplicity in hardware, made the common use of devices such as FSRs unfeasible here.

Through experimentation, a novel pressure sensor was designed. Conductive rubber, which responds to changes in pressure with a decrease in the resistance measured between the planar surfaces at the points directly above and below the point of contact, was ultimately used. The pressure sensor was comprised of three layers of material wrapped around the stick: a thin piece of copper foil carefully adhered directly to the stick, a sheet of conductive rubber (0.5mm think) placed around the first layer of copper, and a piece of finely woven copper fabric secured over the rubber. By this construction, the conductive rubber is held in place simply by the sleeve of copper fabric and a final piece of chamois placed over the entire assembly that closely resembles the leather found on many traditional rubbing sticks. The changes in resistance that occur as a result of pressure fluctuations are measured by making contacts to the two pieces of copper material.

We began the construction of the HyperPuja stick by first hollowing out the middle volume of a traditional rubbing stick. This process left a 0.9" diameter cavity in which to place our electronics. The electronics board that houses the accelerometer, Hall sensors, microcontroller, wireless transmitter, and the battery were all placed inside the barrel of the stick. The electronics board is powered by a 3V camera-style battery that has a lifetime of over 30 hours in this system. The microcontroller used for the HyperPuja is a PIC16LF877, which possesses an internal 10-bit A/D converter. The signals from the Hall sensors and the pressure sensor are sent to inputs here. The accelerometer (ADXL202) outputs two digital signals, which are routed to two other input pins of the PIC. The data stream for all of the sensors is transmitted with a wireless transmitter using a serial protocol that is received remotely. This data stream for all the sensors is then sent using the wireless transmitter with a serial protocol. This data is then collected by a receiver on a remote board that connects to the serial input of the Windows machine used for our experiments. The electronics inside the HyperPuja stick can be seen in Figure 1.

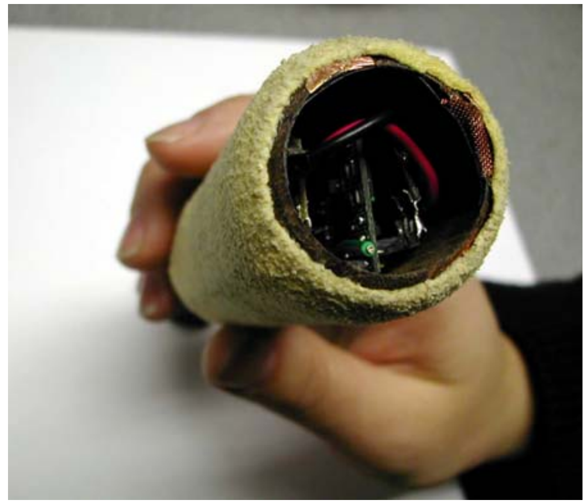


**Figure 1: The electronics inside the HyperPuja stick.**

The complete design can be seen in Figure 2. Here, the magnets used for the velocity measurement, the only apparent component of the sensing system, may be seen inside the bowl.

**Figure 2: The HyperPuja stick with a Tibetan singing bowl. The magnet for the hall effect sensors can be seen inside the bowl.**

## 2.1 Sensor Data and Synthesis Model Parameter Visualization

The HyperPuja data is displayed using a C++ stripchart application. For the first experiment using the HyperPuja to control the physical model of the Tibetan bowl, we established a C++ software link between this GUI and the tcl/tk GUI displaying the input parameters of the physical model. This method of interfacing between the controller and the model was chosen because it offered a test setup that facilitated control of the model with the HyperPuja and also allowed immediate experimentation with un-mapped parameters using the computer mouse.

The whole setup in performance is depicted in Figure 3. The first working prototype of this interface allows a performer to control each of the "bow pressure", "bow velocity", and "integration" parameters of the synthesis model with the pressure, hall effect, and acceleration sensors in the HyperPuja stick, respectively.

**Figure 3: The HyperPuja stick in performance. The laptop screen shows the sensor data display on the left and the sound synthesis GUI on the right.**

## 3. PLAYING THE STICK

These initial experiments with the current implementation of both hardware and software are very encouraging, as with limited practice a player is able to create a convincing performance using the HyperPuja of what sounds and looks much like a traditional Tibetan singing bowl. Often during various practice sessions, casual observers remained unaware that the sound they were hearing was not being generated by the acoustic bowl. From the perspective of the performer, the HyperPuja stick was found to have a comparable "feel", both in terms of weight and the perceived friction produced between it and the bowl, to a traditional rubbing stick.

In playing the model, it is apparent that the nature of the sound produced is the product of the unique evolution of the current performance. That is, the sound builds and progresses in such a way that is strongly influenced by time and reflects the history of the interaction between the stick and the bowl. This behavior reflects the non-linearity of the friction mechanism [1]. This characteristic of the interaction, which is of course also shared by a real Tibetan bowl performance, greatly contributes to the overall experience of playing the HyperPuja, as there is an inherent sense of exploration and discovery in the process of playing. From this initial aesthetic observation, it would seem that some of the challenging and engaging performance characteristics of the traditional instrument are maintained by this controller. The desirability of such traits in new controllers is discussed in [9].

## 4. ACOUSTICS AND SYNTHESIS OF TIBETAN SINGING BOWL

The Tibetan singing bowl has received modeling attention in recent years [8, 23, 17, 3] based on banded waveguide synthesis which was earlier introduced for struck and bowed bar percussion instruments [7]. In the following we will repeat the essential features of the acoustic properties and the synthesis algorithm as it was used in conjunction with the controller.

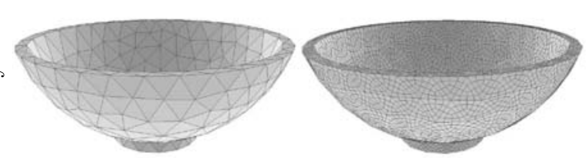



**Figure 4: Mesh of simulated bowl.**

For this discussion we'll assume a Tibetan singing bowl that is geometrically close to a spherical segment. A discretized mesh version of the bowl can be seen in Figure 4). Depending on the rubbing velocity and initial state of the bow (i.e. certain modes may be already ringing), various frequencies can be made to oscillate. Behavior is comparable to rubbing or bowing a wine glass [8, 15] in terms of dynamic envelope, mode locking, mode duplication and related phenomena as a result of the non-linear interaction of the stick-slip-based rubbing.

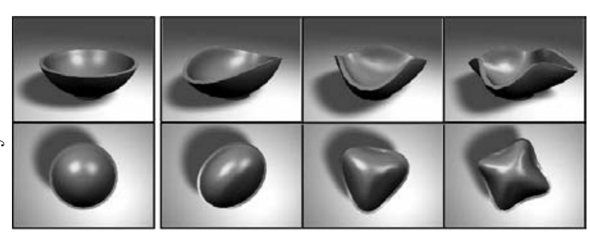



**Figure 5: Simulated mode shapes of the bowl.**

If struck, the bowl will show a modal response of circularly symmetric form. The first few modal shapes are depicted in Figure 5 with exaggerated amplitudes. These shapes will oscillate around the circular rest position in a manner comparable to circular flexing motion of the wine glass depicted in Figure 6. The circularly repeating pattern is depicted in Figure 7.

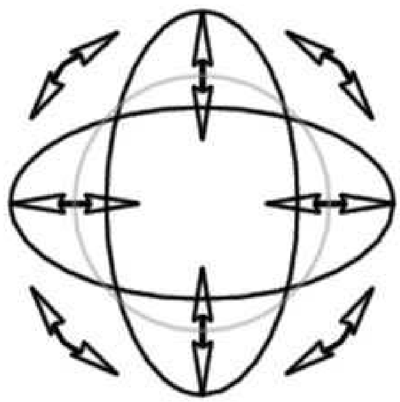


**Figure 6: The wavetrain closure on the rim of a wine glass and corresponding flexual waves as seen from the top (after [13]).**


**Figure 7: Path of circular mode on bowl (used with permission from [6]).**

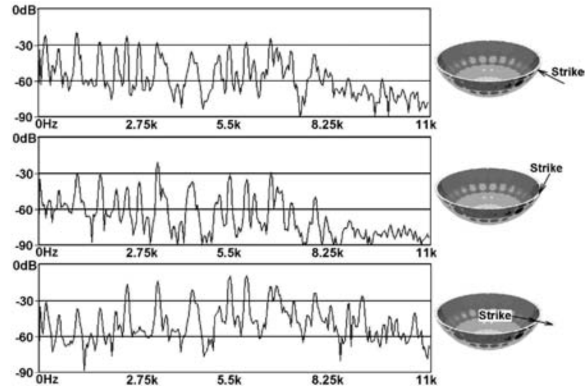



**Figure 8: Spectra of different excitations (used with permission from [6]).**

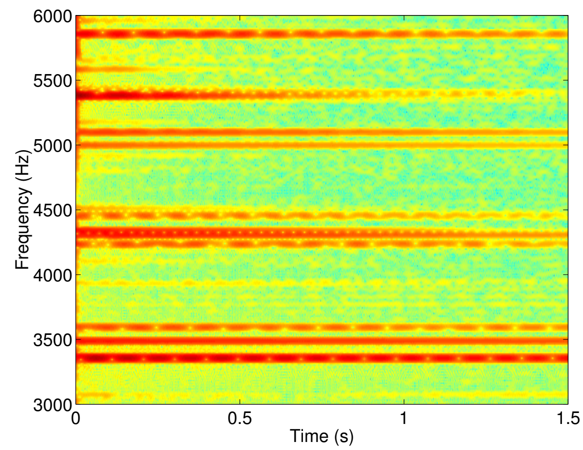



**Figure 9: Beating upper partials in spectrogram of a recorded Tibetan bowl.**

The measured spectra of the struck bowl can be seen in Figure 8 for various impact positions. As can be seen, there are a number of higher modes that lie close together yielding audible beating. The beating can be seen more clearly in Figure 9.

## 4.1 Beating Banded Waveguides

The beating modes combined with the very weak damping poses the main challange for modeling the dynamics using banded waveguides (as depicted in Figure 10.)

For two neighboring banded wavepaths whose center frequencies get close, the respective frequency-bands start to overlap strongly. This means that energy will contribute to traveling waves in both bands simultaneously. To guarantee stability within the frequency region the sum gain of both waveguides can not exceed unity as both are summed together for interaction or feedback. More specifically the gain of the respective banded wavepaths can be calculated from the maximum of the overlapping bandpass filter amplitude characteristics. This maximum has to be tuned to the desired gain, and the respective gain of the bandpasses is adjusted by the weight of the overlap. The resulting simulation of an isolated beating mode pair can be seen in Figure 11. The relative ratio between the modes is 1 : 1.05.

The beating modes following this construct, combined with plain modes then yields the complete simulation of the Tibetan bowl, which can be achieved with less than 20 banded wavepaths including beating mode-pairs.

## 4.2 Synthesis Parameters and Mapping

The complete synthesis model takes radial stick force and rubbing velocity as input. In addition an integration factor is included in the friction model that allows for artificial mode coupling. These are then mapped to sensor outputs. The sensors provide a measure of the radial stick force. In the first version, the sensor data was experimentally scaled and offset to achieve a playable region comparable to the subjective playable region of an actual bowl performance. The velocity is calculated from the time differential between peaks in the hall sensor output. Velocity is maintained throughout one rotation limiting the velocity responsiveness to slowly varying changes relative to average rubbing speeds. These are at about one revolution per 1 second. This value is use to control the rubbing velocity parameter of the model and corresponds to a physically accurate mapping. We have attempted to utilize the free sensor data of the accelerometer to control the integration value of the model. The integration value itself has no known physical interpretation and hence this mapping is quite arbitrary. In simulations that are meant to reproduce original bowl performance, this last mapping is usually disabled.

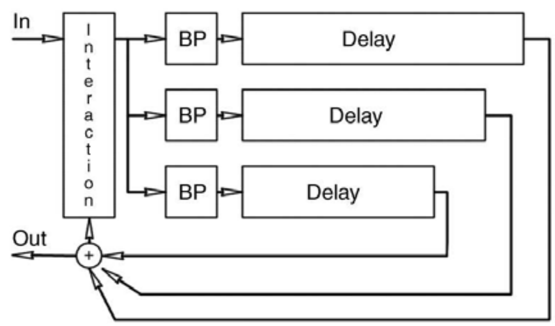



**Figure 10: A complete banded waveguide system.**

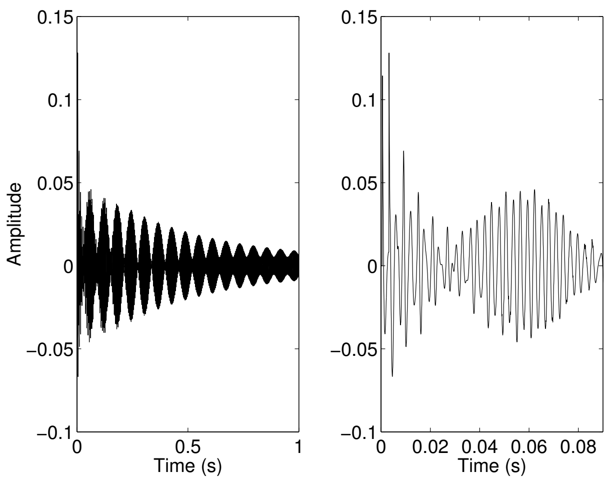



**Figure 11: Left: Evolution of an isolated simulation of a beating mode pair. Right: Initial transient and the first beating period.**

# 5. CONCLUSION

The HyperPuja provides a first controller design that maintains the physicality of the actual rubbing interaction and measures relevant physical parameters for the rubbing motion to be mapped to synthesis algorithms. Previous work either uses frictionless gesture [22] or EMG and position sensing technology [19] that each don't directly map to the original interaction type. While Tanaka and Knapp maintain the physicality of the rubbing gesture in performance, the sensed parameters don't have a physically close correspondence, which affects the mapping to parameters of physical models of the bowl. We see the maintenance of the original physical interaction and the sensing of the interaction-relevant parameters as a significant design advantage, because questions of force-feedback and responsiveness, engagement and intimacy, as well as performance familiarity and analogy are answered by maintaining the physical characteristics of the original behavior. At the same time the interaction gesture is decoupled from the physically sounding body. The interaction can now be performed on non-sounding objects and can be extended to arbitrary sound synthesis and manipulation algorithms. This design also hides technology and makes it unencumbering allowing the performer to focus purely on the physical interaction.

In future work, we will continue to extend the integration of the HyperPuja stick and the physical model of the Tibetan singing bowl. In particular the coupling of gesture and sensor data to synthesis model parameters will be investigated in more detail, also including a more direct unification of the data visualization and a more streamlined performance system. The modeling itself may need improved accuracy in the description of the friction coupling, and so related work in this area will be utilized [16].

We will also experiment with a version of the HyperPuja stick that includes a pressure sensor covering only part of the length of the playing surface, so as to provide a player with the performance option to excite the bowl acoustically using the hard wood surface. Maintaining the acoustic capabilities of the bowl itself and allowing the player to have the tactile feedback of the bowl's vibrations (transmitted to the hand and felt through the stick), while also providing the sound of a virtual bowl could produce a unique electro-acoustic performance experience.

In the next version, we will also experiment with additional sensing techniques for extended performance. For instance, it may prove interesting to measure the proximity of the stick to the bowl as well, or exploit different kinds of tilt of the bow stick using perhaps another accelerometer.

We will investigate the possibilities of using and perhaps making alternate kinds of bowls that will provide different form factors for playing and may be useful for exploring the limits of the physical model. In fact, we look forward to exploring the performance possibilities of bowing (rubbing with the HyperPuja stick) completely different kinds of structures and eliciting quite different sounds from the physical and non-physical synthesis models. Examples of this type include performance of the glass harmonica [8], the musical saw [15] and other friction based sound sources.

# 6. ACKNOWLEDGMENTS

Much gratitude to Yael Maguire for his technical advice and Egon Pasztor for his help with the software integration. Best thanks to James F. O'Brien for his rendered bowl images. Perry Cook for his figures and for earlier collaboration.

# 7. REFERENCES

[1] A. Akay. Acoustics of friction. *Journal of the Acoustical Society of America*, 111(4):1525–1548, April 2002.

[2] B. Bongers. Physical Interfaces in the Electronic Arts. In M. M. Wanderley and M. Battier, editors, *Trends in Gestural Control of Music*, pages 41–70. IRCAM, Paris, France, 2000.

[3] M. Burtner, S. Serafin, and D. Topper. Real-Time Spatial Processing and Transformations of a Singing Bowl. In *Proceedings of the 5th International Conference on Digital Audio Effects (DAFx-02)*, pages 123–126, Hamburg, Germany, September 26-28 2002.

[4] C. Cadoz and M. M. Wanderley. Gesture - Music. In M. M. Wanderley and M. Battier, editors, *Trends in Gestural Control of Music*, pages 71–94. IRCAM, Paris, France, 2000.

[5] P. R. Cook. Principles for Designing Computer Music Controllers. In *Proceedings of the CHI'01 Workshop New Interfaces for Musical Expression (NIME-01)*, Seattle, WA, April 1-2 2001. SigCHI. Available online at `www.csl.sony.co.jp/person/poup/research/chi2000wshp/papers/cook.pdf`.

[6] P. R. Cook. *Real Sound Synthesis for Interactive Applications*. A K Peters, Ltd., July 2002.

[7] G. Essl and P. R. Cook. Banded Waveguides: Towards Physical Modeling of Bowed Bar Percussion Instruments. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 321–324, Beijing, China, October 22-27 1999. International Computer Music Association (ICMA).

[8] G. Essl and P. R. Cook. Banded Waveguides on Circular Topologies and of Beating Modes: Tibetan Singing Bowls and Glass Harmonicas. In *Proceedings of the International Computer Music Conference (ICMC-02)*, pages 49–52, Göteborg, Sweden, September 16-21 2002.

[9] A. Hunt, M. M. Wanderley, and M. Paradis. The importance of parameter mapping in electronic instrument design. In *Proceedings of the 2002 Conference on New Instruments for Musical Expression (NIME-02)*, pages 149–154, Dublin, Ireland, May 24-26 2002.

[10] A. Hunt, M. M. Wanderley, and K. R. Towards a Model for Instrumental Mapping in Expert Musical Interaction. In *Proceedings of the International Computer Music Conference (ICMC-00)*, pages 209–212, Berlin, Germany, August 27-September 1 2000.

[11] F. Iazzetta. Meaning in Musical Gesture. In M. M. Wanderley and M. Battier, editors, *Trends in Gestural Control of Music*, pages 259–268. IRCAM, Paris, France, 2000.

[12] C. Nicols. The vBow: Development of a Virtual Violin Bow Haptic Human-Computer Interface. In *Proceedings of the 2002 Conference on New Interfaces for Musical Expression (NIME-02)*, pages 29–32, Dublin, Ireland, May 24-26 2002.

[13] T. D. Rossing. *Science of Percussion Instruments*, volume 3 of *Series In Popular Science*. World Scientific Publishing, 2000.

[14] J. B. Rovan, M. M. Wanderley, S. Dubnov, and P. Depalle. Instrumental Gestural Mapping Strategies as Expressivity Determinants in Computer Music Performance. In *Proceedings of Kansei - The Technology of Emotion Workshop*, Genova, Italy, October 3-4 1997. Available online at `http://www.ircam.fr/equipes/analyse-synthese/wanderle/Gestes/Exte%rne/kansei_final.pdf`.

[15] S. Serafin, P. Huang, S. Ystad, C. Chafe, and J. O. Smith. Analysis and synthesis of unusual friction-driven musical instruments. In *Proceedings of the International Computer Music Conference (ICMC-02)*, pages 57–60, Göteborg, Sweden, September 16-21 2002.

[16] S. Serafin, C. Vergez, and X. Rodet. Friction and Application to Real-time Physical Modeling of a Violin. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 216–219, Beijing, China, 1999. International Computer Music Association (ICMA).

[17] S. Serafin, C. Wilkerson, and J. O. Smith. Modeling Bowl Resonators Using Circular Waveguide Networks. In *Proceedings of the 5th International Conference on Digital Audio Effects (DAFx-02)*, pages 117–121, Hamburg, Germany, September 26-28 2002.

[18] A. Tanaka. Nime 02 - highlights from the talks, demos, and concerts. available online at `http://www.mis.atr.co.jp/~mlyons/nime02.html`, 2002.

[19] A. Tanaka and B. R. Knapp. Multimodal Interaction in Music Using the Electromyogram and Relative Position Sensing. In *Proceedings of the 2002 Conference on New Interfaces for Musical Expression (NIME-02)*, pages 43–48, Dublin, Ireland, May 24-26 2002.

[20] D. Trueman and P. R. Cook. BoSSA: The Deconstructed Violin Reconstructed. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 232–239, Beijing, China, October 22-27 1999.

[21] D. Wessel and M. Wright. Problems and Prospects for Intimate Musical Control of Computers. In *Proceedings of the CHI'01 Workshop New Interfaces for Musical Expression (NIME-01)*, Seattle, WA, April 1-2 2001. SigCHI. Available online at `www.csl.sony.co.jp/person/poup/research/chi2000wshp/papers/wessel.pdf`.

[22] C. Wilkerson, C. Ng, and S. Serafin. The Mutha Rubboard Controller: Interactive Heritage. In *Proceedings of the 2002 Conference on New Instruments for Musical Expression (NIME-02)*, pages 82–85, Dublin, Ireland, May 24-26 2002.

[23] C. Wilkerson, S. Serafin, and C. Ng. Physical Model Synthesis and Performance Mappings of Bowl Resonators. In *Proceedings of the International Computer Music Conference (ICMC-02)*, pages 219–221, Göteborg, Sweden, September 16-21 2002.

[24] D. Young. The Hyperbow Controller: Real-Time Dynamics Measurement of Violin Performance. In *Proceedings of the 2002 Conference on New Interfaces for Musical Expression (NIME-02)*, pages 65–70, Dublin, Ireland, May 24-26 2002.

# THE PIPE: Explorations with Breath Control

Gary P. Scavone
Center for Computer Research in Music and Acoustics
Stanford University
Stanford, CA 94305
gary@ccrma.stanford.edu

## ABSTRACT

*The Pipe* is an experimental, general purpose music input device designed and built in the form of a compact MIDI wind controller. The development of this device was motivated in part by an interest in exploring breath pressure as a control input. *The Pipe* provides a variety of common sensor types, including force sensing resistors, momentary switches, accelerometers, potentiometers, and an air pressure transducer, which allow maximum flexibility in the design of a sensor mapping scheme. *The Pipe* uses a programmable BASIC Stamp 2sx microprocessor which outputs control messages via a standard MIDI jack.

## Keywords

MIDI Controller, Wind Controller, Breath Control, Human Computer Interaction.

## 1. INTRODUCTION

*The Pipe*, shown in Fig. 1, is an experimental, general purpose music input device designed and built in the form of a compact MIDI wind controller. While acoustic wind instruments, as well as most existing commercial wind controllers, make use of dynamic air flow for activation, *The Pipe* is based on a "flow-free" breath pressure paradigm. The development of this interface was in part motivated by an interest in exploring the use and effectiveness of static-flow breath pressure as a control input. In addition, *The Pipe* provides a variety of common sensor types, including force sensing resistors (FSRs), momentary switches, accelerometers, and potentiometers, which allow maximum flexibility in the design of a sensor mapping scheme. The device uses a BASIC Stamp 2sx microprocessor which can be programmed via a serial interface to a computer and outputs control messages via a standard MIDI jack.

**Figure 1: *The Pipe*: View from above (top) and below (bottom).**

A nearly complete version of *The Pipe* was constructed more than two years ago but then left unfinished when size and space complications arose during final assembly. The impetus to continue work on the device, which ultimately led to a complete rebuild, occurred in the course of music composition experiments with real-time physical modeling algorithms.

## 2. INTERFACE CONCEPT AND DESIGN

*The Pipe* is intended to function either as an independent MIDI controller or as an interface to computer-based synthesis algorithms. During its design, several goals were identified, including:

- the ability to use and experiment with static-flow breath pressure as a control input
- to provide a wind-like control surface for use with existing woodwind tonehole synthesis models
- to provide a more hygienic breath pressure interface for device sharing
- to allow precise variation of controller values
- to make use of as many different sensor types as could reasonably be located on or within its structure
- to be housed within a durable, compact shell

In particular, *The Pipe* is meant to operate in place of several previous experimental digital interfaces built by this author [4, 5], allow generic control of most Synthesis ToolKit in C++ (STK) instruments [2], provide an expressive interface for wind instrument performance control, as well as offer flexibility for future synthesis model control and input sensor developments.

*The Pipe* was conceived as a standalone, battery powered device which would provide a set of finger "keys" in a configuration akin to a musical recorder, as well as contain all necessary circuitry to process the sensor data and output MIDI-formatted control messages. A housing was fabricated from 1.5 inch (3.8 centimeter) inner diameter Acrylonitrile Butadiene Styrene (ABS) piping, cut to a length of approximately 14 inches (35.5 centimeters). A small circuit board was cut so that it could be slid in and out of *The Pipe* along a set of grooves at its downstream end. This layout is diagrammed in Fig. 2. All necessary wiring and electronics are contained within the ABS body to maintain maximum durability.

The finger keys or switches on all commercial wind controllers provide only limited, binary state information. When

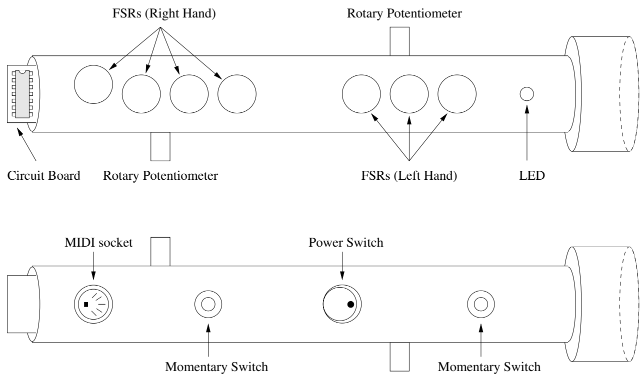



**Figure 2: *The Pipe* layout diagram: View from above (top) and below (bottom).**

a key is depressed or touched, a new MIDI "Note On" message is produced which corresponds to some predefined fingering/note number mapping. A previous controller by this author [4] used FSRs in conjunction with finger keys to enable a 7-bit range of finger position data. *The Pipe* was designed so that the fingers rest directly on or above FSRs. A set of seven finger depressions were drilled along the length of *The Pipe* in a traditional two-hand arrangement. Circular force sensing resistors of 3/4 inch diameter were positioned in the depressions and slots were chiseled for the FSR leads to facilitate an internal wiring system. Initially, the FSRs were covered with only a thin layer of tape and direct finger pressure was applied to them. Later, compressible, foam-like pads were added on top of the FSRs to provide some haptic feedback in the fingering mechanism.

Traditional wind instruments are driven by dynamic air flow through an acoustic air column. Most commercial digital wind instrument controllers have made use of similar breath control schemes (Yamaha WX5, Akai EWI). In an electronic wind controller, this air flow is completely unnecessary and even undesirable given potential complications involved with humid air flow through or near electronic components. In addition, the limited capacity of the human lungs requires that outgoing air flow, and the resulting pressure, be periodically stopped. Pressure sensing in *The Pipe* was therefore based on a static air flow paradigm within an air tight enclosure on the upstream end of the instrument. A removable, contoured mouth cap was designed to be positioned against the performer's face but beyond the mouth and lips to minimize hygienic concerns that might arise when sharing the device. In this scheme, an air-tight seal is formed between the player's face and the cap which allows pressure to be maintained and controlled indefinitely inside the cap while breathing normally through the nose. The mouth cap was fabricated from a short, ABS coupling section.

A variety of additional sensors are included in *The Pipe*. An earlier controller by this author [5] provided tilt sensing in two dimensions using a dual-axis accelerometer. Experience with that device indicated that tilt offers a convenient control parameter that is especially easy to use concurrently with other input sensors. In addition, accelerometers are naturally suited for sensing gestures appropriate when controlling shaker synthesis algorithms [1]. Two rotary potentiometers allow for small or subtle control value variations, as well as a means for setting values which are intended to remain fixed until further modified. These controls were added to address limitations with the use of FSRs, which are difficult to use for precise control and which require continuous action on the part of the player to maintain a non-zero value. Two momentary switches were provided for use as triggers or in conjunction with other sensors to create more complex control schemes.

Given the variety of possible applications, a programmable microprocessor interface was required. *The Pipe* was built using a Basic Stamp 2sx microprocessor by Parallax, Inc.

## 2.1 Hardware Details

The seven FSRs are arranged to be used by the first three fingers of the upper left hand and the four fingers of the lower right hand. A momentary switch and rotary potentiometer are located in proximity to each of the finger "banks". The system was balanced in such a way that it could be easily played with either hand independently.

The electronic components were mounted on a circuit board which was cut so that it could be slid into the downstream end of the ABS housing. The FSRs (Interlink Electronics), switches, potentiometers, MIDI socket, and LED were mounted on the ABS housing (see Fig. 2) and connected to the circuit board via ribbon cables. The BSIIsx is powered by a 9-volt battery, which is housed inside *The Pipe* just below the detachable mouthpiece.

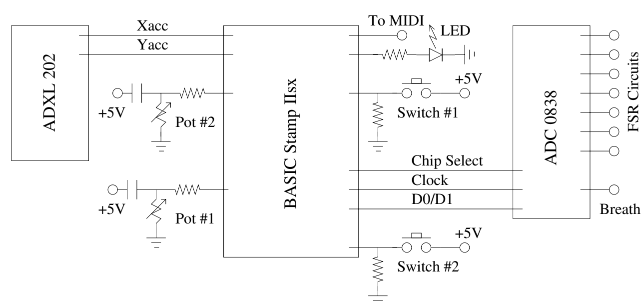



**Figure 3: *The Pipe* circuit schematic.**

The circuit board schematic is shown in Fig. 3. The dual-axis accelerometer, an Analog Devices ADXL202, provides digital output and interfaces to the BSIIsx via two of its sixteen input/output pins. The potentiometers and momentary switches are connected via one input pin each. The MIDI interface jack uses one output pin. The seven FSRs and the air pressure sensor are read through an 8-channel, 8-bit serial I/O multiplexing analog-to-digital converter (National Semiconductor ADC0838), which connects to the microprocessor using three pins. Currently, a Motorola MPXV5010 air pressure sensor is being used with a range of 0 to 1.45 PSI. In the future, a more sensitive device may be substituted. Finally, an LED was included via another pin as a visual feedback mechanism to distinguish program features. In its current form, five input/output pins remain for future modifications or upgrades.

A single MIDI cable is necessary to connect with an external synthesizer or computer. A removable, four-pin connector provides a serial interface to a computer for microprocessor programming. The BASIC Stamp makes use of a simplified and customized form of the BASIC programming language called PBASIC. With the BSIIsx processor, up to eight programs can be stored in 2 Kbytes of memory each. The scaling of sensor data for 7-bit MIDI message output is left completely to the programmer.

## 2.2 Sensor Mapping

As configured in *The Pipe*, the FSRs and breath pressure sensor produce a non-zero output only when activated by finger or breath pressure. This makes them appropriate in situations where the sensor is used to produce a control gesture which always begins and ends at the same equilibrium, non-active value. This behavior mimics the spring-loaded keys of wind instruments. However, sensors of this type are less well suited for situations where one wishes to make modifications, perhaps precise, above and below a mid-range value. The response of the FSRs in particular is highly non-linear and in the current configuration has much greater variation at low pressure values. The momentary switches function in a similar manner to the FSRs but produce only two possible states.

The rotary potentiometers, on the other hand, provide a good means for making small control value changes about a non-zero mean, as well as settings which maintain their state. The accelerometers, when measuring tilt, can be used in a similar way though the usable range in hand-held situations such as this is limited to approximately 16 or less distinct, controllable positions (approximately 4-bits of resolution).

While each sensor type presents particular constraints, it is possible to use combinations of sensors to achieve greater flexibility and augment the raw capabilities of the device. For example, if one wishes to use *The Pipe* in a situation that calls for six "slider-like" controls, the FSRs can be used as "switches" to activate particular control parameters for modification by a nearby rotary pot. Another useful combination involves the use of a momentary switch to activate modifications made using FSRs or the breath pressure sensor. In this way, a control value can be positioned using pressure and then held at a given value by releasing the momentary switch *before* releasing the pressure. Combinations using the dual-axis accelerometer are possible as well. Finally, the control interface provided by *The Pipe* has limits, particularly in situations where a large number of control values need to be modified simultaneously. When using the device to drive a real-time computer synthesis engine, such manipulations can be made into preprogrammed functions on the computer which are simply triggered by the controller.

Maximum flexibility with *The Pipe* is achieved by offering a variety of conveniently located sensors which can be controlled using one or two hands and which can be configured and programmed as desired for a given situation.

Another aspect of parameter mapping concerns the way control values are mapped to synthesis parameters [3]. A few example mapping schemes of this sort are considered in the following section.

# 3. MUSICAL APPLICATIONS

*The Pipe* was designed for a variety of specific uses, as well as made generic enough to function in as-yet unknown situations. One goal was to have an instrument which could function in place of two previous experimental controllers. Another goal was to create an instrument for use as a real-time performance controller. This section documents these applications.

## 3.1 A Tonehole Controller

Physical modeling research reported in 1998 led to an efficient, real-time model of woodwind instrument toneholes [6]. In conjunction with this research, a MIDI wind controller was designed and constructed to provide a playable, intuitive interface for the model [4]. Of particular interest was a finger sensor which could provide a range of position data for mapping to a range of tonehole states between open and closed extremes. The seven FSRs of *The Pipe* can be configured to provide this functionality.

As a tonehole controller, *The Pipe* provides MIDI formatted messages to a computer-based synthesis engine running a real-time implementation of the tonehole model. Custom MIDI control values for finger pressure were designated for each of the toneholes. Additional controls, including breath pressure, breath noise, and register hole state, are configured from the available sensor inputs. The one-to-one mapping of finger pressure to tonehole state provides an intuitive interface to the model. In some instances, however, it becomes difficult to consistently maintain the pressure necessary on all FSRs to prevent inadvertent upstream hole "leakage". This problem points to an inadequacy with the *Pipe's* finger "key" mechanism which will be investigated in the future. In response to this difficulty, an alternative binary position scheme was programmed such that light finger pressure triggers tonehole closure, while a potentiometer is used to vary the rate of finger hole closure.

## 3.2 A MIDI Sequencer

Devices making use of programmable microprocessors can easily be configured as simple MIDI sequencers for use with external MIDI synthesizers. In general, a recurring pattern of notes, or ostinato, is programmed as a loop and subdivisions within that interval are denoted for possible sensor-controlled events. Applications of this type were previously explored using the *Phoney Controller* [5] and are easily repeated and extended with *The Pipe*.

Flexible, improvisatory control mappings within a constrained musical "scene", as created by the sequence, can provide hours of entertainment for musicians and non-musicians alike. Force sensing resistors function well as volume controls for distinct voices and auxiliary rhythm section instruments. Potentiometers are convenient for controlling tempo or selecting voices. Within a fixed modal harmonic scheme, a scale and/or octave changes can be mapped to tilt sensors to provide an easily mastered, "no fail" improvisatory control surface.

## 3.3 A Synthesis ToolKit Interface

The Synthesis ToolKit in C++ (STK) is a flexible set of open-source audio signal processing and algorithmic synthesis classes written in the C++ computer programming language [2]. STK provides a standardized control interface for most of its synthesis algorithms. Configuration of *The Pipe* for generic control of STK algorithms involved a variety of sensor mapping issues.

In most STK algorithms, a control parameter is provided for energy input to the instrument. For wind instrument algorithms, this parameter corresponds to breath pressure, for shaker instruments it corresponds to shake energy, and in other cases it is a volume control. Given the array of sensors on *The Pipe*, outputs from both the air pressure sensor and the accelerometer were mapped to the energy parameter. While providing intuitive control mechanisms, this mapping also allows one to "blow" shaker instruments or "shake" wind instruments.

Most of the remaining STK algorithm parameters are designed to be modified both above and below default values. A mapping strategy to allow this type of control was developed as previously discussed, such that FSRs are used as "switches" to activate particular control parameters for modification by a rotary pot. In this way, the potentiometer decrements or increments a parameter value selected by a corresponding FSR and this value is saved for subsequent adjustment. The other rotary potentiometer is used to make program changes. In addition, the two momentary switches are used to trigger "Note On" and "Note Off" events.

The primary difficulty observed while using *The Pipe* with STK algorithms has been the lack of a visual mechanism for determining existing parameter settings. Given that a single rotary potentiometer, in conjunction with distinct FSRs, is used to modify a number of different parameters, it becomes difficult to remember the last value setting for a particular parameter. Also, this scheme allows only a single parameter to be adjusted at a time. In a performance situation, a more responsive mapping would likely be necessary.

### 3.4 An Expressive Performance Controller

*The Pipe* was designed in part to allow expressive and subtle control of real-time physical models in a performance setting. Explorations in this context are ongoing though several mapping strategies are evident. Breath pressure control offers an intuitive means for "energizing" systems continuously driven by air or bows, such as wind instruments or bowed strings, bars, or bowls. While struck or plucked systems can be triggered using momentary switches, a more natural mapping can make use of velocity-based physical gestures which are calculated from the accelerometer inputs. Static tilt can be appropriately mapped to parameters which are typically varied above or below a default value for relatively short periods of time.

Subtle performance control with real-time physical models demands the ability to make minute parameter adjustments, sometimes simultaneously across several different parameters and/or instruments. When interfacing to a computer-based synthesis environment, some of these demands can be transferred to the synthesis system and simply triggered by the controller to achieve results beyond the capabilities of the device or performer.

## 4. OBSERVATIONS AND FUTURE WORK

Musical activities with *The Pipe* are continuing and possible improvements are actively being explored. Changes to the current breath pressure mechanism are being considered to provide a more air-tight fitting between the mouth cap and player's face, as well as more sensitivity and resolution from the sensor. In addition, finger position sensors with greater sensitivity and better tactile characteristics are desirable. In their current form, there is no way of knowing (without extensive practice) what amount of finger pressure represents full "deflection" of the finger sensors.

In its existing configuration, it is difficult to make adjustments to the rotary potentiometers when both hands are positioned on the finger "keys". It is unlikely that an appropriate modification can be made to the existing structure, though a future improvement could instead make use of a thumb operated "roller" mechanism.

The addition of a small microphone in the mouth cap to record vocalizations has been contemplated, though this would require an extra audio connector. At this point, a lip pressure sensor has not been considered, though input/output pins are still available on the microprocessor for future sensor extensions.

## 5. ACKNOWLEDGMENTS

This author would like to thank Perry Cook for early music controller inspirations and suggestions. A number of individuals on the *Wind Controller Mailing List* provided helpful information about existing commercial wind controllers as well as feedback on wind controller design issues. And discussions with Bill Verplank at CCRMA helped motivate a return to this project after more than a year of inactivity.

## 6. REFERENCES

[1] P. R. Cook. Physically informed sonic modeling (phism): Synthesis of percussive sounds. *Computer Music Journal*, 21(3):38–49, 1997.

[2] P. R. Cook and G. P. Scavone. The Synthesis ToolKit (STK). In *Proceedings of the 1999 International Computer Music Conference*, pages 164–166, Beijing, China, 1999. Computer Music Association.

[3] A. Hunt, M. Wanderley, and M. Paradis. The importance of parameter mapping in electronic instrument design. In *Proceedings of the NIME-02 Conference on New Interfaces for Musical Expression*, pages 149–154, 2002.

[4] G. P. Scavone. The Holey Controller, 1998. http://www-ccrma.stanford.edu/˜gary/.

[5] G. P. Scavone. The Phoney Controller, 1999. http://www-ccrma.stanford.edu/˜gary/.

[6] G. P. Scavone and P. R. Cook. Real-time computer modeling of woodwind instruments. In *Proceedings of the Internation Symposium on Musical Acoustics (ISMA-98), Leavenworth, WA*, pages 197–202, June 1998.

# The STRIMIDILATOR: a String Controlled MIDI-Instrument

Marije A.J. Baalman
TU, Berlin, Germany
marije@baalt.nl

## ABSTRACT
The STRIMIDILATOR is an instrument that uses the deviation and the vibration of strings as MIDI-controllers. This method of control gives the user direct tactile force feedback and allows for subtle control. The development of the instrument and its different functions are described.

## Keywords
MIDI controllers, tactile force feedback, strings.

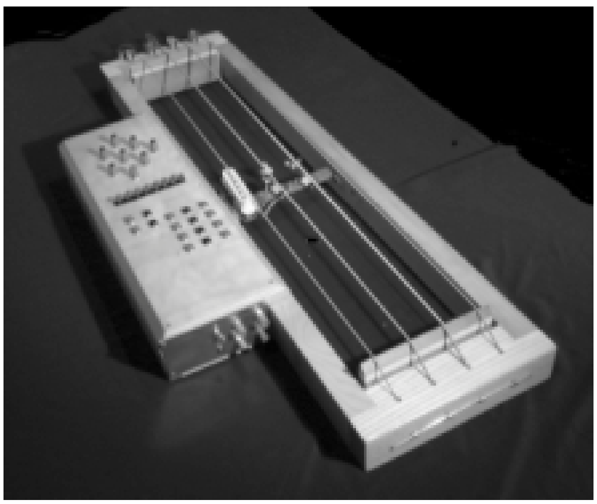


**Figure 1. The STRIMIDILATOR**

## 1. INTRODUCTION
In acoustical instruments there is a direct connection between the interface, i.e. that which the musician manipulates, and the creation of the sound; for example in a string instrument the musician lets the string vibrate and the vibration of the string is amplified by a resonating body. In electronic instruments this connection between interface and the resulting sound is not direct. Here there needs to be a conversion from mechanical movement to an electronic signal, which goes through an electronic circuit and finally is converted back into a mechanical movement of a loudspeaker to produce the desired sound.

The objective for the development of the STRIMIDILATOR (see figure 1) was to create an interface between the mechanical and electronic world, which has an intuitive feel to it, which allows the musician to use subtle hand movements for control and gives the musician tactile force feedback of the instrument (in addition to the auditory feedback that he gets).

The objective was also to create an interface that could be used for various (existing) electronic instruments, therefore it was decided to choose for a conversion to MIDI control signals as the main output of the interface.

The development was carried out as a project during my attendance of the Sonology Course at the Royal Conservatory of The Hague in The Netherlands, from March to September 2002. Further development is still ongoing.

## 2. CONCEPT
The vibration of strings is an ancient way in which musical instruments make sound. As such a large amount of know-ledge has been built up of the mechanical construction and of the practical use of string instruments over the last millennia.

The string in itself provides a good interface for a user. It is easy to manipulate, the effect of the manipulation can easily be seen and felt. A string under tension can either be brought into vibration or be held and given a deviation. Both the vibration and the deviation of the string can be used as controllers.

In the past, various other attempts have been made to use string instruments as the basis for controllers for electronic music. Cutler [1] gives an overview of commercially available MIDI string instruments. All implement pitch-to-MIDI conversion in one way or another; some also implement MIDI controllers depending on pick-position, string vibration envelopes and pitch bend. The vibration of the string is in all cases simplified to a pure MIDI-note, not taking into account the more complex characteristics of the vibration.

The VideoHarp (Rubine & McAvinney [2]) is an example which uses the interface of a harp by using the positioning of the fingers for parameters. Though the writers stress the importance of tactile feedback, this feature is only marginally implemented, through a dependency on how hard a finger is pressed against a plate. The VideoHarp does not use actual strings in its design; as an advantage this has that pitch control can be more continuous; a disadvantage is that the tactile feedback from the strings is missing.

The Web of Michel Waisvisz (see Krefeld [3] or Bongers [7]) uses the tactile force feedback of strings. In this it is quite similar to the STRIMIDILATOR; the difference is that The Web is designed with the aim to change many parameters at once, by creating a large interdependency between all used strings. In the STRIMIDILATOR one parameter is changed at a time and the musician can choose to control more than one parameter with one hand.

### 2.1 Vibration of strings
A vibrating string can of course be used directly to create sound, but this was not the objective of development. Instead, the vibration of the string can be used as a complicated kind of oscillating control parameter. A string that is brought into vibration produces a series of harmonic frequencies, based on its length, thickness and tension. Additionally, due to the mechanical construction (or faults therein) the vibration is never completely harmonic and has some irregularities. Taking all this into account, the vibra-

tion of a string makes a good source for a complex oscillating control parameter, which has a very intuitive interface.

A second way of using the vibration of the string is by following its envelope. In this way the frequency spectrum of the vibration is not used, but the decay of the vibration is used. Depending on the speed of the envelope follower irregularities in the decay are detected or not. This type of control allows for a slowly changing parameter, that can be easily (re)triggered by the musician, giving it the desired start amplitude.

## 2.2 Deviation of strings

By pulling or pushing the string and holding it, the string is given a deviation. This deviation can be used as a parameter for the sound to be created. As an interface this pulling and pushing of a string works quite well. The tension of the string makes that it wants to go back to its rest state; the harder you pull or push the string, the stronger it pulls or pushes back. By pulling or pushing at a different location on the string, the force feedback is different; it is easier to pull a string in the middle, than that it is near its ends, where it is fixed to the body of the instrument. This gives the user the choice where to manipulate the string. If he wants to make grand movements, he can choose to push on the string in the middle, where the force needed is minimal. If he wants to make small changes he can push near the ends, where more force is needed to get a more subtle control.

# 3. DESIGN

The design of the instrument can be divided roughly into three aspects: the mechanical, the electro-mechanical and the electronic design. Of course, each aspect has influence on the other two.

## 3.1 Mechanical design

For the mechanical design there were a couple of demands that needed to be fulfilled:

- The structure should be able to bear the tension of the strings
- The construction should contain the electronic circuits at some place
- The instrument should be comfortable in use and should be durable
- The instrument should look good

For the frame to bear the strings, a wooden construction was chosen. Ash wood was chosen as it is strong, durable and looks good.

The frame was constructed as shown in Figure 2. This way of construction is very robust. The wooden parts, once glued together, cannot glide away.

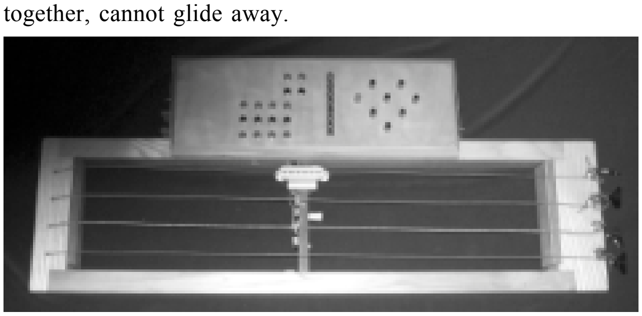


**Figure 2. Frame structure of the instrument.**

Connected to the frame a little box was created, which contains the electronic circuits of the instrument. As we used the MicroLab technology (see below), it was possible to add more controllers on the instrument and we had to create space for these additional functions as well. Therefore, we created space on the box for extra knobs, switches and buttons. On the sides of the box room was made for the connection of various cables, some tuning knobs for the electronics, a MIDI channel switch and the on/off switch.

The box's frame and the top were made of wood, for constructional and esthetic reasons. The bottom and the sides of the box were made of aluminium, in order to reduce electromagnetic interference of the outside world with the electronics inside.

In order to connect the electronic sensors used for the strings to the electronics inside the box a little hole was made into one of the sides of the frame to pass cables through.

## 3.2 Electro-mechanical design

The main problem in designing such an instrument is finding the best way to transfer the mechanical movement into an electronic signal.

### *3.2.1 Deviation*

For detecting the deviation a choice was made for a linear transducer. This is a variable resistor that has a little pin that can move up and down, thus determining the resistance value. Using a little bus and a small elastic band the pin can be connected to the string, as shown in Figure 3.

The box of the resistor was fixed to the frame at the small wooden bar in the middle (fig. 2) in such a way that it could rotate at the connection point, so that the user would have as much freedom in moving the string as possible.

Connecting the linear transducer however influences the behavior of the string. Initially the linear transducers also contained a spring inside, which caused the transducer to pull at the string. As this was an undesired effect, I removed this spring from the transducer. Even with the spring removed, the transducer still influences the string in such a way that the string can no longer vibrate freely. The transducer provides so much damping to the system, that the string, when it is released, goes back to its rest state, without vibration.

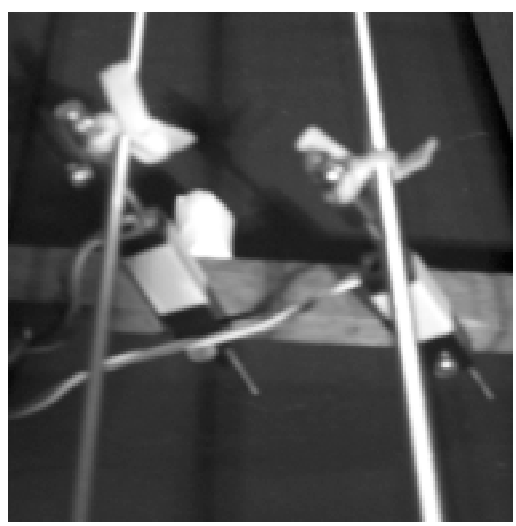


**Figure 3. The linear transducer and the attachment to the string with a little bus and an elastic band.**

#### *3.2.2 Vibration*

For detecting the vibration of the string, another solution had to be found. I chose for a conventional way of translating vibrations into an electronic signal: a coil that one can find on any electric guitar.

As a common guitar coil only gives one signal for all of the six strings together, the choice was made to use a coil for each string. They were placed on metal crossbars that were placed parallel to the strings across two small wooden bars in the middle of the frame (figure 2 shows only one of the wooden bars, the second one and the metal crossbars were added after the photograph had been taken); as the coils are magnetic they stay fixed, but can still be easily moved to allow the user to choose on which place it should pick up the string's movement.

The mechanical construction was made to attach four strings, so a choice had to be made for how to use each string. In the end, to two strings linear transducers were attached and for the other two strings two coils were used.

To avoid influence of one coil on the other, we placed the coils on the two outer strings and attached the linear transducers to the two inner strings. Having the two linear transducers in the middle also has the advantage that one can manipulate both strings with one hand and even using the same hand to trigger the vibrating strings.

## 3.3 Electronic design

The main demand for the electronic design was that the instrument should output signals that could be used directly, without needing further processing by a computer.

As the Royal Conservatory, aided by STEIM, had already developed a general device, the MicroLab [5], for the translation of sensor data into MIDI-data, it was decided to use a modified version of this device for the STRIMIDILATOR. The MicroLab is a cheaper, one-time-only programmable version of the SensorLab of STEIM [6].

The SensorLab was used to prototype the programming of the software in order to discover the best way of translating the MIDI-data. This had as a drawback that the program-code made in Spider (SensorLab's programming language) had to be translated manually into assembly code for the MicroLab (a PIC16F873 chip), which may not have been the most optimal way of programming. As some additional electronic circuits were made, based on the power supply of the SensorLab, for the eventual version of the instrument a power conversion circuit had to be made also.

Eventually the software used the direct sensor data of the linear transducers (with only a condensator placed in parallel in order to limit the influence of noise) and the output of the coils after an electronic envelope follower. The envelope follower can be tuned in its speed and in its amplification of the signal.

In addition to the MIDI-output, the direct output of the coils was given as a (low-frequency!) audio output via a stereo-jack-connection. After amplification these signals could directly be used as control input, provided that the output is sent to a device that can use an analog input as such.

## 3.4 Extra functions

As the choice for the MicroLab technology gave room for additional functions, this opportunity was seized by choosing to add buttons for MIDI-note messages, knobs for additional controller messages, analogue inputs for input of other sensor data and a set of switches for switching between functions of components. The hardware components for these were integrated in the design of the box, where esthetic and ergonomic considerations determined the placing of the knobs, switches and buttons. The buttons are laid out in a pattern which allows the user to be able to reach all of the buttons without moving the hand too much, thus being able to cover 16 semi-tones (see Figure 4).

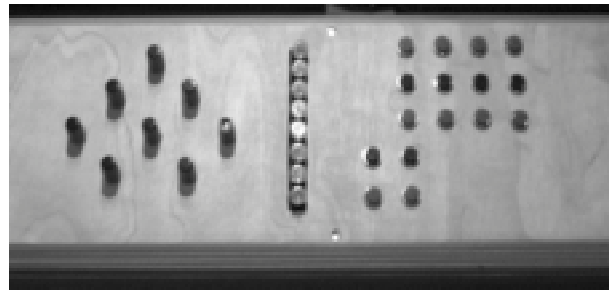


**Figure 4. Button and knob layout on the box of the STRIMIDILATOR. On the left are the knobs, the lowest two are for the note-on and note-off velocity. In the middle are the switch buttons for the various modes. On the right are the note-on/off buttons.**

## 3.5 Software design

The software determines in which way the sensor data was translated into MIDI-data. The design question here was: what are the most interesting ways to translate the sensor data? The design was carried out with the notion that the eventual function of the controller output could be determined in the receiving MIDI-instrument, that is: no care was taken to consider the standardized MIDI-controller number functions. There was taken into account that one may want to receive MIDI on different channels: a switch was assigned to change between sending all data on one channel, or about half of the data on one and the other half on another channel.

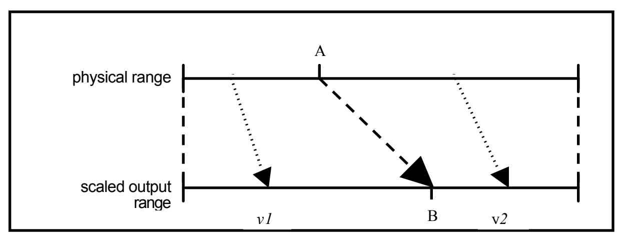



**Figure 5. Dynamical mapping. When switching from one mode to another the last value (B) of the controller is put into memory and the next time the mode is entered, the controller value takes this value as a starting point; the physical value A is mapped to B. While the instrument is in this mode, all other values are mapped accordingly. For example, value *x1* is mapped to *y1* and *x2* is mapped to *y2*.**

For the linear transducers we found three ways to translate the sensor data to MIDI:

1. Directly, that is a direct linear mapping between the resistance value and the MIDI controller value.
2. Through a software-implemented envelope. This function slowly follows the changes of the controller, or allows the user to suddenly "jump" to another value. A timer ensures that eventually the value goes back to zero.

3. Dynamical mapping. This means that when switching from one mode to another the last value of the controller is put into memory and the next time the mode is entered, the controller value takes this value as a starting point. The remaining physical range is then mapped between 0 and 127 according to the new mid-point. This is clarified in Figure 5.[1]

For the coils we used the first two functions mentioned above, both using the signal from the coil, after it passed through the envelope follower implemented as an electronic circuit. This allowed for a choice between a fast and slow envelope of the vibration to be used.

Eight switches determine the mode in which each sensor input is translated. The first two determine as a two bit-number the function (where no switch pressed means that the sensor is not used and no controller data is sent out) for the first linear transducer, the second two do in a similar way for the second transducer. The next two determine which function for each of the two coils is used. The last two determine whether a coil is on or off. Each mode sends out a different MIDI control number.

The resistances of the potentiometers of the knobs were directly and linearly translated into MIDI control data. Two of the knobs were used to determine respectively the note-on and note-off velocity of the buttons, which were mapped to send out MIDI note-messages: a note-on message when the button was pressed and a note-off message when the button was released.

Finally the sensor data coming in from the eight sensor inputs is translated directly, linear to MIDI controller data.

An overview of all the functions is given in Table 1. A photograph of the resulting instrument is shown in Figure 1.

## 4. PRACTICAL EXPERIENCES

The STRIMIDILATOR has been used in two performances so far.

The first performance was in Theater Kikker in Utrecht, the Netherlands, where the author used the instrument in the second half of a 20-minute improvisation.

The STRIMIDILATOR was connected to a Clavia Micromodular [7], both through MIDI and (via a mixer for amplification) through the audio-input.

The audio-input (providing the coil output) was used as a kind of LFO to control the navigation through vowels of a vocal filter. The envelope of the signal was tracked simultaneously to use as the envelope of the resulting sound. This allowed for a direct control by the user of the volume of the sound, as well as of its character.

The direct MIDI-translation of one of the linear transducers was used to determine the frequency of a noisy wind like breathing sound, while the same mode of the other linear transducer was used to navigate through vowels of the vocal navigator through which this sound was coming. For the first (the frequency control), this proved to be very intuitive and handy: it was possible to move slowly from low frequencies to higher ones by slowly moving the pressure on the string from the middle to the edge (and vice-versa for the opposite effect). By using one hand to pull the two strings up and together or let them release, there was a good control over the two parameters of the sound at the same time, also allowing for quick deviations made by quick movement of the fingers in pulling and releasing the strings.

In mode 2 (the envelope follower), the transducers were used respectively to change the density of a ticking sound

**Table 1. Overview of functions of the STRIMIDILATOR built.**

| Controller | # | Mapping | Output |
|---|---|---|---|
| Bass string – linear transducer | 2 | Direct, envelope, dynamic | MIDI ctr. 0-5 |
| Bass string – coil | 2 | Direct,<br>Envelope (with or without software envelope) | Analogue MIDI ctr. 6 – 9 |
| Switches | 8 | Function mapping of strings | - |
| Switch | 1 | MIDI channel switch | Ch. 1-2 |
| Buttons | 16 | Push: Note on<br>Release: Note Off | MIDI note nr. 60-75 |
| Knobs | 6 | Direct | MIDI ctr. 10-15 |
| Knobs | 2 | Note on/off velocity | - |
| Inputs | 8 | Direct | MIDI ctr. 17-23 |

and the frequency of the vocal filter that was controlled by the audio input. Two of the knobs were used to control the frequency and the timbre of the ticking sound. The note-values were only used to transpose a melody up and down. Other functions were not used during the performance.

The second performance was on the festival *"Ver uit de Maat"* in WORM in Rotterdam, the Netherlands, during a 25-minute improvisation.

The direct MIDI-translation of the linear transducers was used to control the frequency of an engine-like sound, panned to the left for one string and to the right for the other. By pulling and pushing both strings with one hand, one could easily create a changing soundscape between the left and right signals.

For one string the envelope follower of the linear transducers was used to control the frequency of an FM-signal, which was modulated by the vibrating string; the amplitude of the vibration determined the amplitude of the resulting sound. In this way it was possible to control the frequency of this signal with one hand, while creating the sound with the other.

The envelope follower of the other linear transducer was used to control the density of clicks, enabling the user to jump between different densities.

During the performance it was also noted that the instrument either should be heavier or should be fixed to the table it is lying on, as while pulling on the strings, the instrument was lifted a little.

On both occasions the deviation of the strings proved to be very intuitive for the control of sound parameters. The dynamical mapping could not yet be tested, but experience with the direct mapping made clear that dynamical mapping, once implemented, should prove to be very useful. The switch between modes sometimes had as a result that the sound suddenly changed when returning to a mode, which was not always desired.

[1] At the moment of writing (10th of April 2003) this function does not yet work in the eventual implementation. Corresponding assembly code remains to be written.

As can be noted, the envelope follower of the string vibration was not yet used in a performance, due to the fact that the function did not work properly yet. This has now been fixed.

It was noted during the testing of the instrument that the vibration of the string caused the other controllers to become less stable. By rewiring the ground and decreasing the length of the wires, this problem was solved.

## 5. CONCLUSIONS

The STRIMIDILATOR provides a good interface for controlling parameters for electronic music. The use of the deviation of strings ensures that good tactile force feedback is given to the artist using the instrument and the strings allow for a subtle and versatile way of control. The implementation for the use of the envelopes of the vibrating strings still needs to be tested.

Further application and experience with the instrument will show on which other points the instrument can be improved.

## 6. ACKNOWLEDGMENTS

To Frank Baldé of STEIM, Amsterdam, for the aid in developing the prototype of the instrument.

To Paul Beekhuizen[2], professional instrument builder from The Hague, for the advice in the mechanical construction of the instrument.

To Lex van der Broek and Jo Scherpenisse of the technical department of the Royal Conservatory in The Hague for the electro-technical and -mechanical aid and programming of the MicroLab.

[2] See http://www.paulbeekhuizen.nl

## 7. REFERENCES

[1] Cutler, M., *The World on a String*, Electronic Musician, May 1, 2001, http://emusician.com/ar/emusic_world_string/

[2] Rubine, D. & McAvinney, P., *Programmable Finger-Tracking Instrument Controllers*, Computer Music Journal, Vol. 14, No. 1, Spring 1990, pp. 26-41

[3] Krefeld, V., *The Hand in The Web: an interview with Michel Waisvisz*, Computer Music Journal, Vol. 14, No. 2, Summer 1990, pp. 28-33

[4] Bongers, B., *The Use of Active Tactile and Force Feedback in Timbre Controlling Electronic Instruments*, ICMC 1994

[5] MicroLab, Royal Conservatory of The Hague, The Netherlands, http://www.koncon.nl. For specific information on the MicroLab send an e-mail to lex@koncon.nl. The MicroLab is not a commercial product.

[6] SensorLab, STEIM, Amsterdam, The Netherlands, http://www.steim.nl

[7] Nord Modular, Clavia, Sweden, http://www.clavia.se

# Microcontrollers in Music HCI Instruction: Reflections on our Switch to the Atmel AVR Platform

Scott Wilson
CCRMA, Department of Music
Stanford University
Stanford, California USA
rswilson@ccrma.stanford.edu

Michael Gurevich
CCRMA, Department of Music
Stanford University
Stanford, California USA
gurevich@ccrma.stanford.edu

Bill Verplank
CCRMA, Department of Music
Stanford University
Stanford, California USA
verplank@ccrma.stanford.edu

Pascal Stang
Department of Electrical Engineering
Stanford University
Stanford, California USA
pstang@stanford.edu

## ABSTRACT

Over the past year the instructors of the Human Computer Interaction courses at CCRMA have undertaken a technology shift to a much more powerful teaching platform. We describe the technical features of the new Atmel AVR based platform, contrasting it with the Parallax BASIC Stamp platform used in the past. The successes and failures of the new platform are considered, and some student project success stories described.

## Keywords

Microcontrollers, Music Controllers, Pedagogy, Atmel AVR, BASIC Stamp.

## 1. INTRODUCTION

Every year since 1996, CCRMA has offered a course focusing on human-computer interaction. After the early versions of the course which involved teleconferencing with San Jose State and Princeton, it became Stanford's "course on controllers", offering a hands-on approach to interaction design for musical applications. Until this past year, the core technology was Parallax's BASIC Stamp BS2SX[6] in conjunction with an analog-to-digital converter (ADC) such as Maxim's Max1270[4] and a variety of sensors[11]. In the summer of 2002 CCRMA introduced a new workshop: Physical Interaction Design for Music (PIDM), a two-week summer course using a new technology platform based on the Atmel AVR microcontroller[1]. That platform was subsequently upgraded and further developed for the controllers course in the fall. The switch has proved largely successful, providing many advantages over the previous platform, and resulting greatly improved student work. It is the authors' hope that this paper will facilitate a dialogue among educators to discuss and assess the merits of different technologies being used for teaching in this field. Furthermore, we hope that this paper will provide motivation for others to frankly evaluate current technologies and make appropriate changes where they are needed.

In both the PIDM and HCI courses, a series of simple exercises are done to introduce sensors, signal-conditioning, microcontroller programming, communication, music programming. In addition, they are all done in the context of measuring human performance: latency, repetition-rate, logarithmic thresholds. Finally, a framework is presented for organizing a user-interface design project. All of these are reported in our first NIME paper [11].

## 2. OVERVIEW OF MICROCONTROLLERS

The AVR series of microcontrollers is a recent addition to the field of 8-bit microcontrollers. Table 1 compares several of the most commonly used microprocessor technologies. The Intel 8051- and Motorola HC11-based processors have long histories, numerous derivatives in the case of the 8051, and are widely used in embedded systems. The PIC and AVR series are more recent technologies that have also become firmly established in embedded systems. The AVR line has several key advantages over the PIC line with which it is clearly designed to compete. Most importantly, the AVR core is 4 time faster than the PIC due to its remarkable single clock cycle instruction execution speed. The AVR core was also designed with attention to the requirements of C compilers. As a result, it has strong compiler support from Atmel and the maturation of its open source compiler and tool chain has been much faster and has far surpassed that of PIC. The AVR's ISP (In System Programming) support saves time and reduces damage to chips by minimizing the number of times they are inserted and removed from a circuit.

All of the processors in Table 1 belong to large families of chips with varying features built into each of them. All have variants with on-board A/D conversion, SPI, I2C and UART support. The HC11 is notably supported as a compile target in Metrowerks CodeWarrior, a widely used cross-platform compiler for embedded systems and main-stream operating systems.

The BASIC Stamp is fundamentally different from these processors in that it embeds a microprocessor and its instruction set is entirely emulated. Therefore, it has been omitted from the table, but it will be discussed in detail in the following section.

| | AVR | PIC | Intel 8051 | Motorola HC11 |
|---|---|---|---|---|
| **Architecture** | Harvard | Harvard | Harvard | Von Neumann |
| **Instruction Set** | RISC | RISC | CISC | CISC |
| **Avg. Clock Cycles per Instruction** | 1 | 4 | 24 | 16 |
| **Std. Clock Speeds** | 8,16Mhz | 10,20,40Mhz | 12Mhz | 8Mhz |
| **MIPS** | 8,16 | 2.5,5,10 | 0.5 | 0.5 |
| **Memory Technologies** | flash | flash,OTP | flash,OTP EEPROM,ROMless | flash,OTP EEPROM,ROMless |
| **Typ. Program ROM** | 16KB | 4KB | 4KB | 8KB |
| **Accumulator** | No | Yes | Yes | Yes |
| **gcc support** | Yes | No | No | Yes |

**Table 1: Comparison of commonly used microprocessors.**

# 3. PREVIOUS PLATFORM - PARALLAX BASIC STAMP

The BASIC Stamp[6] is an excellent teaching tool and is certainly the most self-contained microcontroller platform on the market. The typical BASIC Stamp[1] comes in a 20-pin DIP package with 16 general purpose digital I/O pins. The user programs the chip in BASIC on a Windows PC using Parallax's compiler and monitor software. The Stamp's EEPROM program memory is then programmed via an RS-232 serial connection. The serial connection also serves to transmit run-time data and debug messages back to the monitor software on the PC.

The Stamp BASIC interpreter provides a set of special purpose commands in addition to a minimal BASIC language implementation. Notable feature examples include serial input and output on any pin, X10 lighting device control, pulse width modulation output and RC filter time constant measurement.

Two critical weaknesses of Stamp BASIC are the inability of the multiplication and division operators to handle negative numbers, and the lack of floating point support. Fixed point operations which are fundamental to C and assembly programs are also costly to implement in Stamp BASIC.[2]

The processor executes at a slow rate of approximately 4,000 instructions per second. This slow execution time is the most evident result of the overhead involved in providing such a friendly onboard interpreter. For many applications such as traditional buttons and display interfaces, simple robotics, and low-rate data acquisition and reporting this execution rate is not a problem. However, as students refined and attempted to scale their projects or strove for optimized low-latency performance, the speed of the Stamp became a barrier to further progress. Unfortunately, there is no easy way to upgrade, no incremental step up to a more powerful or better suited processor in the Stamp line without significantly revising a student's initial work investment.

[1]Our most recent teaching experience with the BASIC Stamp was with the BS2SX. Parallax has since released two relevant chips, the BS2P and the Javelin stamp. The BS2P is reported to be 20% faster, has an optional 40-pin package doubling the number of available I/O pins, interfaces directly to parallel LCD modules, I2C chips, One Wire chips, and supports interrupt-driven programming. The Javelin stamp is programmed in a subset of the Java programming language and has significantly larger RAM and program EEPROM, both essential to support the higher level language.

[2]These problems have been partially addressed in the BS2SX and BS2P series.

We encountered several problems in using the Stamp as the platform for our courses. Particularly in our academic courses, fewer and fewer students have previous experience with BASIC. Many of them come in with existing knowledge of C and / or a higher level language such as Java. For these students learning BASIC is frustrating because it is very limited, has a more confusing syntax, and violates many of the stylistic rules they have already learned.

The lack of flexible encapsulation mechanisms in BASIC affects our ability to provide and refine program building blocks for the students. BASIC encourages monolithic programs and cut and paste coding whereas the C paradigm is functional and more structured.

The extremely small RAM and ROM memory space available on Stamp processors is also a limiting factor. Even simple applications can exceed the available memory resources of RAM and ROM.

Another significant downside to the Stamp is in the cost per chip. At approximately 60$ US per unit, costs mounted quickly given that chips were getting accidentally burned out at a rate of about one a week. The final serious drawback to the Stamp platform was the lack of a Linux version of the programming software. In our primarily Linux-equipped lab this was a serious bottleneck during lab sessions[3].

# 4. CURRENT PLATFORM - ATMEL AVR

## 4.1 Hardware

Our current hardware platform is based on the Atmel AVR ATmega163 8-bit RISC microcontroller[4]. We use the AVR on a custom-designed small-footprint board[10] and program it from Linux and Windows platforms, with Macintosh OS X programming also possible. The AVR series has a large active user base ranging from professional embedded-systems designers to hobbyists. One of the most important products of that community has been the AVR's inclusion as a standard build target in the open source Gnu *gcc* compiler.

### 4.1.1 *The Processor*

The AVR ATmega163 microcontroller has a maximum clock speed of 8MHz and the majority of its instructions

[3]At the time of writing, Parallax has finally released a Linux and Macintosh-supported development library for creating IDEs on those platforms.

[4]At the time of writing the ATmega163 has been replaced by the pin-compatible ATmega16 which supports clock speeds of up to 16MHz, the low power ATmega16L now has a maximum speed of 8MHz.

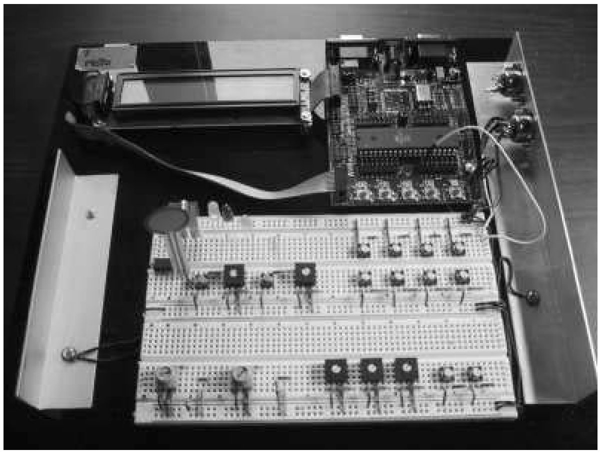


**Figure 1: Prototyping Kit**

complete in a single cycle thus providing assembly code performance near 8 MIPS. Translated into C instructions, we can estimate the execution at a factor of ten slower, still significantly faster than the speed of the Stamp.

The ATmega163 has 16KB of flash program memory, 1KB of data SRAM, and 512 bytes of EEPROM memory[5]. The processor has several interrupt sources so the program may respond to a number of internally and externally-generated events such as timer overflows and the completion of ADC or serial communication operations. The AVRlib[9] function library allows us to package the handling of these interrupts in a straightforward manner which the students quickly learn to use.

Standard microcontroller features found on most of the ATmega series of AVR microcontrollers include an interrupt-controlled UART for serial communication and three independant hardware timers, one of which can be synchronized to an external Real Time Clock (RTC) oscillator. The RTC clock facilitates developing programs that deal in minutes, seconds, and fractions of seconds, useful in many music applications. The processor also supports up to three channels of Pulse Width Modulation (PWM) output. The ATmega processors support the I2C and SPI communication protocols facilitating the addition of external peripheral ICs such as EEPROMs, programmable logic devices (PLDs) and digital to analog converters (DACs).

We chose the AVR series of processors for the cost, memory size, speed, and availability of linux supported development tools. Specifically, we selected the ATmega163 for its eight channels of integrated 10-bit analog to digital conversion (ADC). Having integrated ADCs simplifies the task of reading continuous sensor circuits. The processor provides several choices of analog voltage reference for the ADC including an external reference, simplifying the task of scaling the sensor signals. The conversions complete in as little as $65\mu$s, amply fast for the design of low latency physical interfaces. The ADC support was the most heavily used feature in many student projects.

#### 4.1.2 The Development Board - AVRmini

We use a development board designed by Pascal Stang called the AVRmini[10] to hold the ATmega163 microprocessor. The AVRmini provides convenient access to the I/O ports of the processor via blocks of headers. The headers enable individual pins or entire I/O ports to be connected to a wireless prototyping board using jumper cables. Sensors are also easily connected directly to the pins of the AVRmini using jumpers. The board has resistor packs protecting all of the I/O pins by default which can be selectively disabled by installing bypass jumpers. This simple addition protects the microprocessor from damage caused by incorrect wiring of the I/O pins. We haven't had any processors ruined this way yet, a much better record than we had using Stamp chips.

The AVRmini provides a header to connect an industry-standard character LCD module using four or eight I/O lines. The LCD modules are ubiquitous and inexpensive when purchased from surplus electronics shops. They are a valuable tool for displaying debugging information and for projects requiring state display. The AVRlib[9] library provides several convenient options for output to the LCD. The built-in set of four LEDs and four pushbuttons on the AVRmini may be connected via jumper cables to any of the I/O ports. This functionality has proven useful especially in the early stages of learning about the processor, the board and basic pushbutton and light circuits. The board provides an RTC clock crystal for clocking the third timer which can be enabled via jumpers.

Compiled code is downloaded to the microprocessor from a computer using an RS-232 serial connection. The AVRmini provides two RS-232 connectors. Either one can be patched to the UART so that the programming interface and a serial communications link may be kept connected simultaneously.

The AVRmini supports all 40-pin DIP and 64-pin QFP processors in the AVR series providing flexibility in choosing a processor with features suited to the specific application.

[5]The flash memory stores the compiled program code, the SRAM holds run-time data, and the EEPROM is intended for storing calibration information or other data that must persist over power interruptions.

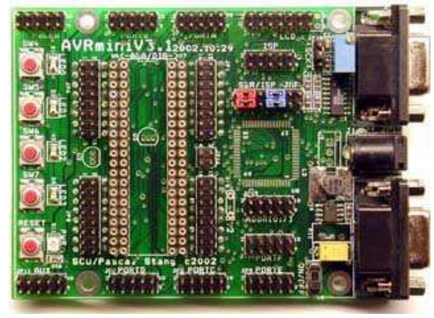


**Figure 2: The AVRmini Development Board (top)**

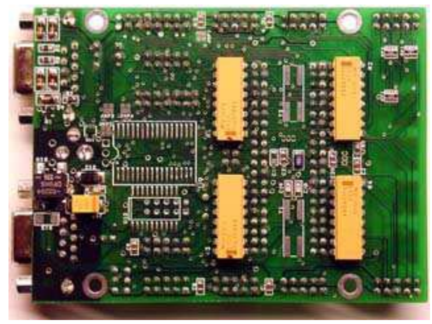


**Figure 3: AVRmini (bottom)**

Provision is also made for a bank of external SRAM of up to 512KB. An efficient switching voltage regulator prolongs battery life for autonomous applications and a separate analog voltage reference regulator may also be installed on the board.

## 4.2 Software

We have used Linux almost exclusively for the programming and performance of the instruments created in these courses because CCRMA's systems are nearly all Linux-based. An IDE for the AVR exists for Windows, and thanks to the open source tool chain built around *gcc*[2], our development environment is programmable from all of the modern operating systems in use today.

### 4.2.1 Compiler and Loader

Programs are written in C with the addition of special purpose macros for the AVR. A makefile automates the process of compiling the AVR C code, linking it with the standard C library and the AVRlib library, generating the memory map and hex code files, and uploading them into the processor. The compiled program is uploaded to the processor using the command line utility *uisp*[7]. Using C enables us to provide the students a library of functions with a consistent interface. As a result, we are able enhance the library and fix bugs without disrupting the students' work flow. While both cut-and-paste code and function libraries are likely to contain code the student does not understand, they can be encouraged to read the code in the library without the danger of their introducing bugs into that part of the code as often happens with monolithic programs.

This is a good example of a case where we can simplify the process for students to make initial operation painless, while keeping everything accessible should they decide to dig deeper. A student can open up the makefile we provide and begin to customize it to suit their needs. By investigating what the makefile and compiler are doing, students can look into the disassembled source code, the memory map, and the hex code files that are generated from the compilation of their C code.

### 4.2.2 Support Library - AVRlib

Pascal Stang's AVRlib[9] library of C support routines is extensive and invaluable for work on this platform. AVRlib includes functions wrapping many of the standard features of the AVR processors. These include support for managing the timers, using the A/D converters, the UART, SPI and I2C interfaces, and the PWM outputs. General purpose code in the library provides bit and byte oriented data buffers, an implementation of a convenient C-style printf function, and terminal emulation facility. There is also support for specific peripherals including character and graphical LCD modules, external SRAM, GPS, ATA hard drives and an MP3 player! This monumental library continues to grow and improve as Pascal uses it in his own teaching and personal projects.

### 4.2.3 Control Output

In an effort to provide the students a flexible set of options for the output of their controllers we supported both MIDI and Open Sound Control (OSC)[5]. For MIDI, the standard MIDI output circuit was attached to the UART pin of the processor and wrapper functions for the UART library routines were provided to give the students functions for standard MIDI message types. OSC support was achieved by compiling the OSC message construction code and a similar set of wrappers around the UART library functions. The OSC messages were received on the Linux computers via RS-232 serial rather than a midi interface. The OSC UDP receiving object for Pd[8] was modified to create an object called OSCSerial that receives messages from the serial port rather than the network[13]. Here, again, students can send MIDI or OSC messages with a single C function call, but have access to the libraries to see exactly what is going on behind the scenes and customize these functions if desired.

In the two courses we have taught using this platform, Pd has been the software of choice for designing the musical side of the projects because of its availability and popularity at CCRMA. Several users have also used the MIDI output functionality to communicate with Max, Pd and commercial music gear.

## 4.3 Motivation for Switching

The aforementioned limitations of the Stamp platform and the lack of Linux software were the initial motivating factors for seeking a new teaching platform. As we considered options for the new platform we realized the benefits of choosing a platform with a stronger more structured programming approach, where C in particular provides a wealth of existing free code that can be ported, modified and appropriated at will. Using *gcc* as the compiler is also an advantage because it provides a coherent transition from programming C in Linux and Mac OS X to programming for the AVR. Equipment cost was another key factor in our switch as a fully outfitted AVRmini board with an ATmega163 processor costs roughly the same as a BS2P Stamp processor.

| | Pros | Cons |
|---|---|---|
| BASIC Stamp | Thorough docs geared to students, self-contained platform, no dev. board needed, simple to program. | Costly, easily damaged, slow execution speed, small memory size, closed platform, Windows dependant. |
| Atmel AVR | Fast execution, low cost, large family, programmable in C, gcc / linux integration, on-board A/D. | Student must confront and understand a complex architecture, documentation sparse, written for engineers. |

**Table 2: Summary of platform pros and cons.**

# 5. OUR EXPERIENCES

A recurring theme in our course has been "Why Microcontrollers?". In other words, why not give students a black box that does 8 channels of A/D conversion on 0-5V signals and generates MIDI messages? The short answer is pedagogical. Using a programmable microcontroller allows the students to learn about computer architecture, digital logic, programming, A/D conversion and serial and parallel communication protocols. In learning to program and use a microcontroller, students develop these skills and intuitions in a practical, hands-on way that would be difficult with theory alone. Furthermore, it gives students exposure to the technology used in actual commercial products, demystifying the world of embedded systems. In this respect, the switch to the AVR has been significant. It represents a shift from essentially a hobbyist's toolkit to professional, commercial-grade technology that is much closer to the technology used in a wide variety of existing commercial devices.

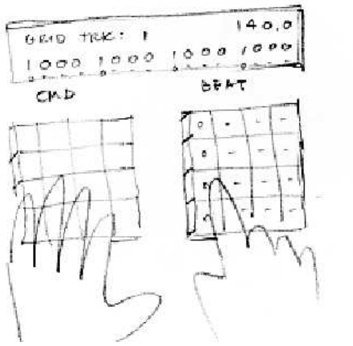



**Figure 4: Beat Matrix**

Admittedly, using a microcontroller, and specifically the AVR, creates more work for the students, though their ability to fully grasp the technology even in the context of our 2-week summer workshop[12] was impressive. The platform's infrastructure, including the AVRlib support libraries, the avr-gcc compiler, and makefiles help make the technology accessible for the novice student, but also importantly gives the advanced student full "under-the-hood" access to a set of well-established, industry standard tools to customize his or her work.

Beyond making students' lives difficult, microcontrollers have enabled students to produce unique, highly successful, innovative projects that would not otherwise be possible. The new platform has greatly improved these facilities, as evidenced in a number of impressive student projects. Example uses of microcontrollers have included parallel LCD interfaces, precise timing, interrupt programming, wireless communication, digital filtering, interpolation, multiplexed I/O and pulse width modulation.

## 5.1 Project Successes

### 5.1.1 *Beat Matrix*

The Beat Matrix, created by David Lowenfels and Gregor Hanuschak, is a MIDI drum sequencer using two 4x4 keypads in conjunction with our development board. The goal was to create a 4 beat sequencer, using one 4x4 grid to represent the sixteenth note subdivisions of the beats. This project's success lies in its self-contained nature, which was afforded by the microcontroller. The controls and display are completely integrated into the device, relying on a computer or synthesizer for sound generation only. Our platform's LCD is integral to the device, giving the user immediate visual feedback for the controls, allowing the user to navigate through drum tracks and displaying the state of the sequencer. David and Gregor learned to manage the microcontroller's internal memory to create sequences, and used interrupts to effectively program the controls, LCD display and MIDI communication to ensure the steady timing essential for a sequencer. The students intend to add features including external memory to store user presets, which should be relatively easy with the current platform and the AVRlib's existing I2C support.

### 5.1.2 *Muggling*

Pascal Stang, Jeff Bernstein and John McCarty developed their "Muggling" project using some of the important capabilities the platform provides. "Muggling" comes from "musical juggling", describing their intention to instrument

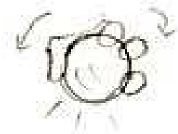


**Figure 5: Muggling**

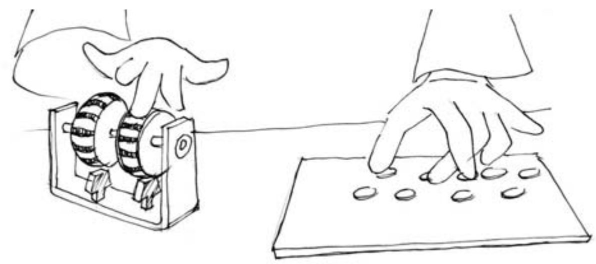


**Figure 6: Rollerblade wheel encoder**

3 juggling balls to send signals that could be used to control musical parameters. The group successfully implemented one ball as a remarkable proof of concept. The ball contains 4 two-axis accelerometers, from which linear and angular acceleration in the x, y and z planes are calculated. Analog acceleration values are sampled, low-pass filtered and interpolated to 14-bits on an AVR323, then transmitted via a Linx wireless radio transmitter to a Linx base station receiver.[3] The portability of the AVRlib code used in class allowed them to easily switch to a more capable microcontroller in the same family without any major changes. The technology's low cost and small package size options allowed the accelerometers, 3-color LED's, microcontroller and radio transmitter/receiver to be mounted inside a 3" diameter ball.

### 5.1.3 *Rollerblade wheel encoder*

The most successful project from our 2-week Physical Interaction Design for Music[12] workshop was a home-made optical encoder built by Chad Cosby. It consisted of a pair of Rollerblade wheels covered in alternating black and white stripes and mounted on a frame to sit on a desk. An inexpensive infrared LED and photodiode pair was attached beside each wheel to act as a light sensor, detecting changes from dark to light as the wheel spun. The LED/photodiode pair is commonly used in robotic devices to allow them to follow dark lines painted on the floor. By timing the transitions, Chad was able to measure the velocity of each wheel, which he encoded as a MIDI control signal and sent to a Linux computer to control sound synthesis parameters in Pd. The ability of the microcontroller to decode digital signals in this manner has also been used with commercial optical shaft encoders, and Duty Cycle Modulated signals from accelerometers.

### 5.1.4 *Rings of Light*

Sara Shaughnessy and Renee Goldschmid mounted the same LED/photodiode packages around the perimeter of two rings to function as proximity sensors when a performer places her hands inside the rings. Each ring contains three sensors, each controlling the intensity and timbre of a note of a chord. A third control ring and a series of force-sensitive pedals adjust the tuning and voicing of the chords. This

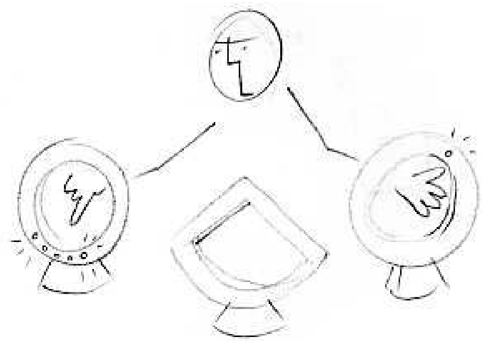


**Figure 7: Rings of Light**

project uses the AVR in combination with programmable logic devices to activate arrays of LEDs that are mounted above each sensor. These LEDs provide immediate feedback to the performer and a compelling display for the audience. Programmable logic and multiplexing were necessary in order to activate 30 LEDs from a limited number of digital output pins. Additionally, the analog inputs were multiplexed to take 12 continuous signals from the proximity sensors and foot pedals into the 8 A/D inputs of the AVR.

## 6. CONCLUSIONS

The switch to our current teaching platform has provided the students with a more fully-featured, robust and flexible system for lab exercises and projects. From a pedagogical standpoint, it has allowed us to deal more directly with microcontroller architecture and embedded systems design, as well as develop a library of functional code that is far more practical than a monolithic, cut-and-paste approach. Advanced students have access to powerful, professional-level tools that promote portability, scaleability, and ample support for refinement and true innovation in their projects. The new platform has the added advantages of easy integration in CCRMA's Linux environment and lower cost.

These improved capabilities have come at the expense of more overhead work on the parts of both the students and instructors. Having developed more support infrastructure and experience in teaching with the technology, it should be possible to use this platform for the PIDM workshop in the future with an improved student experience. Simplified tools for novice users will allow us to better focus on the different pedagogical objectives for a two-week workshop. In the controllers course, the quality of student work has improved dramatically since the introduction of the new platform. The system enabled students to create truly innovative projects employing a wide variety of sensing and communication technologies. The range of successful student projects has provided direct evidence to help us answer the question "Why Microcontrollers?", and has been a gratifying affirmation of our technology shift.

## 7. ACKNOWLEDGMENTS

Thanks to Max Mathews, Wendy Ju, Stefania Serafin, Chris Chafe, Gary Scavone, Fernando Lopez-Lezcano and the CCRMA and music department administrators. Thanks to all of the participants in the 2002 summer workshop who were good sports as our guinea pigs. Thanks to Chad Cosby, David Lowenfels, Gregor Hanuschak, David Lowenfels, Sara Shaughnessy, and Renee Goldschmid.

## 8. REFERENCES

[1] Atmel. `http://www.atmel.com/`, January 2003.

[2] Avr gcc compiler tool chain. `http://www.avrfreaks.net/`, January 2003.

[3] Linx technologies. `http://www.linxtechnologies.com/`, January 2003.

[4] Maxim 1270 12-bit adc. `http://www.maxim-ic.com/`, January 2003.

[5] Open sound control. `http://cnmat.cnmat.berkeley.edu/OSC/`, January 2003.

[6] Parallax inc. `http://www.parallax.com/`, January 2003.

[7] uisp - universal in system programmer. `http://savannah.gnu.org/projects/uisp/`, January 2003.

[8] M. Puckette. Pure data. `http://crca.ucsd.edu/~msp/software.html`, January 2003.

[9] P. Stang. Avrlib: C function library code for atmel avr processors. `http://www.procyonengineering.com/avr/avrlib/`, January 2003.

[10] P. Stang. Avrmini: Homepage of the diminutive atmel avr application/development board. `http://www.procyonengineering.com/avr/avrmini/`, January 2003.

[11] B. Verplank, C. Sapp, and M. Mathews. A course on controllers. In *NIME 2001*, March 2001.

[12] S. Wilson. Ccrma summer physical interaction design workshop website. `http://www-ccrma.stanford.edu/workshops/PIDI2002/`, December 2002.

[13] S. Wilson. Osc serial object. `http://www-ccrma.stanford.edu/~rswilson/OSCSerial`, August 2002.

# Mixxx: Towards Novel DJ Interfaces

Tue Haste Andersen
Department of Computer Science
University of Copenhagen
DK-2100 Copenhagen
Denmark
haste@diku.dk

## ABSTRACT

The Disc Jockey (DJ) software system Mixxx is presented. Mixxx makes it possible to conduct studies of new interaction techniques in connection with the DJ situation, by its open design and easy integration of new software modules and MIDI connection to external controllers. To gain a better understanding of working practices, and to aid the design process of new interfaces, interviews with two contemporary musicians and DJ's are presented. In contact with these musicians development of several novel prototypes for DJ interaction have been made. Finally implementation details of Mixxx are described.

## Keywords

DJ, software, interaction, visualization, controllers, augmented reality.

## 1. INTRODUCTION

The role of a Disc Jockey (DJ) is to select and mix music during playback. The audience can be listening on the radio, on a dance floor, or in an ambient club setting. By playing pieces of music and mixing them together, the DJ seeks to please or provoke the listeners, depending on the setting and audience.

Even today many professional DJ's are still using turntables as playback medium instead of digital based interfaces, such as computers and CD players. This is especially true for the artistic or music-producing DJ's. There are several reasons for this, one of them is fashion and habitude, another is that much contemporary music is still released first (and in many cases only) on vinyl. However, another reason, more important to the work presented here, is the superior interaction possibilities of a DJ turntable when compared to other playback devices. The degree of control and feedback is something that is hard to achieve with todays commercial digital solutions.

In general, DJ systems can be divided into four classes: the analogue systems, digital WIMP (Windows, Icons, Mouse, and Point-and-click) based interfaces, digital interfaces emulating the interaction of analogue interfaces, and finally a new class of digital systems which goes beyond the WIMP interfaces without being limited by the analogue design metaphors. The novel interfaces should not merely be a reproduction of the analogue interfaces, and not limiting the user to only one way of solving a given problem. In this way, the new designs can be used in ways that they have not intentionally been designed for, just as the turntable was not originally designed with scratching in mind.

This paper presents a software system supporting research for this last class of DJ systems. Mixxx [2] is a digital DJ system, which allows for experimentation with interfaces. Several new controllers and visual interfaces have been prototyped and demonstrated using Mixxx, which also will be presented here. To evaluate these new interfaces, Mixxx has to be used in real performance and production situation, and can therefore be regarded as not only a research tool, but also a set of performance tools. The focus of this study will be on the artistic use of DJ interfaces, although it is likely to be of direct importance in areas such as radio and TV production, and other places where sound playback requires a high degree of timing.

The paper is organized as follows: Section 2 describe briefly the setup used by typical DJ's, along with related work. Section 3 presents two interviews with contemporary electronic musicians. A short description of their working practices is given, along with observations of direct importance to the design of new interfaces. Section 4 gives an overview of Mixxx, and the related interface technologies developed, which is followed by a description of the implementation in section 5. Finally a conclusion is given in section 6.

## 2. BACKGROUND

In this section I will briefly describe the interfaces used by DJ's today, followed by a short overview of related research and commercial solutions available.

### 2.1 Live DJ interaction with audio

A traditional DJ setup includes two turntables and a mixer. The control parametes available include continuous variable playback speed, sound level, filtering, and a cross fader for mixing between the two sound sources. Special CD players are also present in most setups. DJ CD players are most notably different from normal CD players in that they have speed control and abilities to store a specific position in a song. Even though the CD players have been available for a long time, and CD's are more convenient to transport when compared to vinyl records, the vinyl still seems to be the preferred playback medium. Digital DJ software solutions are also available, where it is possible to select and mix sounds stored in MP3 format. These solutions often depend on the use of a mouse, and in some cases external knobs connected through MIDI. However, these solutions seem to impose the same problems on the DJ's as the DJ CD players do. When comparing to the turntable, the DJ is loosing visual feedback by the loss of reflection from the grooves in the vinyl, the ability to scratch, and the ability to skip quickly and precisely through a song, by either spinning the record by hand, or by moving the stylus.

The mixer allows for mixing of different audio sources into one, independent control of filters and sound effects, and a special output channel for headphones, which the DJ can use for listening and changing audio before it is mixed with the audio output. This is especially important in disciplines such as *beat mixing*, in which DJ's match two consecutive songs in beat before mixing them together, forming a smooth transition in tempo and pitch from one song to another.

## 2.2 Related work

Research in interfaces of musical instruments is a growing area, with many new ideas on how to interact with novel and old instruments. The work presented here takes a somewhat different approach from other instrument studies, in the sense that it focuses on interaction with pre-recorded audio, instead of generating audio by itself.

Tools for navigating in audio have been explored outside a musical context, mostly in relation to searching in recorded speech. SpeechSkimmer [4] is an example of such a system, where knowledge about pronunciation is used to provide visual cues of where new topics in a conversation are introduced.

A number of commercial digital DJ solutions are available. Most of them provide a GUI for selecting, mixing and controlling playback speed and position, and some have external interfaces with sliders, knobs and rotary controllers, similar to the DJ CD players. One product stands out, namely Final Scratch [20] from Stanton, which produces software and a dedicated DSP box, for connecting the traditional turntables to the computer. Instead of using vinyl as playback medium, special records are used for which the DSP box picks up the position, and sends it to the computer. On the computer, the DJ selects tracks, and in this way is able to use the old turntables, while having the comfort of not carrying hundreds of vinyl discs. The solution is clever, and certainly a step forward when compared to other digital solutions. However, it does not provide any further novel additions to the user interface, apart from a standard scrolling waveform display.

Research on DJ's working practices conducted in an academic setting is rare. Hansen [9] studied specific turntable techniques used by the turntable instrumentalists or the so-called turntablists, musicians who use the turntable as an instrument. These techniques are often employed and invented in the genres of hip-hop, and to a lesser extent techno.

Examples of new music performance tools include Audiopad [18] and Block Jam [17] that makes use of tangible interfaces in the playback and control of music on a sample-based sequencer. These interfaces can be used in a DJ situation to control the arrangement of different tracks and properties of a musical piece as stored in a sequencer, going from a linear playback of a musical piece to a non-linear playback where the DJ or musician is in control. These interfaces solve interaction problems of WIMP interfaces and bring new elements of control in the hands of the DJ. However, music is most often distributed as one linear piece, and thus requires manual segmentation to enable playback and control of individual elements of the musical piece.

# 3. INTERVIEWS

To gain a better understanding of current practices and uses of equipment used by professional electronic musicians and DJ's, two interviews were conducted. The interviews were made as contextual interviews [6], they took place at the DJ's own place and lasted between one and a half to two hours. The interviews were video taped for later analysis and reference. Although only two participants were used, this study made it clear that a number of methods are developed and used individually by each musician, and thus it might be difficult to paint a general picture, even from a large set of interviews.

In the following, the two interviews are described and key issues discovered during the interviews are highlighted. The description is based on observations made during the actual interview, or during analysis of the recorded videos. Both direct verbal information from the interviewed persons, and indirect analysis of the way people act and interact with the instruments are used as basis for the following description.

The first interview was done with an electronic musician who uses only turntables and hardware synthesizers and sequencers in the production and performance of music. This means that no computer with a WIMP interface is involved, only computers with custom interfaces, typically based on a number of buttons, knobs, sliders, dials and a minimum of LCD displays.

A typical composition session by this musician is carried out by playing with the music. Rhythms and bass tracks are programmed on synthesizers, changed and reworked until something which the musician is satisfied with is reached. Then the rhythm tracks are recorded on a hard-disk recorder, and experimentation with the music continues using turntables and keyboards. If something goes wrong, the work is usually completely redone, instead of trying to edit some of the recorded tracks. Preparation for a DJ session includes selecting a set of interesting records, and maybe practice with other musicians if they are involved. Often when several DJ's are playing live together, they set up their equipment on a line, with front to the audience. However, this musician preferred to be able to look at his companion, at least while practicing.

A number of important observations to keep in mind when designing new DJ interfaces were found during the interview:

1. The tempo is constantly adjusted on the turntable to keep the record in synchronization with other sound sources, be it live musicians or a drum machine. By constantly making small adjustments to the playback speed slider, the record is kept in perfect sync with other sources. A slider is superior to +/- pushbuttons with LCD displays, primarily because it can be operated without looking at it, and large changes to the playback speed can be done quickly when searching for the right speed in a new track. The speed slider is also used to adjust phase of beat, instead of stopping the record by holding it with a finger for a very small period of time. This technique is avoided because it results in large transient changes to the pitch when longer cords are played on the record.

2. Visual feedback from the light reflections in the grooves is in general not used. They can be hard to see in a club with limited light sources.

3. When using vinyl records, it is not necessary to know the music on the records in advance. The content can easily be reviewed live, by moving the stylus through the record manually while listening on the headphones.

4. A record is most easily started by scratching over the beat and releasing when in phase with the other sound sources.

5. Composition of songs is primarily done manually. When using drum and bass synthesizers, each part of a song is programmed in a bank. Instead of pre-programming the order of each part, switching between parts is done manually. The reason for this, according to the DJ, is because it is easier, and gives more freedom for improvisations. It was observed that the use of modes [19] on some synthesizers can easily lead to confusion.

6. The filters on the mixer are often used to replace a given frequency band from one track with the same frequency band from another track. Also the filters can be used to make solos, by constantly adjusting them. This is something, according to the DJ, which can be more easily done using WIMP based sequencers when making recordings.

The second musician interviewed works by using WIMP based interfaces extensively, and also plays some acoustic instruments and uses turntables. Different programs were used in this interview including Muzys [16] and ProTools [8]. This musician primarily works by arranging samples of recorded music in a sequencer, both for production and live usage. A number of interesting observations in his usage of the interfaces were also made:

1. Synchronization of a sample with other tracks is done automatically by the program from information about where the beats are in a sample. The beat points is found by the program and adjusted manually. The adjustments are done solely from the visualization of the sampled waveform, not by listening.

2. Time stretching is problematic with respect to sound quality, and therefore pitching is often used, where the pitch is changed along with the playback length.

3. In live sessions, different parts are loaded into the program, and activated by MIDI controls or by clicking with the mouse. New samples cannot be brought in live, since there is no time for adjusting the beat of a sample.

## 3.1 Discussion

The two interviews may not be representative for how electronic musicians and DJ's work in general, but give insight into the process of composition and improvisation using the analogue and digital equipment described previously. The first musician is extremely dependent on how the interfaces are constructed, and uses them to form habits, while the second musician to a much higher extent depends on the features available in WIMP interfaces, at the tradeoff of loosing some control in a live situation. Both depend on the ability to easily search or navigate in the audio, either to synchronize it with other sources, or to get an overview of its contents. It seems that to be able to do the synchronization in a live situation, high precission is required, e.g. as when the record is started by scratching over the beat. The musician having waveform displays available, in some situations relies solely on them, while the first musician only uses haptic and auditory feedback from the instruments.

While WIMP based interfaces are general, they also have many problems, especially in situations where huge requirements is put on the humans shoulder in form of reaction time and precission [19]. When working towards ubiquitous computing, the tangability and feel of human computer interfaces becomes important [10]. Especially in a DJ and other musical situations it seems important, because a high degree of accuracy is required within a narrow time frame. To build new controllers and interfaces, it is therefore of primary importance to understand and possibly model the feel and tangability of the existing analogue interfaces.

For both of the interviewed musicians it also seems of importance to treat computers as individual instruments or sound producing tools. By using each computer as a tool to produce a certain kind of sound, the musicians are able to configure and arrange the tools in their own way, enabling arbitrary physical arrangement and interconnection. The tool based work practice has nothing to do with feel, but may be of importance to creativity imposed by the musicians on the music. By connecting and arranging the tools in different ways, different types of music can be produced. Thus, the musician is not limited by the intended use of the instrument for which it was designed. In this way an instrument designed for one way of producing music can become a broader sound producing tool. The analogue Roland TB303 (Transistor Bass) synthesizer is a good example of an instrument which was designed to be used as a stand-in for a real bass player, but became one of the classic instruments in techno music. The modular analogue synthesizers are an example of a type of instrument which in contrast was designed to be open ended, by letting the musician rewire the analogue circuit producing the sounds.

# 4. DESIGN

The main design goal of Mixxx is to make it possible to conduct interaction studies of novel interfaces in relation to the DJ situation [1]. However, to be able to evaluate novel interfaces and interaction techniques Mixxx must be used in realistic settings of performance and production of music. For this reason Mixxx can be regarded as both a set of performance tools and as a means of studying DJ performances. The studies could for instance be based on quantitative evaluation of controllers, controller mapping, visualization techniques or qualitative studies on the DJ situation in general. By performing such studies it is hoped that new and improved interfaces and ways of interacting with media can be reached.

As such Mixxx is designed to enable for open ended tool based composition as discussed in previous section. The code is modular and enables for graphical and physical interfaces at many levels. As a performance tool Mixxx is currently emulating a traditional DJ setup with mixer, and two playback devices enabling mapping to MIDI controllers and parameter visualizations.

An overview of Mixxx is shown in figure 1. Different interfaces have been proposed and prototyped based on the two interviewed musicians, and a GUI is provided for most of the available controls. The implemented GUI has features resembling other commercial software available, and can be used in comparative studies with other types of interaction. Configuration and track selection is provided solely through the GUI. The prototypes include a controller with the same interface as a DJ mixer, and a rotary controller based on the turntable metaphor. Furthermore a visualization prototype using a Fisheye [7] in waveform displays is shown, along with an augmented turntable currently under development. The augmented turntable brings many of the ideas presented in this article together in one device.

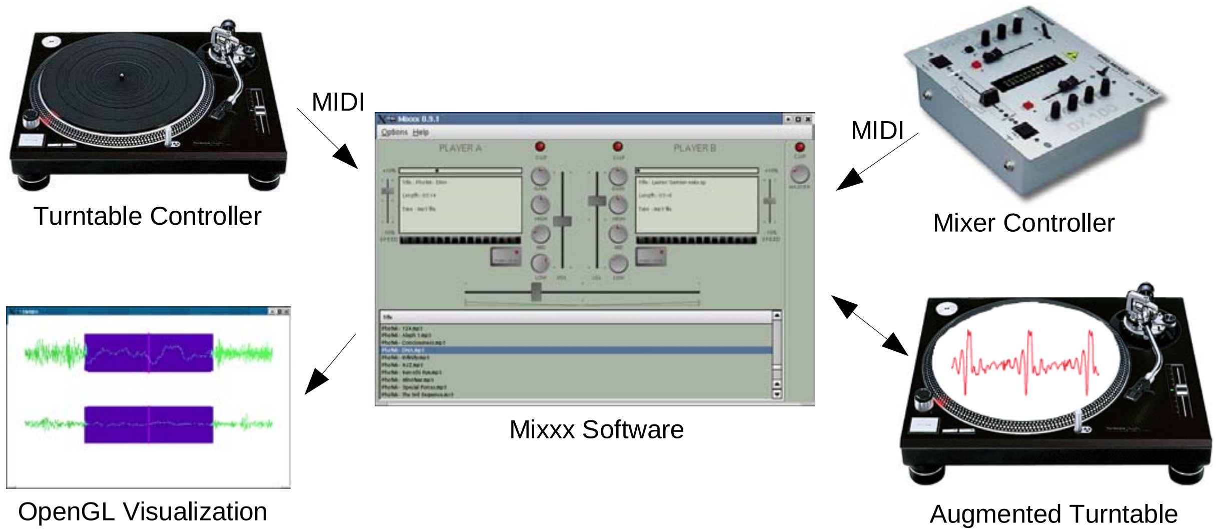



**Figure 1: Overview of Mixxx with the various interaction modules developed so far.**

## 4.1 Mixer interface

As a first step in developing new controllers, a controller box was prototyped with an interface similar to a standard DJ scratch mixer. The prototype is built by replacing the electronics of such a mixer with hardware that converts the controller's analogue output to MIDI messages.

This approach of using external knobs and sliders is identical to commercial MIDI hardware, with the only exception that this interface has the exact look and feel of a scratch mixer.

## 4.2 Turntable metaphor

For searching and synchronization of audio, the analogue turntable seems superior to most digital equipment, although no good visualization techniques are available. Experiments with different rotary controllers have been done, in particular with a modified turntable. The turntable controller works by sensing and outputting the velocity of the deck plate[1] over MIDI. The accuracy of the turntable, when measured as the minimum movement at the edge of the deck plate causing a MIDI velocity message, is less than one millimeter. In this way the turntable can be used to navigate in digital audio, however with another feel that the traditional turntable setup, where a moter is giving active force feedback and the plate is not necessarily following the vinyl when dragging the record back and forth.

As mentioned in section 2 the use of the turntable metaphor is not new. In Final Scratch the metaphor is used, but in a different way. In FinalScratch the turntable stays closer to the analogue DJ turntable, and thus it might be easier for a professional DJ or musician to transfer his or her skills to the FinalScratch setup. However, the purpose of Mixxx and the research conducted with Mixxx, is not to replace the turntable as instrument, but to find new and better ways of navigating and giving new inspiration to the artists. In this sense, staying completely true to the turntable metaphor is likely to limit more than open new design possibilities.

## 4.3 The AudioFish

The AudioFish, earlier presented in [3], is the Fisheye visualization technique [7] applied to waveforms. Different signal values are displayed over time, for each piece of music played back. The idea is to use the fisheye to zoom into a region near the playback position, while still being able to see far into the future and past of the waveform. In the lower left part of figure 1 a screenshot of the AudioFish is shown, where only the waveform is used. This could possibly be overlayed with other time dependent parameters such as information about beat [15], pitch and timbre [11]. On the figure, two tracks are played back, showed one over another. By arranging the tracks as parallel displays they can be compared visually, and thus can be used for synchronizing two or more tracks in time.

The parameter visualization serves a number of purposes:

1. Provides cues of the structure of a song without the need to listen to the song, e.g. by showing energy and tonality as function of time;
2. Allows for matching of parameters from different audio sources using comparative displays;
3. Supports collaborative work through overlay of real-world objects with visualizations of song parameters (Augmented Reality). In this way several musicians can see the parameters more easily than with a traditional computer screen.

The size and zoom of the Fisheye can be changed dynamically, but initial evaluation suggests that this should be done automatically, e.g. as function of the playback speed.

## 4.4 The Augmented Turntable

The Augmented Turntable currently in development, is a combination of the AudioFish with the turntable controller. Using a computer display projected down on the deck plate, the deck plate can be used as a visualization area. By modifying the AudioFish from being plotted in a Cartesian coordinate system, to a circular plot of the waveform in a polar coordinate system [22], the viewing area is used optimally, and the notion of a vinyl groove is reused.

The use of the area on the turntable as visualization supports collaboration to a much higher degree than the use of various external controllers coupled with a WIMP interface. Removing the computer screen means that the musician can

[1]The deck plate is the actual spinning turntable.

now work like a typical DJ artist, and easily have eye contact with other musicians. The use of the circular plot even means that there is no correct angle for the displayed image to be viewed at.

A future possibility is to mount sensors on the deck plate to facilitate a simple point-and-click system, for selecting different parts of a track, or make track selection possible without involving another interface.

The projection is done using a projector above the turntable. This is somewhat primitive but may in the future be replaced by light emitting polymers. Another solution would be to use head mounted see-through displays and the AR ToolKit [12]. This would however require that every person using the turntable would have to wear a display. In all, the projected image currently seems to be a simple and inexpensive solution.

# 5. IMPLEMENTATION

Mixxx is developed in C++ using the QT toolkit [21] and PortAudio [5]. By utilizing these libraries, Mixxx is able to compile and run on MacOS X, Windows, Linux and other Unix derivatives. The program is released under the General Public License, and is freely available for download.

Figure 2 gives an overview of the different modules in Mixxx that are executed in four different thread classes. The internal processing in Mixxx is separated in different objects derived from `EngineObject`, each representing a processing module. Each module takes a buffer as input, and provides a buffer as output. The buffers can be samples, SDIF frames, or other data structures. The modules are similar to other module based processing systems like the Linux audio plugin format LADSPA [13]. Each module is mapped to one or more controller entries, each represented by an object derived from a `ControlObject`. Each entry can be assigned a MIDI value, and thus every GUI control is easily mapped to a MIDI controller. MIDI was chosen as the communication channel to external controllers because of its wide usage. If, however, better time resolution is required, the speed of the serial channel can be changed, at the loss of compliance with MIDI equipment. This may be of interest in research on the required time resolution.

Playback is done in the `Player` object, which request a buffer of samples. To process the samples the `Player` calls a list of `EngineObject`s. The `EngineObject`s signals to the file I/O thread if more samples are needed from the file. Reading and decoding of MP3 files is handled in `SoundSource` objects.

The AudioFish is implemented as a separate module in Mixxx, `MixxxVisual`, running in the main GUI thread. The AudioFish is written using OpenGL to perform the zooming operation directly on the graphics card.

## 5.1 Latency

The latency of the total system is governed by several factors:

- Sound card latency
- Operating system and driver architecture used
- MIDI controller
- Processing speed of one block of audio

The sound card latency can be adjusted dynamically in a preference panel. The system currently uses PortAudio [5] as interface to the platform dependent audio API. On Linux the latency of the total system can thus be below 5 ms, with the right kernel configuration. Lower latencies might be achievable, but has not been researched further. Low latency is also achievable on MacOS X and on Windows.

## 5.2 Controllers

The external MIDI controllers are built using modified hardware, like a turntable or a mixer. For sensing and sending out MIDI events a PIC chip from Microchip [14] was used. The model used is a PIC16F874 with a number of analogue inputs, and digital inputs and outputs. The programs running on the chips are written in C.

In the case of the mixer described in section 4.1, the sliders, potentiometers and buttons are connected to proper inputs on the PIC. The turntable was built using an old belt driven model. The moter was replaced with a high resolution rotary encoder, giving 200 impulses per revolution. Because of the gearing between the deck plate and the rotary, this gives a sub millimeter resolution when moving the deck plate on the outer edge. The PIC chip in the turntable was programmed to output the velocity of the deck plate.

The source code and diagrams for the PIC based controllers will be available on the Mixxx website [2].

# 6. CONCLUSIONS

DJ equipment and work practices have been studied by interviewing two musicians. Although two interviews clearly are not representative for a whole culture, they give insight into the process of DJ'ing and composing music using modern instruments. The use of instruments as tools with open-ended designs, compared to instruments designed for only one purpose is considered important.

Based on these studies the digital DJ system Mixxx has been presented, and demonstrated as an open and extensible base for future research in interaction studies. A number of interface prototypes for Mixxx have been built. Some of these prototypes are based around the turntable metaphor, focusing on tools which can be used by one or more players in a collaborative setting without limiting the interaction to that of analogue equipment.

# 7. ACKNOWLEDGMENTS

Mixxx was developed in collaboration with Ken Haste Andersen. The AudioFish was developed together with Kenny Erleben. Thanks to August Engkilde and DJ Kruzh'Em for the interviews, Kristoffer Jensen and Declan Murphy from the Music Informatics Group at DIKU for inspiring discussions and comments. Finally also a thank to the people at the Doctoral Colloquium, NordiCHI 2002 in Aarhus, for discussions and comments.

# 8. REFERENCES

[1] T. H. Andersen. Live DJ interaction with sound. Doctoral Colloquium at NordiCHI, Aarhus, Denmark, October 2002.

[2] T. H. Andersen and K. H. Andersen. Mixxx. http://mixxx.sourceforge.net/, January 2003.

[3] T. H. Andersen and K. Erleben. Sound interaction by use of comparative visual displays. In *Proceedings of the Second Danish HCI Symposium*. HCØ Tryk, November 2002.

[4] B. Arons. SpeechSkimmer: A system for interactively skimming recorded speech. *ACM Transactions on Computer-Human Interaction*, 4(1):3–38, March 1997.

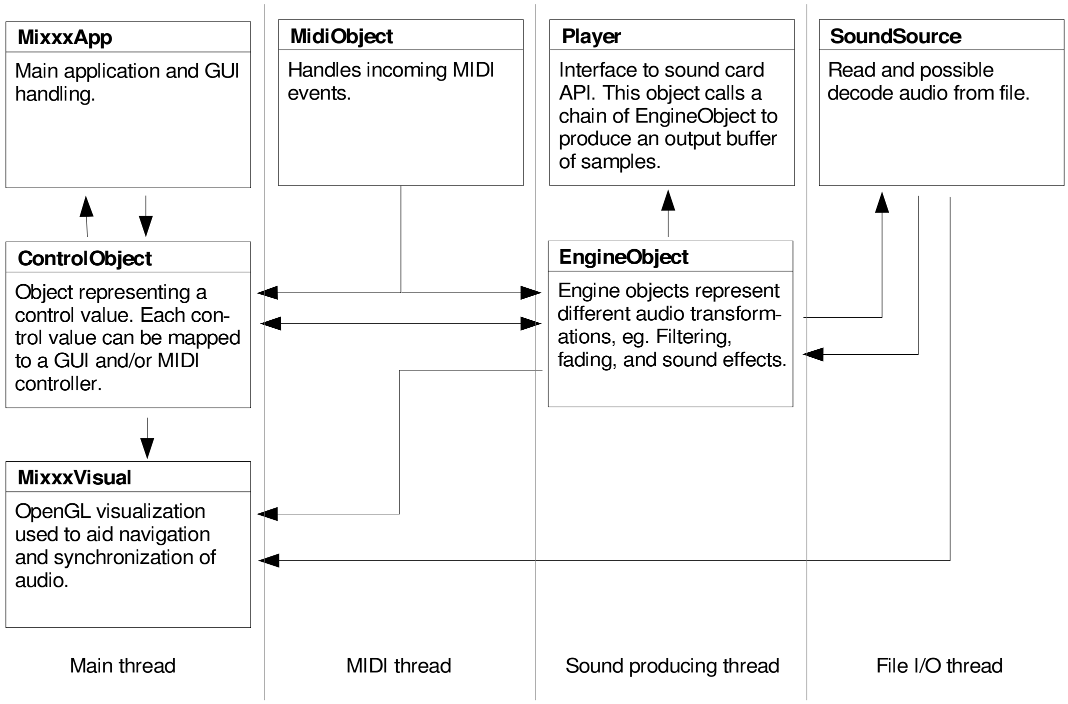



**Figure 2: Overview of the Mixxx architecture, with different modules, executed in four different thread classes.**


[5] R. Bencina and et.al. PortAudio - Portable Audio Library. `http://www.portaudio.com`, January 2003.

[6] H. Beyer and K. Holtzblatt. *Contextual Design*. Morgan Kaufmann Publishers, 1998.

[7] S. Card, J. Mackinlay, and B. Shneiderman, editors. *Readings in Information Visualization*. Morgan Kaufmann Publishers, 1999.

[8] DigiDesign Inc. ProTools. http://www.digidesign.com/, January 2003.

[9] K. F. Hansen. Playing the turntable: An introduction to scratching. *Speech, Music and Hearing*, 42:69–79, 2001. KTH, Stockholm, Sweden, TMH-QPSR.

[10] H. Ishii and B. Ullmer. Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proceedings of CHI*, March 1997.

[11] K. Jensen. *Timbre Models of Musical Sounds*. Ph.D. dissertation, Department of Computer Science, University of Copenhagen, 1999. Report no. 99/7.

[12] H. Kato, M. Billinghurst, and I. Poupyrev. Virtual object manipulation on a table-top AR environment. In *Proceedings of ISAR 2000*, October 2000.

[13] LADSPA developers. Linux Audio Developer's Simple Plugin API (LADSPA). http://www.ladspa.org/, January 2003.

[14] Microchip Inc. Microchip. http://www.microchip.com/, January 2003.

[15] D. Murphy, T. H. Andersen, and K. Jensen. Conducting audio files via computer vision. In *Proceedings of the Gesture Workshop, Genova*, 2003.

[16] Muzys Team. Muzys. http://www.muzys.com/, January 2003.

[17] H. Newton-Dunn, H. Nakano, and J. Gibson. Block Jam. In *Proceedings of the SIGGraph*, 2002. Abstract.

[18] J. Patten, B. Recht, and H. Ishii. Audiopad: A tag-based interface for musical performance. In *Proceedings of the Conference on New Interfaces for Musical Expression*, 2002.

[19] J. Raskin. *The Humane Interface*. Addison-Wesley, 2000.

[20] Stanton. Final Scratch. http://www.finalscratch.com/, January 2003.

[21] Trolltech AS. QT. `http://www.trolltech.com`, January 2003.

[22] F. Vernier, N. Lesh, and C. Shen. Visualization techniques for circular tabletop interfaces. In *ACM Advanced Visual Interfaces (AVI)*, May 2002.

# Score Following: State of the Art and New Developments

Nicola Orio
University of Padova
Dept. of Information Engineering
Via Gradenigo, 6/B
35131 Padova, Italy
orio@dei.unipd.it

Serge Lemouton
Ircam - Centre Pompidou
Production
1, pl. Igor Stravinsky
75004 Paris, France
lemouton@ircam.fr

Diemo Schwarz
Ircam - Centre Pompidou
Applications Temps Réel
1, pl. Igor Stravinsky
75004 Paris, France
schwarz@ircam.fr

## ABSTRACT

Score following is the synchronisation of a computer with a performer playing a known musical score. It now has a history of about twenty years as a research and musical topic, and is an ongoing project at Ircam. We present an overview of existing and historical score following systems, followed by fundamental definitions and terminology, and considerations about score formats, evaluation of score followers, and training. The score follower that we developed at Ircam is based on a Hidden Markov Model and on the modeling of the expected signal received from the performer. The model has been implemented in an audio and a Midi version, and is now being used in production. We report here our first experiences and our first steps towards a complete evaluation of system performances. Finally, we indicate directions how score following can go beyond the artistic applications known today.

## Keywords

Score following, score recognition, real time audio alignment, virtual accompaniment.

## 1. INTRODUCTION

In order to transform the interaction between a computer and a musician into a more interesting experience, the research subject of virtual musicians has been studied for almost 20 years now. The goal is to simulate the behaviour of a musician playing with another, a "synthetic performer", to create a virtual accompanist that will follow the score of the human musician. Score following is often addressed as "real-time automatic accompaniment". This problematic is well defined in [5, 25, 26], where we can find the first use of the term "score following". Since the first formulation of the problem, several solutions have been proposed [1, 2, 4, 6, 7, 8, 9, 10, 14, 16, 17, 18, 20, 22, 23], some academic, others in commercial applications.

Many pieces have been composed relying on score following techniques. For instance, at Ircam we can count at least 15 pieces between 1987 and 1997, such as *Sonus ex Machina* and *En echo* by Philippe Manoury, and *Anthèmes II* and *Explosante-fixe* by Pierre Boulez.

Nevertheless, there are still some limitations in the use of these systems. There are a number of peculiar difficulties inherent in score following, which, after years of research, are well identified. The two most important difficulties are related to possible sources of mismatch between the human and the synthetic performer: On the one hand, musicians can make errors, i.e. playing something differing from the score, because the musical live interpretation of a piece of music means also a certain level of unpredictability. On the other hand, all real-time analysis of musical signals, and in particular pitch detection algorithms, are prone to error.

Existing systems are not general in the sense that it is not possible to track all kinds of musical instruments; moreover, the problem of polyphony is not completely resolved. Although it is possible to follow instruments with low polyphony, such as the violin [15], highly polyphonic instruments or even a group of instruments are still problematic.

Often, only the pitch parameter is taken into account, whereas it is possible to follow other musical parameters (amplitude, gesture, timbre, etc). The user interfaces of these systems are not friendly enough to allow an inexperienced composer to use them. Finally, the follower is not always robust enough; in some particular musical configurations, the score follower fails, which means that it always needs a constant supervision by a human operator during the performance of the piece. The question of reliability is crucial now that all these "interactive pieces" are getting increasingly common in the concert repertoire. The ultimate goal is that a piece that relies on score following can be performed anywhere in the world, based on a printed score for the musicians, and a CD with the algorithms for the automatic performer, for instance in the form of patches and objects for a graphical environment like *j*Max or *Max/MSP*. At the moment, the composer or an assistant who knows the piece and the follower's favourite errors very well must be present to prevent musical catastrophes.

Therefore, robust score following is still an open problem in the computer music field. We propose a new formalisation of this research subject in section 2, allowing simple classification and evaluation of the algorithms currently used.

At Ircam, the research on score following was initiated by Barry Vercoe and Lawrence Beauregard as soon as 1983. It was continued by Miller Puckette and Philippe Manoury [16, 17, 18]. Since 1999, the Real Time Systems Team, now *Real Time Applications* or *Applications Temps-Réel* (ATR)[1], continues work on score following as their priority project. This team has just released a system running in *j*Max based on a statistical model [14, 15], described in section 3. General considerations how score following systems can be evaluated, and results of tests with our system are presented in section 4.

## 2. FUNDAMENTALS

As we try to mimic the behaviour of a musician, we need a better understanding of the special communication involved between musicians when they are playing together, in par-

[1] `http://www.ircam.fr/equipes/temps-reel/`

ticular in concert. This communication requires a highly expert competence, explaining the difficulty of building good synthetic performers.

The question is: How does an accompanist perform his task? When he plays with one or more musicians, synchronizing himself with the others, what are the strategies involved in finding a common tempo, readjusting it constantly? It is not simply a matter of following, but anticipation plays an important role as well. At the state of the art, almost all existing algorithms are only score *followers* strictly speaking. The choice of developing simply "reactive" score followers may be driven by the fact that a reactive system is more easily controllable by musicians, and it reduces the probability of wrong decisions by the synthetic performer.

What are the cues exchanged by the musicians playing together during a performance? They are not only "listening" to each other, but also looking at each other, exchanging very subtle cues: For example, a simple very small movement of the first violin's left hand, or an almost inaudible inspiration of the accompanying pianist are cues strong enough to start a note with perfect synchronisation. There is a real interaction between musicians, a feedback loop, not just unidirectional communication. A conductor is not only simply giving indications to the orchestra, he also pays constant attention to what happens within ("Are they ready?", "Have they received my last cue?") It seems obvious that considering only the Midi sensors of the musician or the audio signal is a severe limitation of the musical experience.

All these considerations regarding the performer behaviour lead towards a multi-modal model, where several cues of different nature (pitch, dynamic, timbre, sensor and also visual information) can be used simultaneously by the computer to find the exact time in the score.

## 2.1 Terminology

We propose a new formalisation, and a systematic terminology of score following in order to be able to classify and compare the various systems proposed up to now.

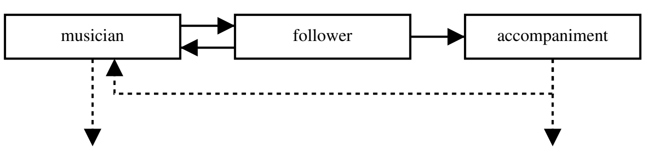



**Figure 1: Elements of a score following system. Dashed arrows represent sound.**

In any score following system, we find at least the elements shown in figure 1: the (human) *musician*, the *follower* (computer), and the *accompaniment* (also called the automatic performance or electronic part). These elements interact with each other. The role of the communication flow from the musician to the computer is clear, because computer behaviour is almost completely based on human performance. On the other hand, the role of auditory feedback from the accompaniment is not negligible; the musician may change the way he plays at least depending on the quality of the score follower synchronisation.

Figure 2 presents the structure of a general score follower. In a pre-processing step, the system extracts some *features* (e.g. pitch, spectrum, amplitude) from the sound produced by the musician. Each score following system defines a different set of relevant features, which are used as descriptors

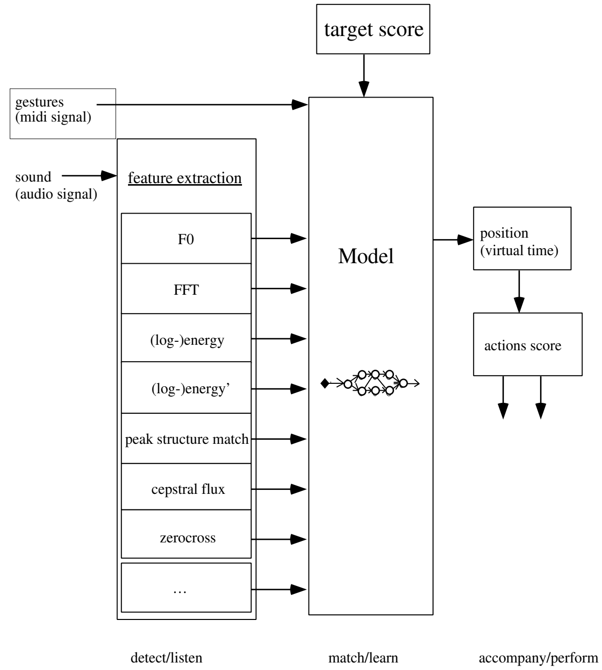



**Figure 2: Structure of a score follower.**

of the musician's performance. These features define the dimension of the *input space* of the model created from the target score. The *target score* is the score that the system has to follow. Ideally this score is identical to the score that the human musician is playing, even though in most of the existing systems, the score is simply coded as a note list. The question of what kind of score format is used for coding the target score is very important for the ergonomics of the system, and for its performance. We present some possible score formats in section 2.2.

The target score is a sequence of *musical events*, that is a sequence of musical gestures that have to be performed by the musician, possibly with a given timing. These gestures can be very simple, i.e. a rest or a single note, or complex, i.e. vibrato, trills, chords, or glissandi. It is important that each gesture is clearly defined in the common music practice, and its acoustic effect is known.

The *model* is the system's internal representation of this coding of the target score. The model is matched with the incoming data in the follower, while the *actions score* represents the actions that the accompaniment has to perform at some specified positions (e.g. sound synthesis or transformations).

The *position* is the current time of the system relative to the target score. The target score contains also *labels* that can be any symbol, but are usually integer values, giving the cue number of the synthesis or transformation event in the actions score that should be triggered at reception of that label. (That's why often the labels are also called "cues".) The labels can be attached to any of the musical events, for instance to the ones that are particularly relevant in the score, but in general each event can have a label, and also the rests in the score.

According to Vercoe [26], the score follower has to fulfill three tasks: Listen-Perform-Learn. Listening and performing are mandatory tasks for an automatic performer, while learning is a more subtle feature. It can be defined as the ability of taking advantage from previous experiences that, in the case of an accompanist, may regard both previous rehearsals with the same musicians and the knowledge gained in years of public performance. It can be noted that sometimes these two sources of experience may reflect different accompanist's choices during a performance, that are hard to model. Learning can affect different levels of the process: the way the score is modeled, the way features are extracted and used for synchronisation, and the way the performance is modeled and recognized.

There are a number of advantages in using a statistical system for score following, which regard the possibility of training the system and modeling different acoustic features from examples of performances and score. In particular, a statistical approach to score following can take advantage from theory and applications of Hidden Markov Models (HMMs) [19]. A number of score followers have been developed using HMMs, such as the one developed at Ircam [14] and others [12, 21]. In fact, HMMs can deal with the several levels of unpredictability typical of performed music and they can model complex features, without requiring preprocessing techniques that are prone to errors like any pitch detectors or midi sensors. For instance, in our approach, the whole frequency spectrum of the signal is modeled. Finally, techniques have been developed for the training of HMMs.

## 2.2 Target Score Format

The definition of the imported target score format is essential for the ease of use and acceptance of score following. The constraints are multiple:

- It has to be powerful, flexible, and extensible enough to represent all the things we want to follow.
- There should be an existing parser for importing it, preferably as an open source library.
- Export from popular score editors (Finale, Sibelius) should be easily possible.
- It should be possible to fine-tune imported scores within the score following system, without re-importing them.

The formats that we considered are:

- Graphical score editor formats: *Finale*, *Sibelius*, *NIFF*, *Guido*
- Mark-up languages: *MusicML*, *MuTaTedTwo*, *Wedelmusic XML Format*
- Frameworks: *Common Practice Music Notation* (CPNview), *Allegro*
- Midi

Midi, despite its limitations, is for the moment indeed the only representation to fulfill all these constraints: It can code everything we want to follow, e.g. using conventions for special Midi channels, controllers, or text events. It can be exported from every score editor, and can be fine-tuned in the sequence editor of our score following system. Hence, we stay with Midi for the time being, but the search for a higher-level format that inserts itself well into the composer's and musical assistant's workflow continues.

## 2.3 Training

One fundamental difference between a computer and a human being is that the latter is learning from experience, whereas a computer program usually does not improve its performance by itself. Since [26], we imagine that a virtual musician should, like a living musician, learn his part and improve his playing during the rehearsals with the other musician. One of the advantages of a score following system based on a statistical model is that it can learn using well-known training methods.

The training can be supervised or unsupervised. Training is unsupervised if it does not need the use of target data, but only several interpretations of the music to be followed.

In order to design a score following system that learns, we can imagine several scenarios:

- When the user inputs the target score, he is teaching the score to the computer.
- During rehearsals, the user can teach the system by a kind of gratification if the system worked properly for a section of the score.
- After each successful performance, so that the system gets increasingly familiar with the musical piece in question.

In the context of our HMM score follower, training means adapting the various probabilities and probability distributions governing the HMM to one or more example performances such as to optimise the quality of the follower. At least two different things can be trained: the transition probabilities between the states of the Markov chain [14], and the probability density functions (PDFs) of the observation likelihoods. While the former is applicable for audio and Midi, but needs much example data, especially with errors, the latter can be done for audio by a statistical analysis of the features to derive the PDFs, which essentially perform a mapping from a feature to a probability of attack or sustain or rest.

Then of course a real iterative training (supervised by providing a reference alignment, or unsupervised starting from the already good alignment to date) of the transition and observation probabilities is being worked on to increase the robustness of the follower even more. This training can adapt to the "style" of a certain singer or musician.

# 3. IMPLEMENTATION

Ircam's score follower consists of the objects *suiviaudio* and *suivimidi* and several helper objects, bundled in the package *suivi* for *j*Max. The system is based on a two-level Hidden Markov Model, as described in [14]:

States at the *higher level* are used to model the music events written in the score, which may be simple notes (or rests) but also more complex events like chords, trills, and notes with vibrato. The idea is that the first thing to model is the score itself, because it can be considered as the hidden process that underlies the musical performance. By taking into account complex events, e.g. considering a trill as an event by itself rather than a sequence of simple notes, it is possible to generalize the model also to other musical gestures, like for instance glissandi or arpeggios which are not currently implemented.

Together with the sequence of events in the score, which have temporal relationships that are reflected in the left-to-right structure of the HMM, also possible performing errors are modeled. As introduced by [5], there are three possible errors: wrong notes, skipped notes, or inserted notes. The model copes with these errors by introducing error states, or *ghost states*, that model the possibility of playing a wrong event after each event in the score. Ghost states can be used not only to improve the overall performances of the system in terms of score following, but also as a way to refine the automatic performance adding new strategies. For instance, if the system finds that the musician is playing wrong events then it can suspend the automatic performance in order to minimize the effect to the audience, or it can suggest the correct actual expected position in the score depending on composer's choices.

States at the *lower level* are used to model the input features. These states are specialized for modeling different parts of the performance, like the attack, the sustain, and the possible rest, and they are compound together to create states at the higher level. For instance, in an attack state, the follower expects a rise in energy for audio or the start of a note for Midi.

The object *suiviaudio* uses the features *log-energy* and *delta log-energy* to distinguish rests from notes and detect attacks, and the energy in harmonic bands according to the note pitch, and its delta, as described in [15], to match the played notes to the expected notes. The energy in harmonic bands is also called PSM for *peak structure match*. For the singing voice, the *cepstral difference* feature improves the recognition of repeated notes, by detecting the change of the spectral envelope shape when the phonemes change. It is the sum of the square differences of the first 12 cepstral coefficients from one analysis frame to another.

The object *suivimidi* uses a simpler information, that is the onset and the offset of Midi notes. The Midi score follower works even for highly polyphonic scores by defining a note match according to a comparison of the played with the expected notes for each HMM state.

Score following is obtained by on-line alignment of the audio or Midi features to the states in the HMM. A technique alternative to classical Viterbi decoding is employed, as described in [14].

The code that actually builds and calculates the Hidden Markov Model is common to both audio and Midi followers. Only the handling of the input and the calculation of the observation likelihoods for the lower level states are specific to one type of follower.

The system uses the *j*Max *sequence editor* for importing Midi score files, and visualisation of the score and the recognition (followed notes and the position on the time axis are highlighted as they are recognised).

# 4. EVALUATION

Eventually, to evaluate a score following system, we could apply a kind of Turing test to the synthetic performer, which means that an external observer has to tell if the accompanist is a human or a computer. In the meantime, we can distinguish between subjective vs. objective evaluation:

## 4.1 Subjective Evaluation

A *subjective* or *qualitative* evaluation of a score follower means that the important performance events are recognised with a latency that respects the intention of the composer, which is therefore dependent on the action that is triggered by this event. Independent of the piece, it can be done by assuming the hardest case, i.e. all notes have to be recognised immediately. The method is to listen to a click that is output at each recognised event and observe the visual feedback of the score follower (the currently recognised note in the sequence editor and its position on the time axis are highlighted), verifying that it is correct. This automatically includes the human perceptual thresholds for detection of synchronous events in the evaluation process.

A limited form of subjective evaluation is definitely needed in the concert situation to give immediate feedback whether the follower follows, and before the concert to catch setup errors.

## 4.2 Objective Evaluation

An *objective* or *quantitative* evaluation, i.e. to know down to the millisecond when each performance event was recognised, even if overkill for the actual use of score following, is helpful for refinement of the technique and comparison of score following algorithms, quantitative proof of improvements, automatic testing in batch, making statistics on large corpora of test data, and so on.

Objective evaluation needs reference data that provides the correct alignment of the score with the performance. In our case this means a reference track with the labeled events at the points in time where their label should be output by the follower. For a performance given in a Midi-file, the reference is the performance itself. For a performance from an audio file, the reference is the score aligned to the audio. Midified instruments are a good way to obtain the performance/reference pairs because of the perfect synchronicity of the data.

The reference labels are then compared to the cues output by the score follower. The offset is defined as the time lapse between the output of corresponding cues. Cues with their absolute offsets greater than a certain threshold (e.g. 100 ms), or cues that have not been output by the follower, are considered an error. The values characterising the quality of a score follower are then:

- the percentage of non-error labels
- the average offset for non-error labels, which, if different from zero, indicates a systematic latency
- the standard deviation of the offset for non-error labels, which shows the imprecision or spread of the follower
- the average absolute offset of non-error labels, which shows the global precision

There are other aspects of the quality of a score follower not expressed by these values: According to classical measures of automatic systems that simulate the human behavior [3], error labels can be due to the *miss* of a correct label at a given moment, or to the *false alarm* of a label incorrectly given. Based on these two measures it is possible to consider also the number of labels detected more than once, or the zigzagging back to an already detected label.

Again, the tolerable number of mistakes and latencies of the follower largely depend on the kind of application and the type of musical style involved. It can be noted that, for this kind of evaluation, it is assumed that the musician does not make any errors. It is likely that, in a real situation, human errors will occur, suggesting as another measure the time needed by the score follower to recover from an error situation, that is, to resynchronise itself after a number of wrong notes are played. The tolerable number of wrong notes played by the musician is another parameter by itself, that in our system can be experimentally measured through simulations of wrong performances. This aspect is part of the training that can be done directly when creating the model of the score as an injection of a priori knowledge on the HMMs.

### 4.3 Evaluation Framework

To perform evaluation in our system, we developed the object *suivieval*, which takes as input the events and labels output by the score follower, the note and outputs of the reference performance, and the same control messages as the score follower (to synchronize with its parameters). While running, it outputs abovementioned values from a running statistics to get a quick glance at the development and quality of the tested follower. On reception of the stop message, the final values are output, and detailed event and match protocols are written to external files for later analysis.

We chose to implement the evaluation outside of the score following objects, instead of inserting measurement code to them. This *black box testing* approach has the advantages that it is then possible to test other followers or previous versions of our score following algorithm to quantify improvements, to run two followers in parallel, and that evaluation can be done for Midi and audio, without changing the code of the followers.

However, with the opposite *glass box testing* approach of adding evaluation code to the follower, it is possible to inspect its internal state (which is not comparable with other score following algorithms!) to optimise the algorithm.

### 4.4 Tests

We have collected a database of files for testing score followers. This database is composed of audio recordings of several different interpretations of the same musical pieces, by one or several musicians, and the corresponding aligned score in Midi format. The database principally includes musical works produced at Ircam using score following (Pierre Boulez *Anthèmes II*, Philippe Manoury *Jupiter*, ...) but also several interpretations of more classical music (Moussorgsky, Bach).

The existing systems that are candidates for an objective comparative evaluation are: *Explode* [16], *f9* [17], *Music Plus One*[2] [23], *ComParser*[24], and the systems described in [2, 9]. This evaluation is still to be done.

#### *4.4.1 Audio Following*

On our follower, we carried out informal subjective tests with professional musicians on the performance of the implemented score follower together with a *j*Max implementation of *f9*, a score follower that is based on the technique reported in [17], and a *j*Max implementation of the Midi follower *Explode* [16], which received the input from a midified flute. Tests were carried out using pieces of contemporary music that have been composed for a soloist and automatic accompaniment.

In *Pluton* for flute, the audio follower *f9* made unrecoverable errors already in the pitch detection, which deteriorated the performances of the score follower. With *Explosante-fixe* the midified flute's output was hardly usable, and lead to early triggers from *Explode*. Our audio follower *suivimidi* follows the flute perfectly.

Other tests have been conducted with other instruments, using short excerpts from *Anthèmes II* for solo violin, with a perfect following both of trills and chords. The different kind of events, that are not directly modeled by *f9* or *Explode*, required ad hoc strategies for preventing the other followers to loose the correct position in the score.

An important set of tests have been carried out on the piece *En Echo* by Philippe Manoury, for a singer and live-electronics. Different recordings of the piece have been used, they were performed by different singers and some of them included also background noise and recording of the live-electronics in order to reproduce a concert situation. The performances of *f9*, which is currently used in productions, are well known: there are a number of points in the piece where the follower gets lost and it is necessary to manually resynchronize the system. On the other hand, *suiviaudio* succeeded to follow the complete score, even if there was some local mismatch for the duration of one note.

Tests on *En Echo* highlighted some of the problems related to the voice following. In particular, the fact that there are sometimes two consecutive legato notes in the score with the same pitch for two syllables, needed to be directly addressed. To this end we added a new feature in our model, the *cepstral flux*, as shown in figure 2. Moreover, new events typical of the singing voice needed to be modeled, as fricatives and unvoiced consonants.

#### *4.4.2 Midi Following*

Monophonic tests have been developed for the Midi follower *suivimidi*. The testing of Midi followers is easier because it is possible to change the performance at will, without the need of a performer. In case of a correct performance, *suivimidi* was always perfectly following, and it has been shown to be robust to errors affecting up to 5 subsequent notes, even more in some cases.

Real life tests with the highly polyphonic *Pluton* for midified piano showed one fundamental point for score following: Ideally, the score should be a high-level representation of the piece to be played. Here, for practical reasons, we used a previous performance as the score, with the result that the follower got stuck. Closer examination showed that this was because of the extensive but inconsistent use of the sustain pedal, which was left to the discretion of the pianist, resulting in completely different note lengths (of more than 50 seconds) and polyphony. Once the note lengths were roughly equalised, the follower had no problems, even in parts with a trill that was (out of laziness) not yet represented as a single trill score event. This test shows us a shortcoming of the handling of highly polyphonic scores, which will be resolved by the introduction of a decaying weight of each note in the note match probability.

## 5. CONCLUSION AND FUTURE WORK

We have a working score following system for *j*Max version 4 on Linux and Mac OS-X, the fruit of three years of research and development, that is beginning to be used in production. It is released for the general public in the Ircam Forum[3]. Porting to *Max/MSP* is planned for next autumn.

Two other running artistic and research projects at Ircam extend application of score following techniques:

One is a theatre piece, for which our follower will be extended to follow the spoken voice, similar to [11, 13]. This addition of phoneme recognition will also bring improvements to the following of the singing voice.

The other is the extension of score following to multimodal inputs from various sensors, leading towards a more

[2] `http://fafner.math.umass.edu/`

[3] `http://www.ircam.fr/forumnet/`

modular structure where the Markov model part is independent from the input analysis part, such that you can combine various features derived from audio input with Midi input from sensors and even video image analysis.

## 6. ACKNOWLEDGMENTS

We would like to thank Philippe Manoury, Andrew Gerzso, François Déchelle, and Riccardo Borghesi without whose valuable contributions the project could not have advanced that far.

## 7. ADDITIONAL AUTHORS

Norbert Schnell, Ircam - Centre Pompidou, Applications Temps Réel, email: schnell@ircam.fr

## 8. REFERENCES

[1] B. Baird, D. Blevins, and N. Zahler. The Artificially Intelligent Computer Performer: The Second Generation. In *Interface – Journal of New Music Research*, number 19, pages 197–204, 1990.

[2] B. Baird, D. Blevins, and N. Zahler. Artificial Intelligence and Music: Implementing an Interactive Computer Performer. *Computer Music Journal*, 17(2):73–79, 1993.

[3] D. Beeferman, A. Berger, and J. D. Lafferty. Statistical models for text segmentation. *Machine Learning*, 34(1-3):177–210, 1999.

[4] J. Bryson. The Reactive Accompanist: Adaptation and Behavior Decomposition in a Music System. In L. Steels, editor, *The Biology and Technology of Intelligent Autonomous Agents*. Springer-Verlag: Heidelberg, Germany, 1995.

[5] R. B. Dannenberg. An On-Line Algorithm for Real-Time Accompaniment. In *Proceedings of the ICMC*, pages 193–198, 1984.

[6] R. B. Dannenberg and B. Mont-Reynaud. Following an Improvisation in Real Time. In *Proceedings of the ICMC*, pages 241–248, 1987.

[7] R. B. Dannenberg and Mukaino. New Techniques for Enhanced Quality of Computer Accompaniment. In *Proceedings of the ICMC*, pages 243–249, 1988.

[8] L. Grubb and R. B. Dannenberg. Automating Ensemble Performance. In *Proceedings of the ICMC*, pages 63–69, 1994.

[9] L. Grubb and R. B. Dannenberg. A Stochastic Method of Tracking a Vocal Performer. In *Proceedings of the ICMC*, pages 301–308, 1997.

[10] L. Grubb and R. B. Dannenberg. Enhanced Vocal Performance Tracking Using Multiple Information Sources. In *Proceedings of the ICMC*, pages 37–44, 1998.

[11] A. Loscos, P. Cano, and J. Bonada. Low-Delay Singing Voice Alignment to Text. In *Proceedings of the ICMC*, 1999.

[12] A. Loscos, P. Cano, and J. Bonada. Score-Performance Matching using HMMs. In *Proceedings of the ICMC*, pages 441–444, 1999.

[13] A. Loscos, P. Cano, J. Bonada, M. de Boer, and X. Serra. Voice Morphing System for Impersonating in Karaoke Applications. In *Proceedings of the ICMC*, 1999.

[14] N. Orio and F. Déchelle. Score Following Using Spectral Analysis and Hidden Markov Models. In *Proceedings of the ICMC*, Havana, Cuba, 2001.

[15] N. Orio and D. Schwarz. Alignment of Monophonic and Polypophonic Music to a Score. In *Proceedings of the ICMC*, Havana, Cuba, 2001.

[16] M. Puckette. EXPLODE: A User Interface for Sequencing and Score Following. In *Proceedings of the ICMC*, pages 259–261, 1990.

[17] M. Puckette. Score Following Using the Sung Voice. In *Proceedings of the ICMC*, pages 199–200, 1995.

[18] M. Puckette and C. Lippe. Score Following in Practice. In *Proceedings of the ICMC*, pages 182–185, 1992.

[19] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, 1989.

[20] C. Raphael. A Probabilistic Expert System for Automatic Musical Accompaniment. *Jour. of Comp. and Graph. Stats*, 10(3):487–512, 1999.

[21] C. Raphael. Automatic Segmentation of Acoustic Musical Signals Using Hidden Markov Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):360–370, 1999.

[22] C. Raphael. A Bayesian Network for Real Time Music Accompaniment. *Neural Information Processing Systems (NIPS)*, (14), 2001.

[23] C. Raphael. Music Plus One: A System for Expressive and Flexible Musical Accompaniment. In *Proceedings of the ICMC*, Havana, Cuba, 2001.

[24] Schreck Ensemble and P. Suurmond. ComParser. Web page, 2001. http://www.hku.nl/~pieter/SOFT/CMP/.

[25] B. Vercoe. The Synthetic Performer in the Context of Live Performance. In *Proceedings of the ICMC*, pages 199–200, 1984.

[26] B. Vercoe and M. Puckette. Synthetic Rehearsal: Training the Synthetic Performer. In *Proceedings of the ICMC*, pages 275–278, 1985.

# Indirect Acquisition of Instrumental Gesture Based on Signal, Physical and Perceptual Information

Caroline Traube
Laboratoire d'acoustique musicale
Faculté de musique, Univ. de Montréal
200, avenue Vincent-d'Indy
Montréal (Qc), Canada H3C 3J7
caroline.traube@umontreal.ca

Philippe Depalle
Music Technology Area
Faculty of Music, McGill University
555 Sherbrooke Street West
Montreal (Qc), Canada H3A 1E3
depalle@music.mcgill.ca

Marcelo Wanderley
Music Technology Area
Faculty of Music, McGill University
555 Sherbrooke Street West
Montreal (Qc), Canada H3A 1E3
mwanderley@acm.org

## ABSTRACT
In this paper, we describe a multi-level approach for the extraction of instrumental gesture parameters taken from the characteristics of the signal captured by a microphone and based on the knowledge of physical mechanisms taking place on the instrument. We also explore the relationships between some features of timbre and gesture parameters, taking as a starting point for the exploration the timbre descriptors commonly used by professional musicians when they verbally describe the sounds they produce with their instrument. Finally, we present how this multi-level approach can be applied to the study of the timbre space of the classical guitar.

## Keywords
Signal analysis, indirect acquisition of instrumental gesture, guitar

## 1. INSTRUMENTAL GESTURE
When musicians play on a traditional musical instrument, they usually interact with a control surface made of keys (piano, clarinet), strings (violin), mouthpieces (trumpet), reeds (oboe), etc. In most cases, many years of motor skills development are necessary to control the instrument adequately in order to intentionally produce sounds of a given quality or timbre.

In this paper, we will call *instrumental gesture* the actual instrument manipulation and playing technique on an instrument [2]. We will consider here the *effective gesture*, defined as the purely functional level of the notion of gesture, i.e., the gesture necessary to mechanically produce the sound (like blowing in a flute, bowing on a string, pressing a key of a piano and so on). We will call *instrumental gesture parameters* the parameters characterizing the components of the instrumental gesture. They are, for example, the speed of an air jet, the location of a pluck along a string, and the pressure applied with a bow on a string. The variations of these parameters have an effect on the timbre and are usually clearly perceived by a trained listener such as a professional musician.

Considering the case of the guitar and referring to the typology established in [2] and [3], plucking is an *excitation* and *modification* gesture, while fingering is a *selection* and also a *modification* gesture since the choice of fingering on the neck of the guitar (string/fret combination) affects timbre as well.

## 2. RELATIONSHIP BETWEEN GESTURE AND TIMBRE
Musical expression has been traditionally related to expressive timing and dynamic deviations in performance [8]. Less attention has been given to the study of timbre and how it relates to musical expression. This is probably due to the difficulty of defining the features of timbre, which are related to the physical aspects of sound in very complex ways. On the other hand, pitch, duration and volume are perceptual phenomena that have fairly simple physical correlates. Here, we propose to limit the scope of the study to the aspects of timbre that musicians can clearly control, perceive and verbally describe.

### 2.1 Perceptual dimensions of an instrumental timbre space
Early studies on instrumental timbre were performed by David Wessel and John Grey in the late 1970's [9]. Based on similarity judgments, those studies used multidimensional scaling algorithms to reduce the number of dimensions in the timbre space. Timbral features such as brightness (associated with the spectral center of gravity), spectral irregularity (spectral flux) and transients density were identified. It is important to note that these axes were used to differentiate between different orchestral instruments—a macroscopic view of timbre—as opposed to differentiating between the possible palette of timbres in a single instrument—a microscopic view of timbre. This is precisely the viewpoint of our approach. In particular, we want to identify the dimensions of the timbre subspace corresponding to the classical guitar, the instrument chosen for investigation and validation in this study.

### 2.2 Source-mode of timbre perception
Handel proposed in 1995 an explanation for timbre perception, saying that the subjective identification of timbre could involve the observer's perception of the physical mechanisms and actions in the sound production. This is the *source-mode* of timbre perception, as opposed to the *interpretative mode* of timbre perception [10]. Other facts support this view such as the source-filter model of speech perception. It is also interesting to realize that mechanics and materials of vibrating systems are the bases for traditional Western musical instrument as well as World instrument classification systems (e.g. von Hornbostel & Sachs classification in aerophones, chordophones and membraphones).

Considering the evident relationships between physical model constituents, instrumental gesture and perceptual attributes, we believe that the gestural information can be accessed via the identification of the parameters of a physical model. As it has been done for speech vowels, we propose to define an articulatory timbre space for individual musical instruments and to determine the relationships between this articulatory space and a perceptual timbre space (as defined by Grey in [9]).

# 3. INDIRECT ACQUISITION OF INSTRUMENTAL GESTURE

## 3.1 Direct vs indirect acquisition

There are different ways to capture the characteristics of instrumental gesture [5]:

- through *direct acquisition* of physical variables with sensors on the instrument or on the performer,
- through *indirect acquisition* of performance parameters from the analysis of the acoustic signal (namely from a recording).

In recent years we have seen an important development of technologies related to sensors and gestural interfaces. For example, many musical instruments can be augmented with devices that can monitor the performer's actions (choice of keys, pressure applied to a mouthpiece, etc.) and turn it into MIDI information. Direct acquisition is clearly a simpler way to capture the physical features of a gesture but it is potentially invasive and may ignore the interdependency of the various variables. For example, sensors on a clarinet could detect the air jet speed and the fingering but would not account for the coupling between the excitation and the resonator. As opposed to direct acquisition, indirect acquisition is based on the assumption that the performance parameters can be extracted from the signal analysis of the sound being produced by an instrument. The main difficulty of this task is to determine in the signal the specific acoustic signature of a particular performance parameter that has a perceivable influence on the sound.

## 3.2 From acoustic signal to instrumental gesture information

Most traditional musical instruments are stable during a performance, i.e., the acoustical properties of the instrument do not change over the time of the performance and an energy continuum needs to exist between the gesture and the perceived sound [3]. It is also interesting to note that in the case of most traditional acoustic instruments, the gestural interface is also part of the sound production unit. For instance, the reed, keys and holes of a clarinet are the elements the musician interacts with, but they are also responsible for the sound production [15].

Figure 1 schematizes the exchange of information between the three elements of a performance process: the performer, the instrument and the listener. Note that a musician is at the same time a performer and a listener.

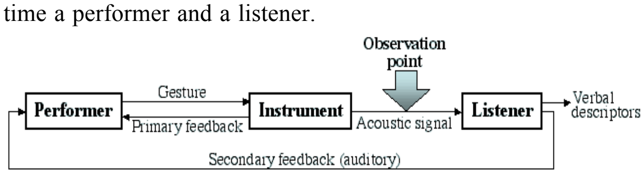



**Figure 1. Interactions between the performer, the instrument and the listener.**

The performer applies a gesture on the instrument, which in turn reacts to the gesture by producing a sound and by providing the performer with primary feedback, which can be visual, auditory (clarinet key noise, for instance) and tactile-kinesthetic [15]. The listener perceives sounds and attaches labels to them. Expert performers/listeners are generally able to discriminate and intuitively describe a large variety of sounds produced by their instruments.

In the approach that we propose, the observation point in the performance process loop is the acoustic signal, from which we extract structural information that allows us to get to the gestural information. To generate the data, musicians are recorded playing tones with specific gestures, varying one gesture parameter at a time. Figure 2 illustrates the procedure that we propose to access instrumental gesture information from the acoustic signal.

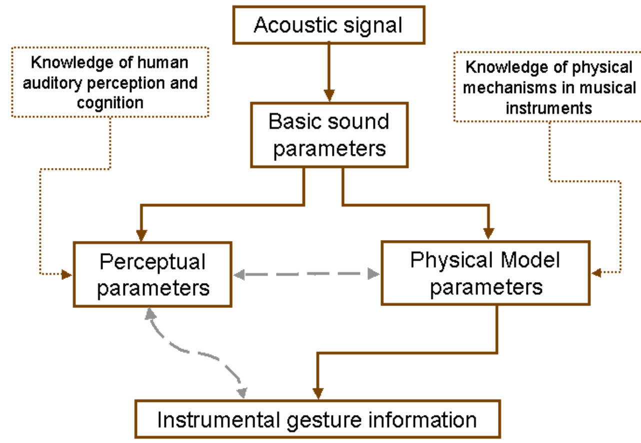



**Figure 2. From acoustic signal to instrumental gesture information.**

In the first stage of the analysis of the data, basic sound parameters are extracted from the acoustic signal, through time- and frequency-domain analysis. These *low-level parameters* include the short-time energy (related to the dynamic profile of the signal), fundamental frequency (related to the sound melodic profile), spectral envelope, amplitudes, frequencies and phases of sound partials, and power spectral density [16]. Using the knowledge of physical mechanisms taking place in musical instruments, physical model parameters are derived from the basic sound parameters. These parameters generally give direct access to the instrumental gesture parameters.

Finally, in order to understand the effect on timbre of the variation of the instrumental gesture parameters, we also use perceptual measures, which we could call *high-level parameters*, as opposed to the low-level parameters defined earlier. These perceptual measures are also derived from basic sound parameters and in particular from the amplitudes of the spectral components. They include widely used measures such as the spectral centroid, spectral irregularity, odd/even harmonic ratio, low/high harmonic ratio, and log-rise time [10]. These parameters are interesting to examine because they are correlated to perceptual attributes such as brightness, metallic quality and nasality. A strong correlation can generally be found between perceptual attributes and instrumental gesture parameters. For example, plucking a string closer to the bridge increases brightness. Modifying the angle of the air jet on the mouthpiece edge of a transverse flute affects brightness as well.

Although this study addresses issues related to the general problem of timbre recognition, the approach that we propose here for the analysis of instrumental timbre differs from the phenomenological approach taken in many timbre recognition systems described in the literature (in [6] for example). Timbre recognition systems implementing neural networks or using principal component analysis require a learning stage, meaning that a timbre can only be identified and labeled by the system after being compared to other typical examples of that timbre. Therefore they do not make explicit the relationships between the physical phenomena, the performer's actions and the obtained timbre. Here, we rather propose to develop analysis tools that use the knowledge that we have about the physical phenomenon taking place in the musical instrument and its effect on the acoustic signal, leading to an analytical model of the interaction between the performer and the instrument (cf. Figure 1).

# 4. APPLICATION TO EXPLORING THE TIMBRE SPACE OF THE CLASSICAL GUITAR

In order to validate the proposed approach for the analysis and understanding of the timbre of a musical instrument and its relationships with the physical phenomena and the performer's gesture, we will present how the approach is applied to the study of the timbre space of the classical guitar. The dimension of that timbre space that we want to start with is the one corresponding to brightness.

The guitar is an instrument that gives the player great control over the timbre. Different plucking techniques involve varying instrumental gesture parameters such as (a) the finger position along the string, (b) the inclination between the finger and the string, (c) the inclination between the hand and the string and (d) the degree of relaxation of the plucking finger. In [12], the author reports three analysis techniques that were used to investigate these four instrumental gesture parameters. Among these analysis techniques, Principal Component Analysis is used to verify that each of the instrumental gesture parameters induces significant changes in the cepstral envelope. However, it is not clear that this methodology can constitute an indirect acquisition system because the four sets of guitar tones were analyzed separately.

In the approach we propose, we rather want to make explicit the correspondences between a perceptual timbre space and a gestural timbre space of the instrument.

## 4.1 Timbre descriptors used by guitar players

As a starting point for the exploration of the timbre space, we want to inquire about the timbre descriptors commonly used by professional musicians. We asked 22 guitarists to define 10 adjectives they commonly use to describe the timbre nuances they can produce on their instrument. We asked the participants to intuitively describe the timbre itself ("How does it sound?") and to describe the gesture associated with it ("How do you make it?"). The compilation of these data lead to an inventory of over 60 adjectives. Dark, bright, chocolatey, transparent, muddy, wooly, glassy, buttery, and metallic are just a few of those adjectives used by guitarists to describe the brightness, the color, the shape and the texture of their sounds.

When playing the guitar, the location along the string where the plucking is performed strongly affects the resulting timbre. If the plucking point is closer to the bridge, the sound is brighter, sharper, more percussive. If the plucking point is closer to the middle of the string or the soundhole, the resulting sound is warmer, mellower, duller, as expressed by expert performers/listeners.

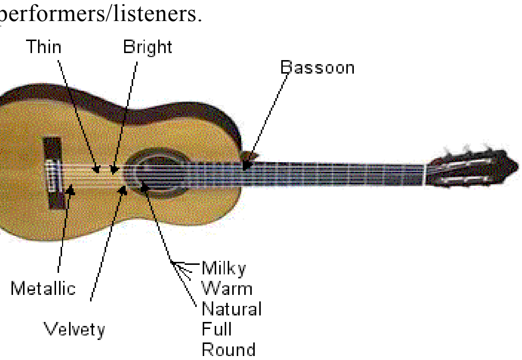



**Figure 3. Timbre descriptors and corresponding plucking locations along the string according to guitarist Zane Remenda (participant in guitar timbre study).**

So, for the case of the guitar, we find that a dimension of the gestural timbre space (the plucking position) clearly corresponds to a dimension of the perceptual timbre space (the brightness). As illustrated on Figure 4, we should be able to check this correspondence by calculating the spectral centroid of the spectrum, which is a measure on the acoustic signal that has been shown to strongly correlate with perceived brightness [10].

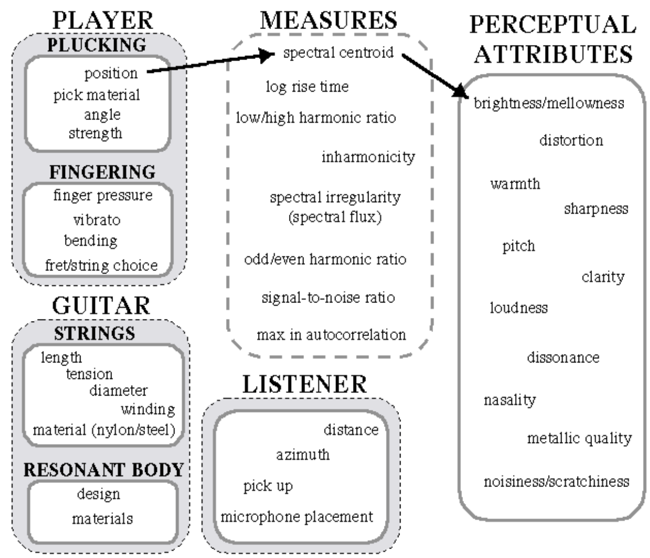



**Figure 4. Factors influencing the timbre of guitar tones.**

Figure 4 also inventories the other factors influencing the timbre of tones played on a guitar. Besides instrumental gesture parameters (characterizing the plucking and the fingering), the materials and the physical features of the instrument itself affect the timbre as well, in the sense that they constrain the palette of timbre nuances that can be achieved by the performer. Finally, the listening conditions also have an impact, due to the particular radiation pattern from the instrument, the characteristics of the microphone (in the case of a recording) and the acoustics of the room.

## 4.2 Acoustic signature of plucking position

The next step in our approach is to learn about the physical phenomenon taking place in the instrument.

Plucking a string sends an acceleration impulse along the string in both directions. Those impulses are reflected at the ends of the string (the bridge on one side and the nut or the finger on the other side). All those impulses combine to form a standing wave on the string. The resultant motion consists of two bends, one moving clockwise and the other counter-clockwise around a parallelogram [7]. In the ideal cases, the output from the string (force at the bridge) lacks those harmonics that have a node at the plucking point. The amplitude $C_n$ of the $n$th mode of the displacement of an ideal vibrating string of length $l$, with an initial vertical displacement $h$ is given by:

$$C_n(h, R) = \frac{2h}{n^2 \pi^2 R(1-R)} \sin(n\pi R) \qquad (1)$$

where $R$ is the relative plucking position, defined as the fraction of string length from the point where the string was plucked to the bridge [13].

The location of the plucking point along a string has an effect on the spectrum of the sound that is similar to the effect of a comb filter. In fact, in a simple digital physical model of a plucked-string instrument, the resonant modes would translate into an all-pole structure, while the initial conditions (a triangular-shaped initial displacement for the string and a zero-velocity at all points) would result in a FIR comb filter structure. The delay of this comb filter is related to the time the wave needs to travel from the plucking point to the fixed end of the string (the bridge or the nut) and back. Therefore, the comb filter delay can be expressed as the product of the relative plucking position $R$ and the fundamental period $T_o$.

The comb filtering effect is illustrated on Figure 5 showing the magnitude spectrum of a guitar tone plucked at 12 cm from the bridge on a 58 cm open A-string. The relative plucking position $R$ is approximately 1/5 ( 12 cm / 58 cm = 1 / 4.83 ). If it was exactly 1/5, all harmonics with indices that are multiples of 5 would be completely missing.

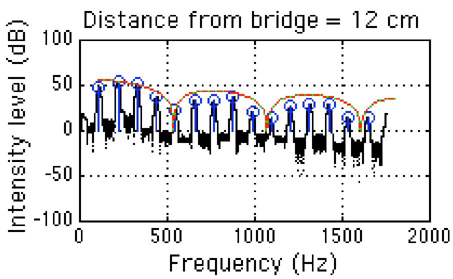



**Figure 5. Magnitude spectrum of a guitar tone plucked at 12 cm from the bridge on a 58 cm open A-string.**

## 4.3 Perceptual effect of plucking position

In order to understand the effect on timbre of the variation of parameters related to the performer's actions, we derive perceptual measures from the basic sound parameters.

As guitarists intuitively associate increasing brightness with decreasing plucking distance from the bridge, we assume that it is possible to check this correspondence by calculating the spectral centroid $SC$ of the spectrum:

$$SC = \frac{\sum_{n=1}^{N} f_n C_n^2}{\sum_{n=1}^{N} C_n^2} \qquad (2)$$

where $C_n$ is the magnitude of the $n$th spectral component and $f_n$ its frequency [10]. Figure 6 displays the plots of the theoretical spectra for various plucking distances, calculated from the theoretical expression of the amplitude of the velocity modes (proportional to $n\ C_n$). We can visually notice that the center of gravity of the spectrum would decrease as the plucking distance from the bridge increases.

This trend is in fact confirmed by the plot displayed on Figure 7, showing the spectral centroid of the theoretical spectra (shown on Figure 6) as a function of plucking distance from the bridge. Also shown on Figure 7 is the spectral centroid curve from the spectra of recorded guitar tones played with different plucking distances. The real data curve follows the same trend as the theoretical curve although the spectral centroid is generally lower.

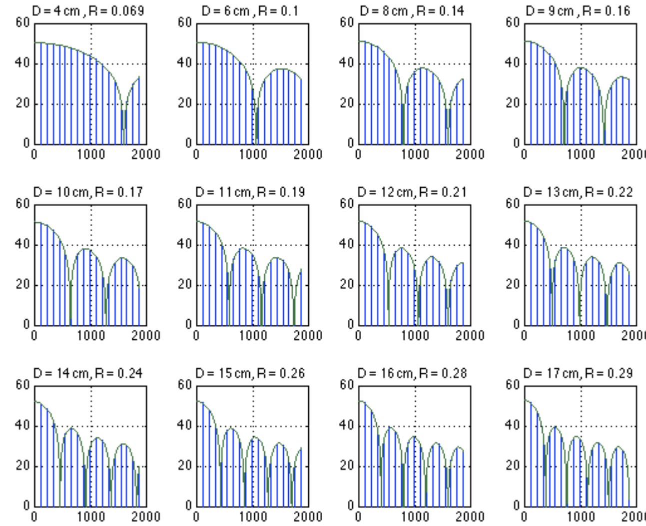



**Figure 6. Variation of theoretical spectral envelope (magnitude in dB vs frequency in Hz) with plucking position.**

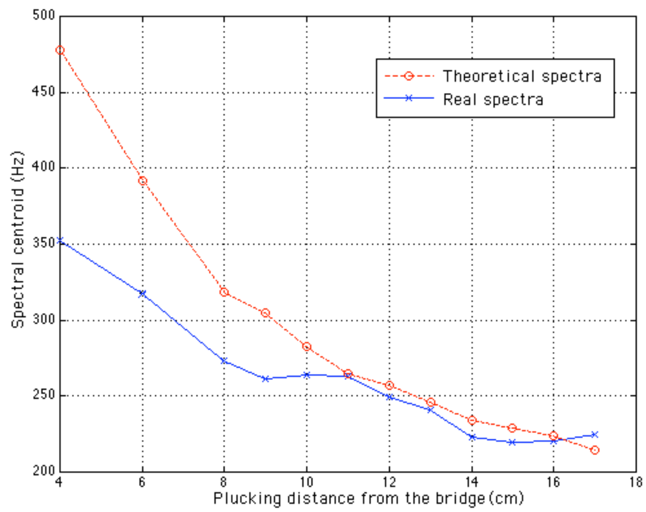



**Figure 7. Variation of spectral centroid with plucking distance from the bridge.**

## 4.4 Indirect acquisition of plucking position

In order to derive the plucking position from the recording of guitar tones, we propose a signal processing method that extracts the location of the zeros in the spectral envelope starting from a FFT-analysis and a measure derived from the autocorrelation. This work adds on to other methods proposed previously and reported in [1], [13] and [14].

The autocorrelation function can be very useful to estimate the fundamental frequency of a periodic signal, since it should show a maximum at a lag corresponding to the fundamental period. Figure 8 displays the plots of the autocorrelation function calculated for 12 recorded guitar tones plucked at various distances from the bridge on an open A-string (fundamental frequency = 110 Hz).

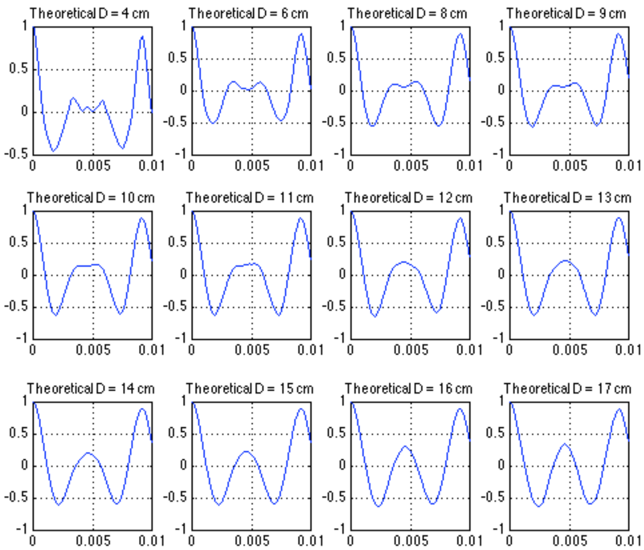



**Figure 8. Autocorrelation graphs for 12 guitar tones plucked at distances from the bridge ranging from 4 cm to 17 cm.**

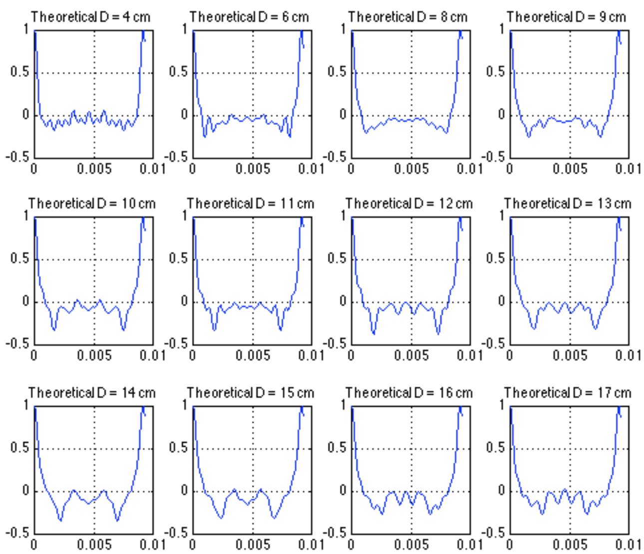



**Figure 9. Log-correlation graphs for 12 guitar tones plucked at distances from the bridge ranging from 4 cm to 17 cm.**

As expected, the graphs show a maximum around 1/110 = 0.009 seconds, the fundamental lag of the autocorrelation. One can also see that the autocorrelation takes on different shapes for different plucking positions but the information about the comb filter delay can not be extracted in an obvious way, directly from these graphs.

To increase the negative contribution of low amplitude harmonics (around the valleys in the comb filter spectral envelope), the log of the squared Fourier coefficients $C_n$ are used to calculate a modified autocorrelation function, that we propose to name *log-correlation* and express as follows:

$$\Gamma(\tau)=\sum_{n=1}^{N}\log\left(C_n^2\right)\cos\left(\frac{2\pi n}{T_o}\tau\right) \qquad (3)$$

Figure 9 displays the log-correlation graphs for the same 12 recorded guitar tones (as for Figure 8). Those plots reveal an interesting pattern: the minimum appears around the location of the lag corresponding to the relative plucking position. We can conclude that the relative plucking position can be approximated by the ratio $R = \tau_{min} / \tau_o$ , where $\tau_{min}$ is the lag corresponding to the global minimum in the first half of the log-correlation period, and $\tau_o$ is the lag corresponding to the fundamental period $T_o$, as shown on Figure 10.

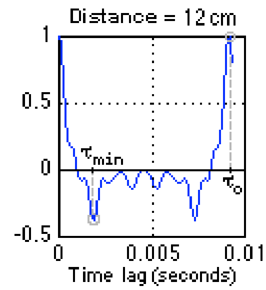



**Figure 10. Log-correlation for a guitar tone plucked 12 cm from the bridge on 58 cm open A-string.**

Figure 11 summarizes the estimation results for the data set of 12 tones. Except for a significant error for the first distance (at 4 cm from the bridge), the estimation is accurate for all other distances (within 1 cm of error). At 4 cm, $R = 4 / 58 = 1 / 14.5$ and the error probably comes from the fact that the spectrum contains only one "zero" over the frequency range that is considered (up to the 15th harmonic).

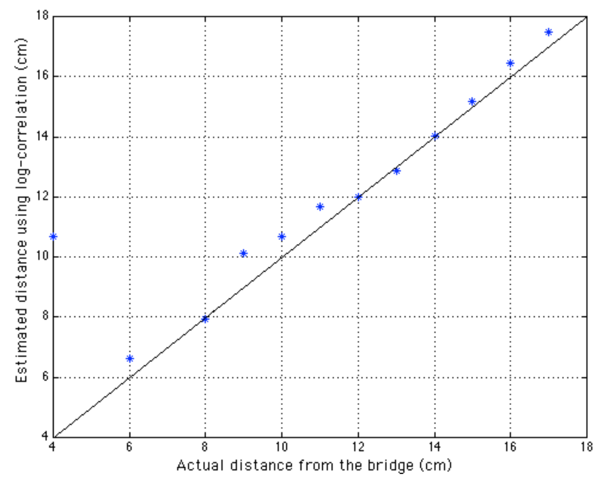



**Figure 11. Plucking point estimation with log-correlation (estimated distance vs actual distance from the bridge)**

## 5. CONCLUSION

In this paper, we have proposed a multi-level approach for the extraction of instrumental gesture parameters from the characteristics of the signal captured by a microphone and based on the knowledge of physical mechanisms taking place on the instrument. Starting from the timbre descriptors commonly used by professional musicians when they verbally describe the sounds they produce with their instrument, we explore the relationships between some features of timbre and gesture parameters. Finally, we presented how this multi-level approach can be applied to the study of the timbre space of the classical guitar. More specifically, we have confirmed the relationship between perceived brightness and decreasing plucking distance from the bridge (intuitively expressed by guitarists) and we have presented a way to extract the plucking position from the signal, which is related to the delay of a comb filter in the physical modeling of the instrument.

The search for other relationships between physical model constituents, instrumental gesture parameters and perceptual attributes would be worth being pursued. Inspired by Grey's timbre space study, a multidimensional scaling analysis of guitar tones could be useful to determine the dimensions of the subspace of guitar timbre nuances.

This works finds applications in the context of hybrid instruments, generating control parameters for physical model based synthesizers and automatic tablature generation.

# 6. APPENDIX

The recorded tones that are used in this study were played with a plastic pick, 0.88 millimeters in thickness and triangular shaped, on a plywood classical guitar strung with nylon and nylon-wrapped steel Alvarez strings. The intended plucking points were precisely measured and indicated on the string with a marker. The tones were recorded with a Shure KSM32 microphone in a sound-deadened room, onto digital audio tape at 44.1 kHz, 16 bits. The microphone was placed in front of the sound hole, approximately 25 cm away, which was far enough to capture a combination of waves coming from different parts of the string, in that way limiting the filtering effect of the pick-up point. A 4096-samples portion was extracted from the middle of the tone (after the attack) and the Fast Fourier Transform analysis was performed with zero-padding factor of 8 and parabolic interpolation. The magnitudes of the first 15 harmonics were used to calculate the log-correlation and the spectral centroid.

# 7. REFERENCES

[1] Bradley, K., M.-H. Cheng, and V.L. Stonick. Automated Analysis and Computationally Efficient Synthesis of Acoustic Guitar Strings and Body. in IEEE ASSP Workshop on Applications of Signal Processing to Audio and Acoustics. 1995.

[2] Cadoz, C. Instrumental Gesture and Musical Composition. in International Computer Music Conference. 1988: International Computer Music Association.

[3] Cadoz, C. and M.M. Wanderley, Gesture - Music, in Trends in Gestural Control of Music, M.M. Wanderley and M. Battier, Editors. 2000, Ircam - Centre Pompidou: Paris, France. p. 71-94.

[4] Delalande, F., La gestique de Gould: éléments pour une sémiologie du geste musical., in Glenn Gould, Pluriel, G. Guertin, Editor. 1988, Louise Courteau Editrice Inc. p. 83-111.

[5] Depalle, P., S. Tassart, and M. M. Wanderley, Instruments virtuels, in Résonance. 1997.

[6] Egozy, E. B., Deriving Musical Control Features from a Real-Time Timbre Analysis of the Clarinet. Master thesis, 1995. Department of Electrical Engineering and Computer Science. Massachussetts Institute of Technology.

[7] Fletcher, N. H. and T. D. Rossing. The Physics of Musical Instruments, 1998, 2d edition, New York, Springer-Verlag.

[8] Gabrielsson, A. The Performance of Music. In Diana Deutsch (ed) The Psychology of Music, 1999, San Diego, CA: Academic Press. Second Edition, pp. 501-602.

[9] Grey, J.M., Multidimensional perceptual scaling of musical timbres. Journal of the Acoustical Society of America, 1977. 61(5): p. 1270-1277.

[10] Hajda, J.M., et al., Methodological issues on timbre research, in Perception and Cognition of Music, I. Deliège and J. Sloboda, Editors. 1996, Pychology Press, Oxford: Hove, UK. p. 253-306.

[11] Machover, T., Hyperinstruments - A progress Report 1987-1991. 1992, MIT Media Laboratory Massachussets Institute of Technology.

[12] Orio, N. The timbre space of the classical guitar and its relationship with the plucking techniques. in ICMC - International Computer Music Conference. 1999.

[13] Traube, C. and J.O. Smith. Estimating the plucking point on a guitar string. in Conference on Digital Audio Effects (DAFX00). 2000. Verona, Italy.

[14] Traube, C. and J.O. Smith. Extracting the fingering and the plucking points on a guitar string from a recording. in IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA'01). 2001. Mohonk Mountain House, New Paltz, New York.

[15] Wanderley, M.M., Gestural Control of Music. Proceedings of the International Workshop on Human Supervision and Control in Engineering and Music. Kassel, Germany, pp. 101-130.

[16] Wanderley, M.M. and P. Depalle, Contrôle Gestuel de la Synthèse Sonore, in Interfaces Homme-Machine et Création Musicale. 1999, Hermes Science Publications. p. 145-163.

# Bio-Sensing Systems and Bio-Feedback Systems for Interactive Media Arts

Yoichi Nagashima
Shizuoka University of Art and Culture / Art and Science Laboratory
1794-1 Noguchi-cho, Hamamatsu, Shizuoka, Japan
430-8533
nagasm@computer.org

## ABSTRACT

This is a report of research and some experimental applications of human-computer interaction in multi-media performing arts. The human performer and the computer systems perform computer graphic and computer music interactively in real-time. In general, many sensors are used for the interactive communication as interfaces, and the performer receives the output of the system via graphics, sounds and physical reactions of interfaces like musical instruments. I have produced many types of interfaces, not only with physical/electrical sensors but also with biological/physiological sensors. This paper is intended as an investigation of some special approaches: (1) 16-channel electromyogram sensor called "MiniBioMuse-III" and its application work called "BioCosmicStorm-II" performed in Paris, Kassel and Hamburg in 2001, (2) sensing/reacting with "breathing" in performing arts, (3) 8-channel electric-feedback system and its experiments of "body-hearing sounds" and "body-listening to music".

## 1. INTRODUCTION

As the research called PEGASUS project (Performing Environment of Granulation, Automata, Succession, and Unified-Synchronism), I have produced many systems of real-time performance with original sensors, and have composed and performed many experimental works at concerts and festivals. The second step of the project is aimed "multimedia interactive art" by the collaboration with CG artists, dancers and poets.

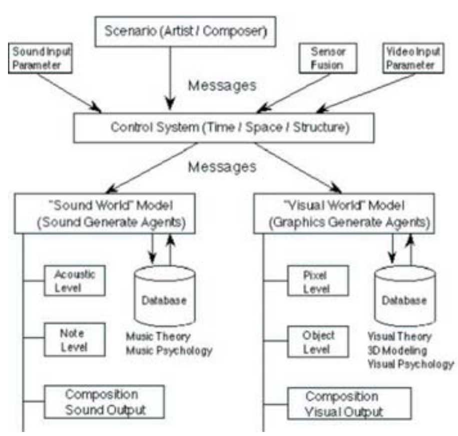



**Figure 1: Conceptual system block diagram of PEGASUS project, multimedia interactive art.**

Fig.1 shows the concept of the keyword: "listen to the graphics, watch the music". The third step of the project is aimed "biological or physiological interaction between human and system". I had produced (1) Heart-beat sensor by optical information at human earlobe, (2) Electrostatic touch sensor with metal contacts, (3) single/dual channel electromyogram sensor with direct muscle noise signals. And now I report the newer sensors in this paper.

## 2. MINI BIO MUSE - III

At first I report the development of a compact/light 16-channels electromyogram sensor called "MiniBioMuse-III" (Fig.2). This sensor is developed as the third generation of my research in electromyogram sensing, because there are many problems in high-gain sensing and noise reduction on stage (bad condition for bio-sensing).

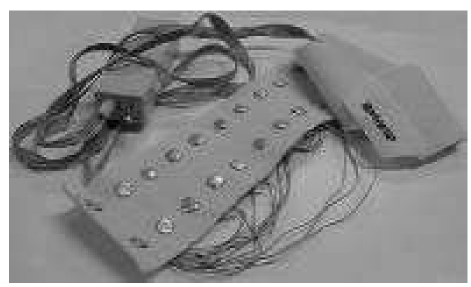


**Figure 2: "MiniBioMuse-III" (only for one arm)**

### 2.1 Development of "MiniBioMuse-III"

The front-end sensing circuit (Fig.3) is designed with heat-combined dual-FETs, and cancels the common-mode noises. There are 9 contacts on one belt, one is common "ground".

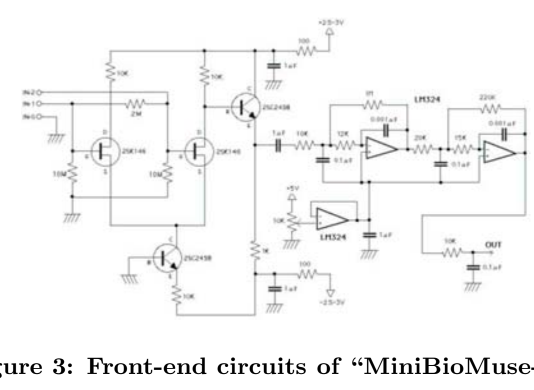


**Figure 3: Front-end circuits of "MiniBioMuse-III"**

**Figure 4: High-density assembled front-end circuits (8ch)**

Each 8-channel electromyogram signals for one arm/hand (fig.4) is demultiplexed and converted to digital information by 32bits CPU, and converted to MIDI information for the system (fig.5). This system also generates 2 channel Analog voltage outputs for general purpose from MIDI inputs.

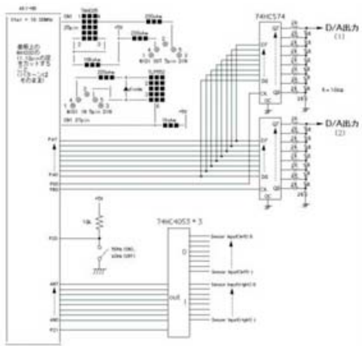


**Figure 5: 32bits CPU circuits of “MiniBioMuse-III”**

This CPU also works as software DSP to suppress the Ham noise of environmental AC power supply. Figure 6 shows its algorythm of ”Notch Filter” for noise reduction.

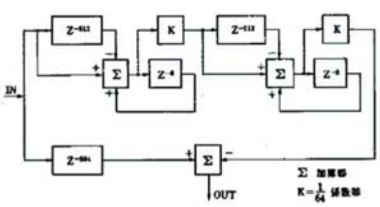


**Figure 6: Noise reduction algorythm of CPU**

## 2.2 Output of “MiniBioMuse-III”

Figure 7/8 shows the output analog signals of front-end circuits of “MiniBioMuse-III”. Figure 7 is the “relax” state of the performer, and figure 8 is “hard-tension” state of the performer.

Figure 9 is the Max/MSP screen of the MIDI output of this sensor. 8channels + 8channels electromyogram signals are all displayed in real time, and used for sound generating parameters. The maximun sampling rate of this sensor is about 5msec, but this sampling rate is changed by the host system for its ability of MIDI receiving.

## 2.3 Performance with “MiniBioMuse-III”

I have composed one new work with this “MiniBioMuse-III” for my Europe Tour in September 2001. The title of the work is “BioCosmicStorm-II”, and the system is constructed in Max/MSP environments. 16 channel electromyogram signals are all displayed in Max/MSP screen and projected on stage, so audience can easily understand the relations between sound and performance. This work was performed in Paris (CCMIX), Kassel and Hamburg. Generated sounds are three types in scenes : (1) 8+8 channels bandpass-filters with white noises, (2) 16 individual-pitch sine-wave genarators, and (3) 3+3 operators and 10 parameters of FM synthesis generators. All sounds are real-time generated with the sensor.

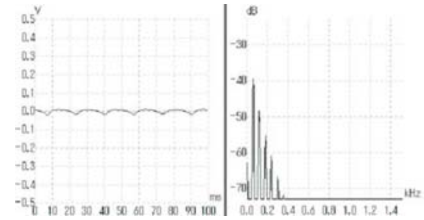


**Figure 7: Sensing output signals of “relax”**

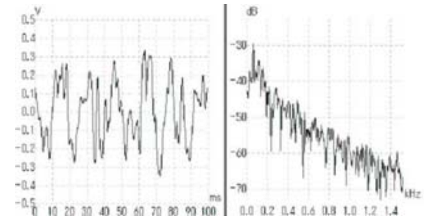


**Figure 8: Sensing output signals of ”hard tension”**

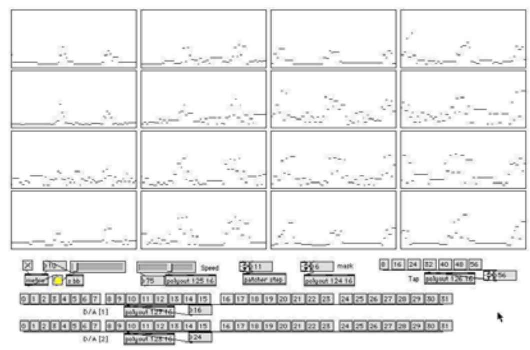


**Figure 9: MIDI output of ”MiniBiouse-III” (Max/MSP)**

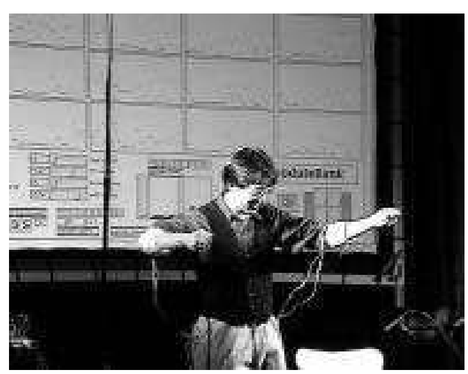


**Figure 10: Performance of “BioCosmicStorm-II”**

## 2.4 Parameters Mapping in Composition

Figure 11 shows the main patch of Max/MSP for the work "BioCosmicStorm-II" in running mode. There are 16 real-time display windows of 16 channel EMG inputs. Figure 12 shows the same patch in editing mode. The 16 sensor outputs were mapped into "sinusoidal synthesis" part of this work is programmed in this main patch using the Hide-mode objects.

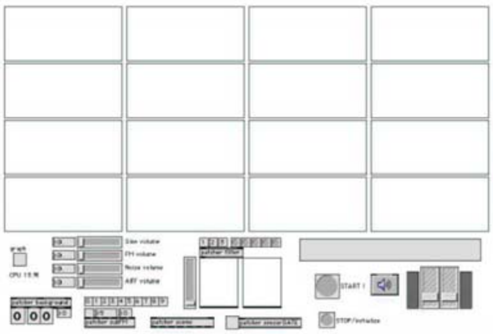


**Figure 11: Main Patch of "BioCosmicStorm-II"**

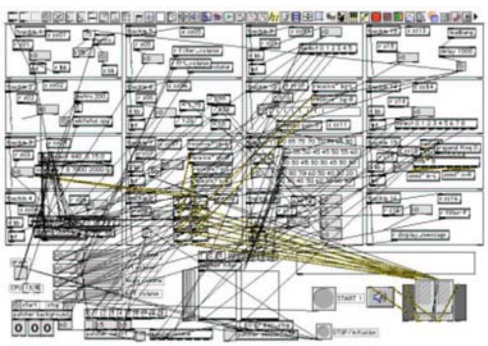


**Figure 12: Edit Mode of Main Patch**

Figure 13 shows the sub-patch of the work for the "FM synthesis" part. This FM algorithm is designed by Suguru Goto and distributed by himself in DSPSS2000 in Japan. The multiband filters were also used in this work.

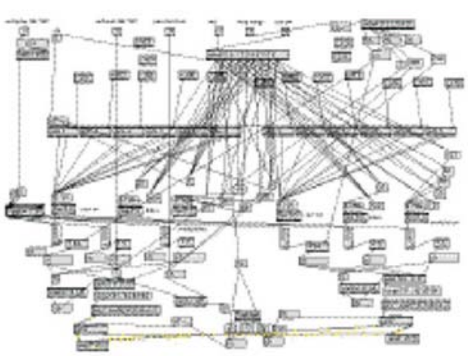


**Figure 13: SubPatch for FM Synthesis Block**

# 3. "BREATHING MEDIA"

In computer music performance, the human performer generates many information which computer system can detect, but "sound" and "image" of performance have fatal problems of its delay. The final sound of the performance and image of the movements of performer are detected just after its generation, and the system has limited conversion time and limited computation time, so the performer feels the delay of response in every time. Thus I have developed two types of new sensors with which computer system can detect the actions before by sound or by image of the performance.

## 3.1 Vocal Breath Sensor

Vocal performer acts with heavy breathing, and her(his) breast and belly repeats expansion and contraction. So I used rubber tube sensor (Fig.14) which changes its resistance with the tension, and produced the Vocal Breath Sensor system to convert the breathing information to MIDI in real-time (Fig.15).

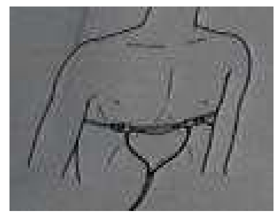


**Figure 14: Rubber Tube Sensor**

**Figure 15: "Vocal Breath Sensor" System**

The sensing information (Fig.16) is used to change signal processing parameters of her voices and to arrange parameters of real-time computer graphics on stage. The audience can listen to her voice and watch her behavior with the tight/exaggerated relation effected and generated by the system which detects the changes before the sound.

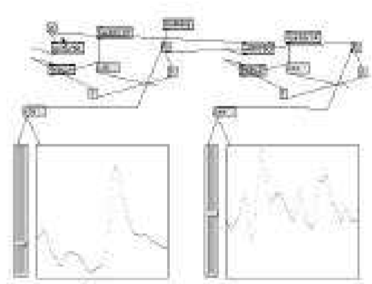


**Figure 16: Output of "Vocal Breath Sensor"**

## 3.2 SHO Breath Sensor

I report the development of a compact/light bi-directional breath pressure sensor for SHO (Fig.17). SHO is the Japanese traditional musical instrument, a mouth organ. The SHO player blows into a hole in the mouthpiece, which sends the air through bamboo tubes which are similar in design and produce a timbre similar to the pipes in a western organ. The bi-directional breath pressure is measured by an air-pressure sensor module, converted to digital information by 32bits CPU, and converted to MIDI information.

**Figure 17: SHO Breath Sensor**

The authoring/performing system displays the breathing information in real-time, and helps the performer for delicate control and effective setting of the parameters (Fig.18). The output of this sensor shows not only (1) the air-pressure inside the SHO, and (2) the volume of SHO sound of course, but also (3) preliminary preparation operation and mental attitude of the performer, so it is very important for the system to detect this information before sound starts.

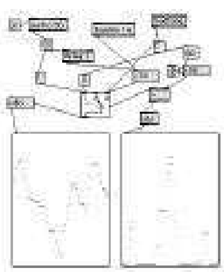


**Figure 18: Example of SHO Breath Data**

## 3.3 Performances with SHO Breath Sensor

**Figure 19: Performance of "Visional Legend" (Kassel)**

This bi-directional SHO breath sensor is originally produced by myself, and used by Japanese SHO performer/composer Tamami Tono Ito. The "Breathing Media" project is her own. I have composed interactive multimedia art called "Visional Legend" which was performed in Kassel (2 concerts) and Hamburg in September 2001 (Fig.19). She composed some works using this sensor, and Figure 20 shows the work "I/O". Her breath controls both sound synthesis and live graphics.

**Figure 20: Performance of "I/O" (SUAC Japan)**

# 4. BIO-FEEDBACK SYSTEM

Finally, I report the newest development of a compact and light 8-channel biological feedback system (Fig.21). The feedback signal is high voltage (10V-100V) electric pulses like "low frequency massage" device (Fig.22-23)). The wave-shape, voltage and density of pulses are real-time controlled with MIDI from the system. The purposes of this feedback are: (1) detecting performer's cues from the system without being understood by audience, (2) delicate control of sounds and graphics with the feedback feeling in virtual environment, (3) live performance of outside of anticipation with the electric trigger.

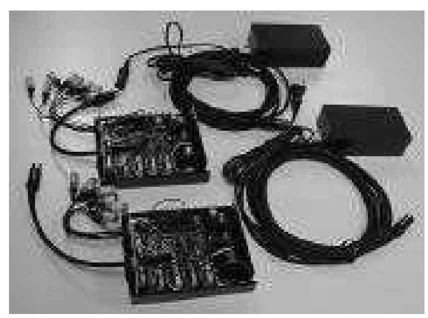


**Figure 21: Bio-Feedback System**

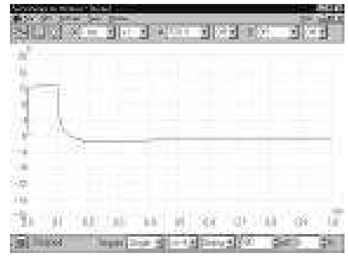


**Figure 22: Example of Bio-Feedback signal**

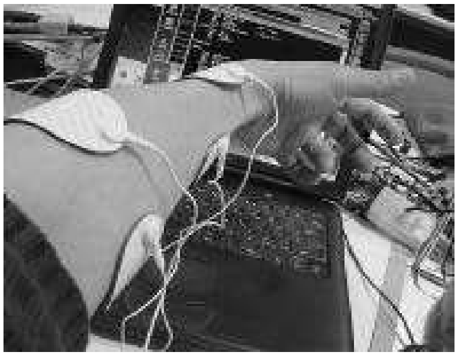


**Figure 23: Bio-Feedback contacts**

## 4.1 Application Example (1)

Figure 24 shows the performance of the work “It was going better If I would be sadist truly.” composed and performed by Ken Furudachi in February 2002 in Japan. There were 2 DJ (scratching discs) performers on stage, and the DJ sounds generates many types of bio-feedback signals with Max/MSP and this system. The performer shows the relation between input sounds and output performance just by his body itself. This work is the first application of the system.

**Figure 24: Performance of “It was going better If I would be sadist truly.”**

## 4.2 Application Example (2)

Figure 25 shows the performance of the work called “Flesh Protocol” composed by Masayuki Akamatsu and performed by Masayuki Sumi in February 2002 in Japan. The performer is a professional dancer, so he can receive two times bigger electronic pulses with his strong and well-trained body. The composer produces many noises and sounds with Max/MSP, and the converted signals control the body of the performer on stage. The relations of them are well shown in real-time with the screen and motions on stage.

**Figure 25: Performance of “Flesh Protocol.”**

## 4.3 Application Example (3)

Figure 26 shows the performance of the work called “Ryusei Raihai” composed by Masahiro Miwa in March 2002 in Japan. The four performers connected to the system are ”instruments” of the special message in Internet with the composer’s filtering program. When one special data occurs in the network, one of the performers is triggered by the system, then he/she plays bell on the hand in real-time.

**Figure 26: Performance of “Ryusei Raihai”**

## 4.4 Possibility of “Hearing pulse”

I want to discuss about a possibility of ”hearing pulse” without using ears. In experiments during development of this system, I found many interesting experiences to detect ”sounds” without acoustic method (speaker, etc). The numbed ache from this Bio-feedback system is different with the waveshape, frequency etc. This shows the possibility for hearing-impaired person that a sound can be perceived without using an ear. Another experiment, the sourse is changed from simple pulses to musical signals also shows the possibility of listening to the music with this feedback.

### 4.5 Combined force display system

It is well known that EMG sensor is a good instrument, but there is a weakpoint compared with other "mechanical" instruments. EMG sensor does not have the "physical reaction of performance" like guitar, piano, etc. To resolve this, one musician generates very big sound as the body-sonic feedback of the performance. I have just started researching the combination of EMG sensor and bio-feedback system with the same electrode using time-sharing technique. Figure 27 shows the block diagram for the combination. Card-size controller "AKI-H8" contains 32bits-CPU, A/D, D/A, SIO, RAM, FlashEEPROM and many ports. Electrode channels (max:8ch) are multiplexed for A/D and D/A. "High impedance of separation, high input voltage" analog switch is controlled by the CPU ports synchronizing with the system.

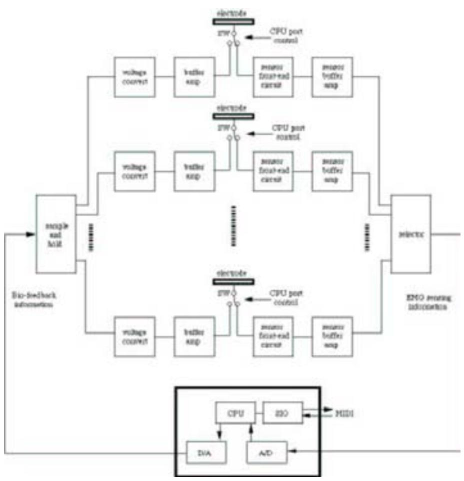



**Figure 27: Blockdiagram of "Combined System"**

## 5. CONCLUSIONS

Some researches and experimental applications of human-computer interaction in multi-media performing arts were reported. Interactive multi-media art is the interesting laboratory of human interfaces and perception/cognition researches. So I will continue these researches with many experiments.

## 6. REFERENCES

[1] Y. Nagashima. PEGASUS-2: Real-Time Composing Environment with Chaotic Interaction Model. In *Proceedings of ICMC*, ICMA, 1993.

[2] Y. Nagashima. Multimedia Interactive Art : System Design and Artistic Concept of Real-Time Performance with Computer Graphics and Computer Music. In *Proceedings of Sixth International Conference on Human-Computer Interaction*, ELSEVIER, 1995.

[3] Y. Nagashima. A Compositional Environment with Interaction and Intersection between Musical Model and Graphical Model – Listen to the Graphics, Watch the Music. In *Proceedings of ICMC*, ICMA, 1995.

[4] Y. Nagashima. Real-Time Interactive Performance with Computer Graphics and Computer Music In *Proceedings of 7th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design, and Evaluation of Man-Machina Systems*, IFAC, 1998.

[5] Y. Nagashima. BioSensorFusion: New Interfaces for Interactive Multimedia Art In *Proceedings of ICMC*, ICMA, 1998.

[6] Y. Nagashima. 'It's SHO time' – An Interactive Environment for SHO(Sheng) Performance In *Proceedings of ICMC*, ICMA, 1999.

[7] Y. Nagashima. Composition of "Visional Legend" http://nagasm.suac.net/ASL/kassel/index.html

[8] C. Bahn, T. Hahn, and D. Trueman. Physicality and Feedback: A Focus on the Body in the Performance of Electronic Music In *Proceedings of ICMC*, ICMA, 2001.

[9] R. Boulanger and M. V. Mathews. The 1997 Mathews' Radio Baton and Improvisation Modes In *Proceedings of ICMC*, ICMA, 1997.

[10] A. Camurri. Interactive Dance/Music Systems In *Proceedings of ICMC*, ICMA, 1995.

[11] J. Chadabe. The Limitations of Mapping as a Structural Descriptive in Electronic Instruments In *Proceedings of NIME*, 2002.

[12] C. Chafe. Tactile Audio Feedback In *Proceedings of ICMC*, ICMA, 1993.

[13] R.J.K. Jacob. Human-Computer Interaction: Input Devices In *Input Devices, ACM Computing Surveys 28(1)*, 1996.

[14] S. Jorda. Improvising with Computers: A Personal Survey (1989-2001) In *Proceedings of ICMC*, ICMA, 2001.

[15] T. Kanamori, H. Katayose, S. Simura, and S. Inokuchi. Gesture Sensor in Virtual Performer In *Proceedings of ICMC*, ICMA, 1993.

[16] T. Kanamori, H. Katayose, Y. Aono, S. Inokuchi, and T. Sakaguchi. Sensor Integration for Interactive Digital Art In *Proceedings of ICMC*, ICMA, 1995.

[17] T. Kanamori, H. Katayose, Y. Aono, S. Inokuchi, and T. Sakaguchi. Virtual Performer In *Proceedings of ICMC*, ICMA, 1993.

[18] R. B. Knapp and H. S. Lusted. A Bioelectric Controller for Computer Music Applications In *Computer Music Journal 14(1)*, 1990.

[19] R. B. Knapp and H. S. Lusted. *The Computer Music Tutorial* Cambridge, MA: The MIT Press, 1996.

[20] R. Rowe. *Interactive Music Systems - Machine Listening and Composing* Cambridge, MA: The MIT Press, 1993.

[21] A. Tanaka. Musical Technical Issues in Using Interactive Instrument Technology with Application to the Biomuse In *Proceedings of ICMC*, ICMA, 1993.

# Characterizing and Controlling Musical Material Intuitively with Geometric Models

Ali Momeni
Center for New Music and Audio Technologies
Department of Music (CNMAT)
University of California Berkeley
1-510-643-9990
ali@cnmat.berkeley.edu

David Wessel
Center for New Music and Audio Technologies
Department of Music (CNMAT)
University of California Berkeley
1-510-643-9990
wessel@cnmat.berkeley.edu

## ABSTRACT
In this paper, we examine the use of spatial layouts of musical material for live performance control. Emphasis is given to software tools that provide for the simple and intuitive geometric organization of sound material, sound processing parameters, and higher-level musical structures.

## Keywords
Perceptual Spaces, Graphical Models, Real-time Instruments, Dimensionality Reduction, Multidimensional Scaling, Live Performance, Gestural Controllers, Live Interaction, High-level Control.

## 1. INTRODUCTION
Spatial arrangement is a natural way to organize things. Similar objects or objects that belong together are placed proximate to each other and dissimilar or disassociated objects are kept at a distance. Maps are useful for exploration and without them we are quite likely to miss the most interesting paths from here to there. In this paper we describe the development of some general tools that bring such cognitively compelling spatial metaphors to the problem of characterizing and controlling musical material in an intuitive way.

Abstract geometric descriptions of musical material have been around for some time. Shepard [14], Krumhansl [5], and Lerdahl [7] provide numerous examples of organizing pitch in a spatial manner. Cognitive psychologists have made extensive use of techniques like multidimensional scaling to map perceptual and cognitive structures. The study of music perception and cognition has benefited greatly from such techniques. Timbre, rhythm, harmony, and texture spaces have enriched our understanding of the behavior of musical material [19].

Our goal goes beyond the search for insight into the perceptual and cognitive structure of musical material. We are interested in musically expressive control in a real-time live-performance context as well. Geometric models of low dimensionality, say one, two, or three dimensions, provide natural hooks for a large class of controllers such as joysticks, tablets, gloves, etc. The problem is that most interesting musical objects such as timbres, rhythms, and processing algorithms have considerably higher dimensionality. As a result, musically sensible dimensionality reduction is a central focus of our research.

In this paper we present a number of different musical material spaces. We describe tools that make their design easy and their use in performance potentially musically expressive. What is new about this work is that it brings spatial metaphor modeling and dimensional reduction techniques to the actual practice of composing and performing music.

## 2. HISTORY
The application of multidimensional scaling and related geometric models to audio has a long and rich history and we can touch on but a few of the highlights here. Most of the early studies in the 1960's [9, 10] were carried out by experimental psychologists interested in understanding the mechanisms of auditory perception. Many of the these early studies were purely psychoacoustic in character involving steady state tones and had little direct application to music or even the study of music perception. Other studies, like that of Wedin and Goode [17] used tones from traditional acoustic instruments but were not really carried out with an application to composition in mind. The first published reference we are aware of to the compositional application of multidimensional scaling was made by Milton Babbitt [1].

In the early 1970's one of the authors [20] proposed that the spatial layouts of timbres from multidimensional scaling could function as a palette of musical material whose organization could provide intuitive navigational advice to the composer. John Grey followed suit with his landmark PhD thesis and subsequent experiments [3]. After a number of experimental replications confirmed the basic structure of a timbre space for harmonic acoustic instrument sounds playing the same pitch and loudness [21] the challenge was to show that the spatial layout could actually make predictions about perception of musically viable sequences. Wessel demonstrated that auditory stream formation and rhythmic organization of *klangfarben* sequences could be predicted from a timbre space. It was also demonstrated that a timbre space could be used to specify perceivable timbral transpositions [8, 21].

In 1978, Jean-Claude Risset created a timbre space using multidimensional scaling and used it to compose passages of his work *Mirages* for the Ensemble Intercontemporain. This we believe to be the first application of the technique to a large-scale composition. This piece was performed in the concerts celebrating the official opening of IRCAM that fall. Other compositional applications followed but admittedly the practice of using perceptual spaces for composition has not fallen into general use. The technique, as it was, is far too tedious.

## 3. MOTIVATION AND JUSTIFICATION
Our goal here is the make the use of geometric models for the characterization and eventual control of musical material more approachable. The first thing that had to go was the tedium of making pair-wise dissimilarity judgments. Consider that if we wish to work with a 100 different percussion sounds by the classical methods we would be

required to make 4950 dissimilarity judgments. The second necessity is a user interface that combines the process of producing the geometric model with the use of the space in real-time performance.

In the mid 1980's experiments in laying out two-dimensional timbre spaces by arranging the sound objects on a screen proved successful.

The spaces laid out in this simple intuitive manner were shown to be very much like those generated by the multidimensional scaling of pair-wise dissimilarity judgments [6, 18]. Further justification for this spatial layout technique is provided by Goldstone [2].

## 4. SOFTWARE IMPLEMENTATION AND APPLICATIONS

A space designer and explorer was implemented using Cycling 74's Max/MSP and Jitter software. The main patch, named *space-master*, allows one to design a 2-d perceptual space made up of a number of *objects*. Each object can be a recorded sample, a single number, or a list of numbers. Each data point is also the center of a Gaussian kernel, whose value at any given point in the space indicates the weight of its associated data point in the interpolated mixture. The result is a 2-d space that allows weighted interpolation among all data points based on the values of the Gaussian kernels at each point in the space. This implementation aimed to meet several design goals: 1) one unified environment for both design and exploration of a space, 2) a space-designing environment that allows auditioning and real-time adjustments in locations and weights of each data point, 3) the ability to manage large sets of data, 4) the possibility of compelling and mutable graphic representations, 5) the ability to automate movements in the space, 6) the ability to easily save, recall and adjust any parameter in the spaces and to switch among spaces with ease, and 7) the ability communicate with other applications for building, modifying or analyzing spaces. The introduction of Jitter, a set of externals for Max/MSP that allow storage, manipulation and visualization of matrices, made these design goals possible.

### 4.1 Designing And Using a Space

This section describes the process of designing a space, entering data into the space and using the space. We go into some detail about the data storage methods used in this implementation, as well as some programming details specific to the Max/MSP/Jitter environment.

#### *4.1.1 What Is a Space?*

A space is made up of a set of Gaussian kernels whose centers, amplitudes and standard deviations are specified by the user. Each Gaussian kernel is associated with a list of floating point numbers—coordinates in a high-dimensional space—that are also specified by the user. The space is visualized in two dimensions by an image that is a bird's-eye projection of all the Gaussian kernels onto a plane. The space is also visualized by a 3-d surface plot of the kernels.

#### *4.1.2 Creating a Space*

The process of designing a space begins by opening the patch *space-master* (Figure 1). The user first defines the "list-length" parameter (dimensionality) and places a desired number of points onto the space using the patch's graphical interface.

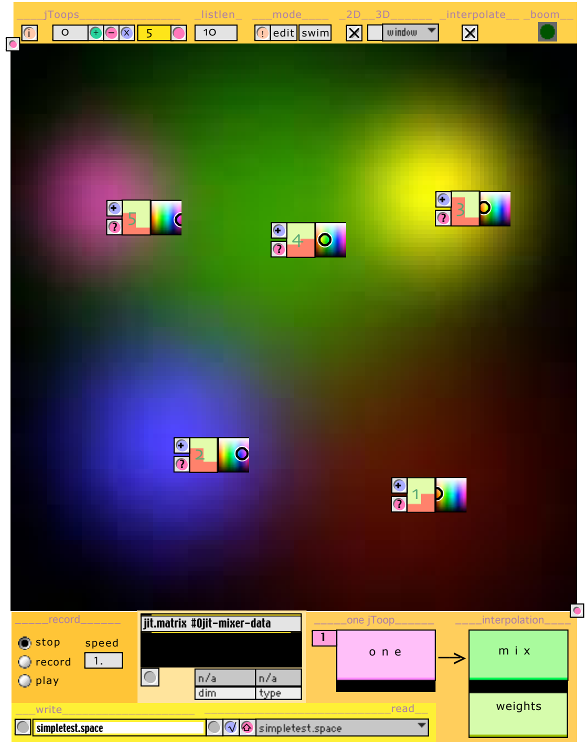



**Figure 1. The figure shows the main interface for the *space-master* patch. The large black square is the space being designed; the five rectangular objects in the space are five *one-point* abstractions; the two sliders behind each *one-point's* ID number (1 through 5) are the Gaussian kernel's amplitude and standard deviation. The figure shows the 2-d representation of these kernels as colored regions whose center is beneath the top left corner of each *one-point* abstraction.**

The patch dynamically creates—or destroys—an abstraction called *one-point* for each data point in the space. At its creation time, each *one-point* instance is given an ID number and is linked to the main patch. Each *one-point* abstraction has its own user interface which includes sliders for the amplitude and standard deviation of the Gaussian kernel as well as a color swatch for the graphical representation of the kernel. With the main patch in "edit" mode, the user then moves each *one-point* instance to a desired location in the space, selects an amplitude and standard deviation for the kernel, selects a color and clicks the "+" button in the top left of the abstraction. This creates a Gaussian kernel centered at that point (the current mouse location) and links that kernel to the data point whose "+" button was clicked. The background image of the space is updated to show a 2-d representation of all the kernels in the space by mapping the height of the kernels to a brightness scale applied to the selected color of that kernel. For more accurate visualization of the Gaussian kernels, *space –master* also renders the kernels in 3-d (Figure 2). Gaussian or similar kernels provide not only a mechanism for interpolation but also for extrapolation beyond the perimeter of the points in the space.

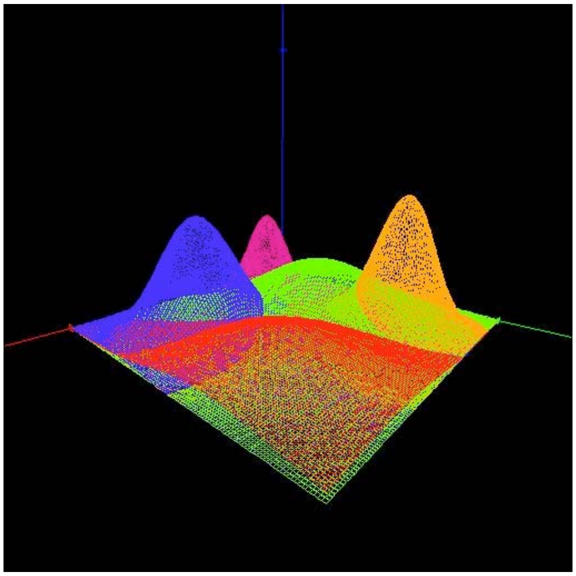


**Figure 2. This figure shows a 3-d representation of the same space as that in Figure 1. The color of each Gaussian kernel corresponds to its color in the 2-d representation.**

Auditioning while designing a space is a necessary capability. In our implementation, each *one-point* has a button labeled "?" which, when clicked, sends that *one-point's* ID to a global receiver named *space-master-doer*. The destination receive object can then reside in any other Max/MSP patch and perform any desired playback, synthesis or calculation that is appropriate to the space at hand.

Once the space is created, the user puts the patch in "swim" mode; this converts the image representing the space in 2-d to a JPEG format image file. This image is assigned as the background image for a 'pictslider' graphical interface object, a 2 dimensional slider that allows interaction using the mouse or incoming control values.

The space is now ready to be saved; after providing a filename (we will use the name *simpletest.space* for this example) and pressing the "W" button, 4 files are created on the hard disk: three .jxf files (Jitter's binary file format for matrices) that contain the *space, data,* and *points* matrices, and a JPEG file with an image of the 2-d representation of the space. A naming convention is used for consistency: a space by the name of *simpletest.space* would be comprised of four files:

*simpletest.space.points.jxf*

*simpletest.space.data.jxf*

*simpletest.space.space.jxf*

*simpletest.space.space.jpeg*

### *4.1.3 Entering Data into the Space*

In certain applications of spatial layouts, the weight of each data point at any given coordinate in the space is all that is necessary for performing the interpolation (e.g. see section 4.2.1 below). For these applications, the space is now complete and ready to be used. However, this system also allows each point to represent a list of floating point numbers; it then uses the weight of each data point at a space coordinate as a weight for its associated list in the overall weighted-interpolated mixture of all the lists. For these applications, the user must also provide the lists of floating point numbers among which to interpolate.

Now that the points have been placed in the space and kernels have been created, we are ready to enter data into the data-matrix for the space. This is done within the Max/MSP programming environment by making an instance of the *space-master* patch with an argument indicating the name of the space, e.g. *simpletest.space*. The user communicates with the *space-master* patch using a set of OpenSoundControl messages [22]. For entering data into the space the message '/store' is used: the '/store' messages must be followed by an integer and a list of floating point numbers. The list is then linked to the data point whose ID is the provided integer. For example, '/store 3 0.2 1.4 .8' would store the list '0.2 1.4 .8' as the data for point number 3 in the space.

Once the user has entered lists for all of the points in the space, the space is ready to be used to interpolate among the lists.

### *4.1.4 Interpolating Among Stored Lists*

The '/lookup' message is used for triggering calculations by *space-master*. The message '/lookup' must be followed by two integers. These integers represent the x and y coordinates in the space at which we wish to make an interpolation.

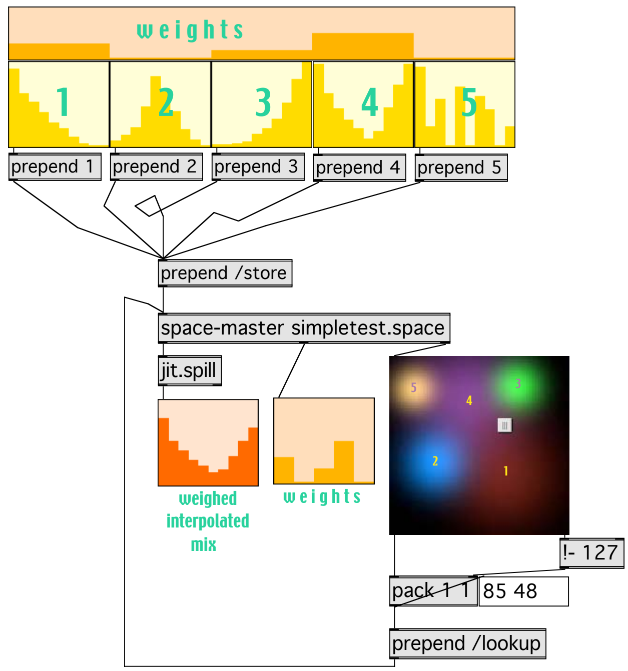



**Figure 3. A Max/MSP patch that stores 5 lists into *space-master* and then uses a two dimensional slider to interpolate between the lists.**

The '/lookup x y' message will output a list of numbers between 0 and 1 that represent the normalized values of each kernel at the point (x, y) in the space. This list comes out of the *space-master's* second outlet. In a space with 5 data points this list would have 5 items. The '/lookup x y' message will also trigger the patch to calculate the resulting weighted interpolation between all stored lists. That is, the normalized value of each point's kernel at (x, y) is used as the weight for that point's associated list in the weighted-interpolated list. The calculated list comes out of the first outlet of *space-master* (Figure 3).

### *4.1.5 Data Storage*

The main patch uses three Jitter matrices to store all of the data associated with a space:

*data-matrix*: A 2-d, 1-plane matrix whose rows contain the lists of floating point numbers among which we want to interpolate; in a space with 5 points where each point represents a list of 10 numbers, this would be a 10x5 matrix.

*space-matrix*: A 3-d, 1-plane matrix that contains the height of the 2-d Gaussian surface for each data point in the space. By default, the designed interpolation spaces are 128x128 points in size; therefore, in our example space with 5 lists of 10 items, this would be 128x128x5 matrix.

points-matrix: A 1-d, 9-plane matrix whose cells contains all the information associated with each data point in the space:

1) Whether or not the point is active

2) Size of interpolation space (default 128)

3-4) x and y location of the point in the space

5-6) Amplitude and standard deviation of the Gaussian kernel

7-9) RGB values for the point's graphic representation

This matrix it is used to reconstruct a space for making changes.

The three .jxf files created when the user saves a space correspond to the above three matrices.

It is worthy noting that the chosen data storage mechanism not only makes calculations efficient within Max/MSP, it also provides a hook for creating spaces in other applications besides Max/MSP. Although the described method for creating spaces is extremely useful for designing with one's intuition (i.e. spaces based on subjective similarity measures), it can be impractical for very large sets of data. There are, however, numerous computational techniques in the large body of research that addresses the fundamental problem of dimensionality, that allow one to derive locations in a 2-d space from large sets of high-dimensional data. These algorithms are often implemented in applications like MatLab. We have developed a number of auxiliary Max/MSP patches which translate matrices exported from another application as delimited text files, into Jitter matrices ready to be used by *space-master*.

### *4.1.6 More Detailed Notes on Max/MSP/Jitter Implementation*

When designing a space, the *one-point* abstractions created for each point in the space are instantiated using Max/MSP scripting capabilities. Each instance is connected to the main patch using 'send' and 'receive' objects. 'Send' and 'receive' objects are used as opposed to Max's patch-chords in order to have a less cluttered space-designing environment. Furthermore, each *one-point* abstraction is instantiated as a 'bpatcher'; this allows the user to see the abstraction's own interface and adjust parameters for that point.

The entire functionality of *space-master* is accessible through a set of OpenSoundControl messages [22]. The patch *space-master* understands a large number of OSC messages including read/write commands, clearing commands, turning interpolation on/off, turning 3-d rendering on/off, adding/removing/modifying data points and their kernels. The technique of using OSC as a communications scheme allows efficient management of multiple spaces within one application [23].

All data storage for *space-master* is done using Jitter matrices. This allows us to work with dimensionalities and numbers of points in the space that are larger than 256, Max's inherent limit on the number of items in a list. In addition to using Max's lists for entering data into the data-matrix, data can also be entered using the name of a Jitter matrix after the '/store n' message. Similarly, the weighted interpolated list produced by *space-master* is output as a 1-d, 1-plane Jitter matrix containing 32-bit floating points, again to avoid the 256-item limit of lists in Max. The matrix can be easily converted to a list using the 'jit.spill' object.

Jitter's OpenGL functionality brings the benefits of hardware-accelerated 3-d graphics and its amenities like the ability to freely rotate the 3-d objects or zoom in and out of the surface plots. To take advantage of this feature, the 3-d representation of the space rendered by *space-master* is produced using OpenGL.

## 4.2 APPLICATIONS

Numerous applications of the interpolation technique were developed using Max/MSP and Jitter. These applications, which are generally in the form of real-time performance instruments, aim to provide the composer/performer with high-level control of a process that functions in a high-dimensional space, that is, a process that has a large number of control parameters. The described system allowed us to use our musical intuition to define the spaces. This in turn made performing with these spaces intuitive and rewarding.

### *4.2.1 Drum Space: Timbre Space of Percussion Samples*

A simple application of this interpolation technique is a 2-d timbre space designed to interpolate among percussive sounds. The strong similarity in the amplitude envelopes for percussive hits contributes to very effective fusing between the sounds and thereby makes them ideal candidates for weighted-amplitude mixing. These samples, 38 low sounding membranophones ranging from tympanis and concert bass drums to a south Indian mirdangam and a classic Roland 808 kick drum, were laid out subjectively on the 2-d space to achieve the musical goal of a continuous space of low drum sounds that contains interesting mixtures (Figure 4).

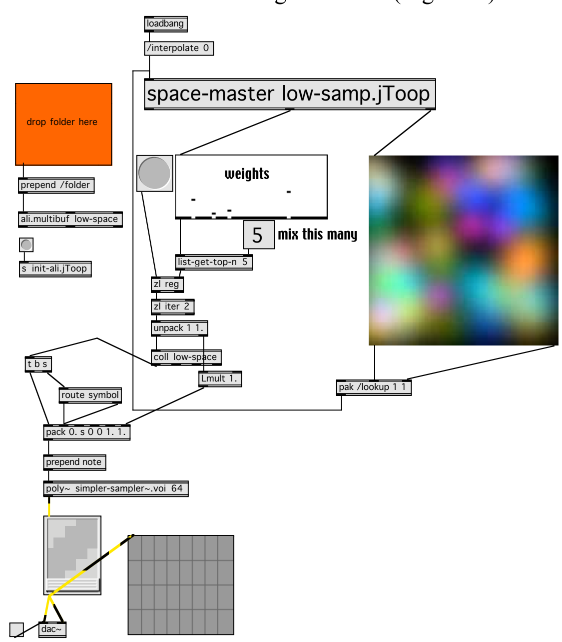



**Figure 4. Each colored region in the space represents an audio sample of a low drum. As the user navigates around the space, the list of normalized weights at that coordinate in the space is output from *space-master's* second output. Each time playback is triggered, the values in this list are used to weigh the amplitudes of the corresponding samples in the mixture.**

In this example, the list of normalized weights at a given coordinate in the space is the only required data for producing a weighted mix of the samples; in other words, the data matrix in this space is empty. List interpolation was therefore turned off to preserve computation power.

An additional parameter in the patch designates how many samples to mix at a time. Experimentation proved that it was computationally wasteful to mix any more than 5-7 samples at a time since the relative amplitudes for samples beyond the strongest 5-7 contributors were low enough to render them unnoticeable in the mix.

### 4.2.2 Res Space: Timbre Space Based on Transformations of Resonance Models

Models of resonance provide an efficient and highly mutable approach to sound synthesis. CNMAT's *resonators~* object for Max/MSP [4] supply an implementation of parallel banks of 2-pole resonators and a set of possible transformations for existing models. When provided with a frequency, an amplitude and a decay-rate for each frequency component in the resonance model, *resonators~* efficiently models a bank of resonating filters with the given parameters that can then be excited by an impulse, enveloped noise or other audio signals. The *res-transform* [4] object allows one to apply global transformations to a resonance model by way of a large number of scaling, adding, or component producing functions. Together they allow musicians to produce an extremely wide range of sounds from one set of data, usually derived from analysis of a recorded sound.

One obstacle to finding musically satisfying results with *resonators~* and *res-transform* is the complexity of the interdependencies between various transformation parameters. For instance, the 'spectral-slope' and 'spectral-corner' parameters to *res-transform* work together to redistribute energy to different parts of the frequency spectrum. Similarly 'cluster-size', 'frequency-around', 'attenuation-spread' and 'rate-spread' all contribute to the interesting timbres and intricate beating patterns that result when additional frequency components are created from ones already extant in the model. Furthermore, parameters like 'rate-scale' (which changes the decay rates of the frequency components) can significantly affect the perceptual loudness of the model.

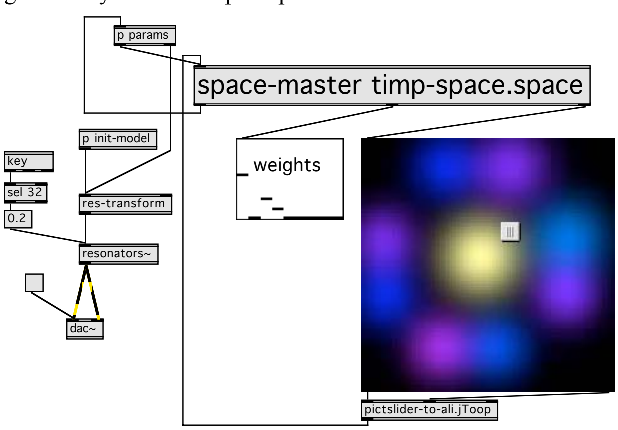



**Figure 5. The space in the figure above shows the resulting models as nine colored regions where the center represents the unaffected model; the models are arranged in pairs to convey the local similarities within the larger space. The two regions to the right, for instance, both scale the frequency of the model upwards, lower the decay-rates and adjust the gains to achieve uniform loudness but each slightly differently, whereas the regions in the bottom are two low sounding models made up 2-times clusters with differing beating patterns.**

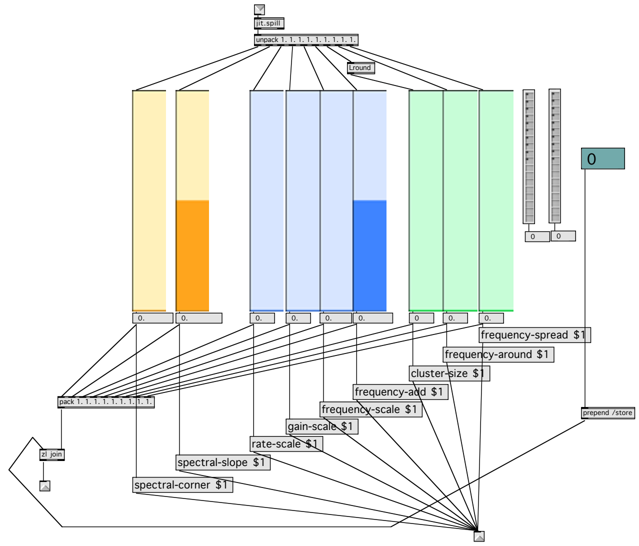



**Figure 6. The list of floating point number representing a weighted interpolation among sets of *res-transform* parameters is unpacked, formatted and sent to *res-transform* out of the right outlet.**

A timbre space was produced around the resonance model of a tympani and eight transformations of that model (Figure 5). In addition to the unaffected model consisting of 50 frequency, amplitude, decay-rate triplets, eight additional models were produced by carefully exploring what *res-transform* can do to the model; specifically, additional models were found by experimenting with the following eight parameters of *res-transform:* 'spectral-slope', 'spectral-corner', 'rate-scale', 'gain-scale', 'frequency-scale', 'frequency-add', 'cluster-size', 'frequency-around', 'frequency-spread'. Desirable sets of these *res-transform* parameters were then stored and interpolated among using *space-master* (Figure 6).

Two features of this approach to managing related models of resonance are noteworthy: first, interesting musical results are quite often found when exploring the space in between the known regions. In regard to live performance control, this provides not only a continuously mutable model for synthesis of percussive sounds, it also opens the way for countless new sonic possibilities derived from those designed by the composer. Second, the use of Gaussian kernels—as opposed to Euclidian distance for instance—is crucial in this application. Since many of the transformations apply frequency-scaling to the model, using weight function that are not adjustable by way of a parameter like standard deviation would result in a timbre space that is filled with glissandi. In our experience, continuously changing pitch adds a transparent synthetic quality to the sound which detracts from the efficacy of the instrument a drum-like entity. By using adjustable Gaussian kernels, however, these glissandi can be effectively localized to one region of the space.

It is worth noting that a timbre space of resonances could also be constructed by interpolating among the much larger lists of frequency/amplitude/decay-rate for each model (as opposed to interpolating among lists of *res-transform* parameters). The advantage of this approach is that one could use completely unrelated models as the points in the space. The disadvantages are: first, since the models are unrelated effectively localizing the aforementioned glissando problem and still getting a satisfying level of blending between the models becomes much more difficult; second, since a timbrally rich model often requires at least 30 frequency components (i.e. yielding 90 floating point numbers to represent each model), the interpolation among them becomes much more computationally intensive.

### *4.2.3 Reverb Space*

We step away now from the realm of synthesis and into that of processing. In the recent years it has become feasible to provide very high quality reverberation entirely in software. This is evident in numerous high quality reverberators now available in the form of VST plugins in use in many professional studios. The parameter space for these units, however, is often very high in dimensionality and therefore difficult to maneuver beyond simply switching from one preset to another. In another application of our dimensionality reduction technique, a space of reverb settings was constructed to control TrueVerb, room modeler developed by Waves (Figure 8).

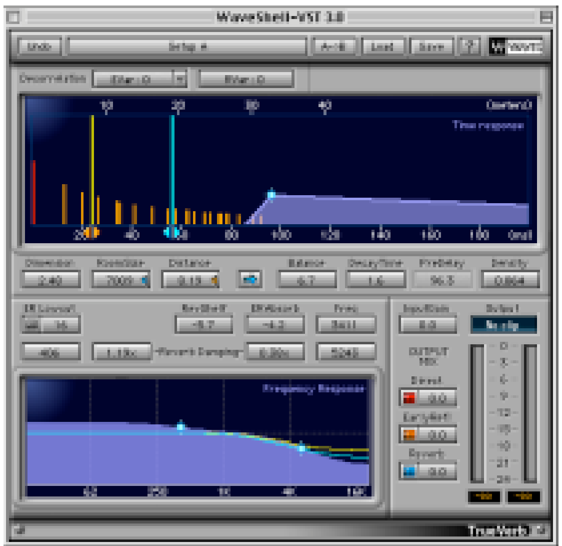



**Figure 8. The user interface for the VST plugin Waves TrueVerb Room Modeler. This plugin, like many other high quality processing units has a large number of finely adjustable parameters—too many to control independently in real-time.**

TrueVerb, which "combines two separate modules - an Early Reflections simulator, and a Reverb - to produce a high quality, natural-sounding room effect" [16] has a total of 45 parameters. As composers and performers interested in having high-level control over reverberation without having to manage 45 knobs, we developed a patch that contains a reverb space with six regions corresponding to six distinct settings of the TrueVerb module (Figure 9). Moving about in this 2-d reverb space shifts from the sound of a small hall, through that of a dampened practice room, and into a lush stadium. Changes occur gradually and smoothly, in a way that would be impossible to reproduce if the parameters were to be controlled individually in real-time.

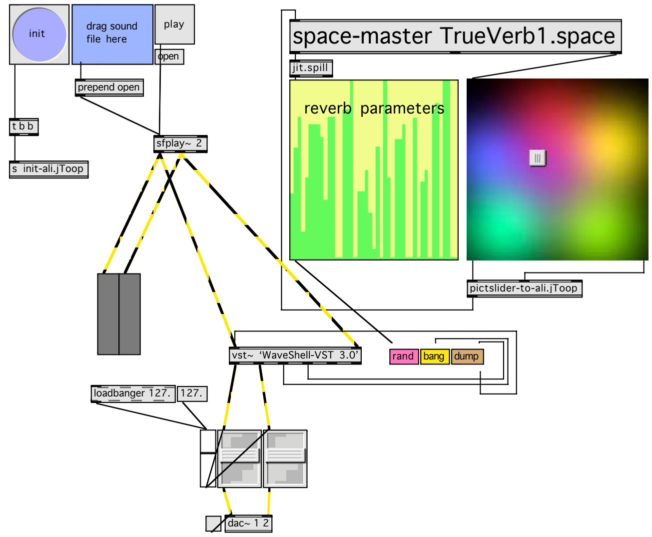



**Figure 9. Patch developed to interpolate among various reverb settings of the VST plugin TrueVerb by Waves.**

### *4.2.4 Grain Space: Amplitude Envelopes, Waveforms, Durations and Harmonic Content of Granular Clouds*

A real-time performance instrument was developed around the concept of granular clouds (Figure 10) [12]. The instrument is a generalized granular synthesizer that polyphonically produces grains of sound, that is enveloped waveforms with a given frequency, amplitude and durations. It uses probability tables to select the duration and pitch of each grain while the amplitude envelope and the waveform are results of *space-master* interpolations. Four separate *space-master* modules with the following contents were used:

1. Five amplitude envelope types
2. Nine 512-sample wave tables for the waveform
3. Five different probability tables for duration
4. Fourteen different probability tables for the pitch

Each time the instrument is triggered, it probabilistically chooses a duration between 10ms-3000ms, a frequency between 40HZ-5000HZ, and it uses the current interpolated waveform and amplitude envelope to play a grain. Each instance of this instrument uses a 64-voice polyphonic player to allow extremely dense clouds of grains whose control parameters are controlled in real-time using the spatial arrangement.

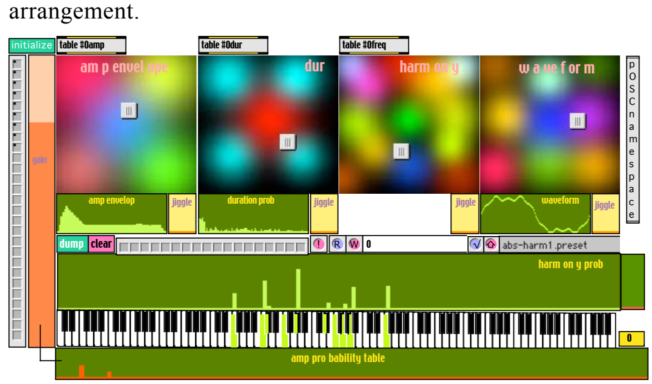


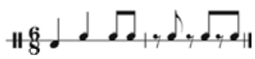

**Figure 10. The graphical interface for the Grain Space instrument. The four colored spaces in the top area of the patch correspond to amplitude envelope, probabilistic grain duration, probabilistic grain pitch, and the waveform in the grain. The smaller interface objects below each space (with the exception of the "harmony" space) show the current interpolated mix. The interface comprised of a keyboard**

**and set of sliders visualizes the current interpolated probability distribution for the grain's pitch. Note that an additional 'jiggle' parameter is provided for each space to allow for slight—or drastic—deviations in the parameters of the grain by jiggling the location in each interpolation space by a given percentage.**

In performance, it is possible to use multiple instances of this instrument too produce contrasting granular clouds with entirely different sets of parameters.

#### *4.2.5 Beat Space: High-level Control of Rhythmic Material Using Spatial Layouts*

A real-time instrument was developed to allow performance of probabilistic variations derived from a simple accent pattern. For an example of this instrument's usage we choose the omnipresent west African bell pattern:



We represent this rhythm as the duration vector

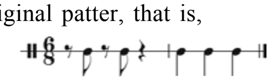
[2 2 1 2 2 2 1]

where the eighth note is the lowest duration-value, represented by the number 1 in the above vector. The patch then assigns probabilities to each beat in the pattern. Beats that contained hits in the original pattern are assigned high probabilities, while the beats in between are assigned low ones. Working with probability ranges between 0 and 1, the above pattern would be deterministically represented the probability vector

[1 0 1 0 1 1 0 1 0 1 0 1]

Note that there are a total of 12 probabilities, each corresponding to a beat in the pattern; the 1's represent eighth note slots where there was a hit and the 0's represent rests.

The patch then creates a perceptual space comprised of 5 regions (Figure 11). The region in the center represents the original pattern by way of the above deterministic beat-wise probability vector. The region to the right represents the opposite, a probability vector emphasizing all of the beats missing in the original patter, that is,



which is represented by the beat-wise probability vector:

[0 1 0 1 0 0 1 0 1 0 1 0]

The region at the top of the space is assigned the densest possible pattern made of the minimal pulse value (every eighth notes's probability equal to 1) and the region to the bottom the opposite (every eighth note's probability equal to 0). The region to the left is initially assigned the same probability values as the original pattern, but is reserved for user-defined probability. The user can draw a new probability vector and click the button to the left of the probability display to store the probability vector in the region in the left of the space. By navigating within this space, the user is able to produce rhythmic patterns that range in level of syncopation in relation to the original pattern (from center to right of space) and overall density (the vertical axis of the space).

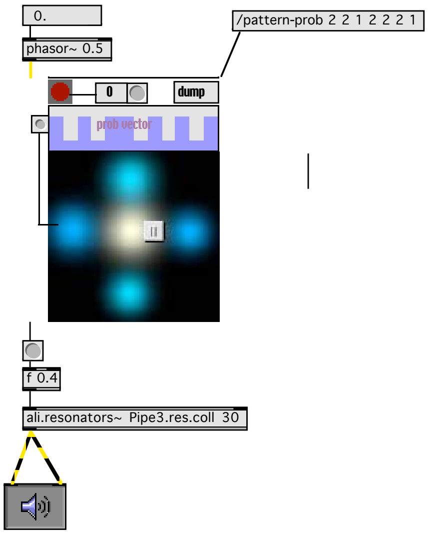



**Figure 11. The interface for the Beat Space instrument. The perceptual space is made up of 5 regions whose associated data is calculated by the patch each time a new rhythmic pattern is entered into the patch.**

#### *4.2.6 Boids Space: Perceptual Space for the Parameters of a Bird Flocking Algorithm*

A real-time performance instrument was developed that utilized Eric Singer's Max/MSP implementation [15] of Craig Reynolds's famous bird flocking algorithm [11]. This Max/MSP externals, named Boids, models the flight path of a designated number of birds with regard to a set of 17 parameters (e.g. centering tendency, maximum speed, inertia, repelling tendency, etc.). Different settings of these parameters give very particular results in the overall shape and movement of the flock (Figure 12).

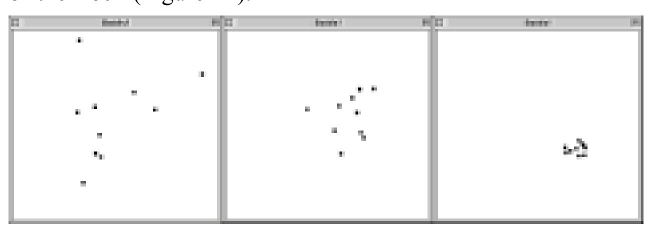


**Figure 12. This figure shows three different arrangements of the parameters to the Boids bird flocking algorithm. Each black dot represents the location of a "boid" in the flock's flight area. As parameters for the algorithm are changed, the flight tendencies of individual birds change, thereby changing the overall shape and behavior of the entire flock.**

In the Boids Space instrument, designed for live interaction with a MIDI piano, each bird was represented in sound by one voice of a resonators~-based synthesizer. The vertical position of the bird was mapped to quantized tempo, and the horizontal position to pitch register. Pitch material was extracted in real-time from the MIDI piano, and applied to the current pitch register of each bird. The performer using Boids Space controls the attraction point for the flock of birds, as well as the overall flocking tendencies by way of a perceptual space made up of seven predefined sets of Boids' parameters (Figure 13). He also controls the overall amplitude and resonance for the synthesizers, The overall effect was a tremendous amount high-level of control over number of independent voices. By navigating the space for the Boids algorithm's parameters, it was possible to find interesting regions of varying rhythmic and registral correspondence.

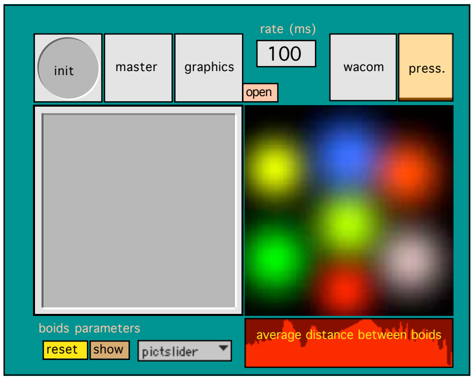



**Figure 13. The interface for the Boids Space Instrument. The 2 dimensional slider on the left controls the attraction point for the flock of birds, the colored space on the right allows the user to interpolate among seven different sets of parameters to the Boids algorithm. Average distance between the Boids was calculated and mapped to reverberation wetness.**

### *4.2.7 SPACE Space*

Finally, we introduce the notion of creating spaces of spaces. As conveyed by a number of the previous examples, sophisticated applications of geometric arrangements in performance environments can involve multiple perceptual spaces in one instrument. The same method of dimensionality reduction that is applied to each individual perceptual space can also be applied to the entire system in order to organize specific arrangements of the individual components. Specifically, a space of spaces can be created by using a mother-space to interpolate between sets of coordinates in the daughter spaces (Figure 14). For example, in order to control four daughter-spaces with one mother-space, one would store 8-membered lists that contains desirable x-y coordinates for each daughter-space in the mother-space and interpolate between these lists. The result is an even higher level of control over a system with a very high number of parameters, by way of navigation in a space that has only two dimensions.

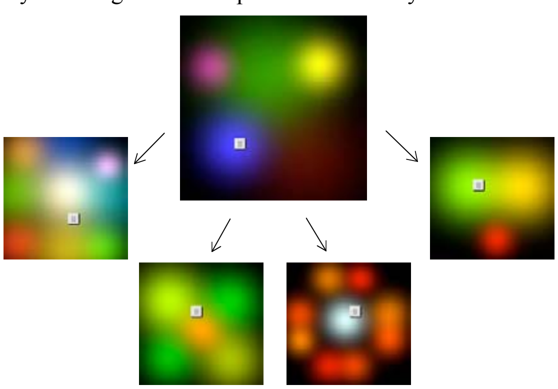


**Figure 14. This figure depicts the notion of creating spaces of spaces. The larger space in the center allows one to interpolate among sets of coordinates in the 4 smaller spaces below. Each of the smaller spaces subsequently controls the parameters to a different musical process.**

## 5. CONTROLLERS

Use of gestural controllers for real-time performance is an ongoing focus of research at CNMAT as well as in the interactive computer music community in general. At CNMAT we have developed a great deal of software for controllers like the Buchla Thunder, Saitek Cyborg 3D Joysticks and Wacom drawing tablets. The reduction of high-dimensional parameter spaces down to 2 dimensions has proved to be an extremely effective technique for controlling real-time computer instruments at a high level. The paradigm is simple: gestural controllers translate a performer's physical gestures in space into streams of data for controlling musical processes. Although some controllers can output more than 3 unique streams of data as a result of a single physical gesture, they most often result in 1 to 3 dimensions of control data. In order to allow intimate control over a sophisticated musical process, it is often necessary to control many more parameters than a simple one-to-one mapping of a controller allows. Let us take the task of controlling reverb by way of a Wacom drawing tablet as an example. The Wacom interface serves as a good example because it in fact outputs five continuous streams of data from a single gesture with the pen (horizontal and vertical position, 2 dimensions of tilt, and pressure). However, even with such rich output from the controller, a one to one mapping of its five dimensions to a high quality reverberator like Waves TrueVerb would give the performer very little control, for the reverb unit has not five but 45 parameters. It is also worth mentioning that with the Wacom tablet, as is the case with many other controllers that output more than 3 continuous streams from the same gesture, it can be exceedingly difficult to reliably reproduce a gesture in performance. On the other hand, the horizontal and vertical positions alone can be mapped to a perceptual reverb space like the one described earlier, thereby giving the performer full control over this process with maximal accuracy and reliability in the gesture-to-control mapping. Furthermore, visual feedback of the process under control by the performer becomes a much simpler problem when the controllers physical existence in 3 dimensions is directly correlated with an intuitively laid out perceptual space that functions in 2 dimensions.

## 6. FUTURE WORK

Besides building more real-time instruments that make use of graphical layouts of high-dimension spaces, we intend to experiment with performing transformations on the spaces in real-time. That is, by moving points around in the space or by transforming their kernels, one can find entirely different sets of interpolated results from the same data set. These transformations would not only expand the palette of musical capabilities that an instrument has, they could also elucidate structural similarities and dissimilarities in the data that may not have been evident in the original spatial layout.

We also plan to work more with purely numerical techniques for creating spaces. The technique of Locally Linear Embedding as demonstrated by Sam T. Roweis and Lawrence K. Saul's work [13] seems extremely promising in its applications to our work with real-time instruments and our desire to organize and perform with very large sets of data.

## 7. ACKNOWLEDGMENTS

Our thanks to Joshua Kit Clayton, Adrian Freed, Eric Singer, Matt Wright, and the Max/MSP/Jitter community for the help with the development. We also thank David Bithell, Ben Jacobs and Roberto Morales, for their most insightful musical interactions with our instruments.

## 8. REFERENCES


[1] Babbitt, M. (1965) The use of computers in musicological reserach. Perpectives of New Music, 3 (2).

[2] Goldstone, R.L. (1994) An efficient method for obtaining similarity data. Behavior Research Methods, Instruments, & Computers, 26, 381-386.

[3] Grey, J.M. (1977) Multidimensional perceptual scaling of musical timbres. Journal of Acoustical Society of America, 61, 1270-1277.

[4] Jehan, T., Freed, A. and Dudas, R., Musical Applications of New Filter Extensions to Max/MSP. in International Computer Music Conference, (Beijing, China, 1999), ICMA, 504-507.

[5] Krumhansl, C.L. Cognitive Foundations of Musical Pitch. Oxford University Press, Oxford, 1990.

[6] Krumhansl, C.L. Why is musical timbre so hard to understand? in Nielzén, S. and Olsson, O. eds. Structure and perception of electroacoustic sound and music, Elsevier, Amsterdam, 1989, 43-53.

[7] Lerdahl, F. Tonal Pitch Space. Oxford University Press, 2001.

[8] McAdams, S. and Cunibile, J.C. (1992) Perception of timbral analogies. Philosophical Transactions of the Royal Society, London (Series B 336), 383-389.

[9] Plomp, R. Timbre as multidimensional attribute of complex tones. in Plomp, R. and Smoorenburg, R.G. eds. Frequency analysis and preiodicity detection in hearing, Suithoff, Leiden, 1966.

[10] Plomp, R. and Steeneken, J.M. (1969) Effect of phase on the timbre of complex tones. Jounal of the Accoustical Society of America, 46, 377-380.

[11] Reynolds, C.W., Flocks, Herds and Schools: A Distributed Behavioral Model, in Computer Graphics. in SIGGRAPH, (1987), 25-34.

[12] Roads, C. Microsound. MIT Press, Cambridge, Ma, 2001.

[13] Roweis, S.T.S., Lawrence K., Locally Linear Embedding. 2003, (http://www.cs.toronto.edu/~roweis/lle/).

[14] Shepard, R.N. Structural Representations of Musical Pitch. in Deutsch, D. ed. Psychology of Music, Academic Press, New York, 1982, 343-390.

[15] Singer, E., Boids. 1998, (http://www.ericsinger.com).

[16] Waves, TrueVerb Room Emulator. 2003, (http://www.waves.com/htmls/prods/indi/tv.html).

[17] Wedin, L. and Goude, G. (1972) Dimension analysis and the perception of instrumental timbre. Scandinavian Journal of Psychology, 13 (3), 228-240.

[18] Wessel, D., Bristow, D. and Settel, Z., Control of Phrasing and Articulation in Synthesis. in International Computer Music Conference, (Champaign, Urbana, USA, 1987), International Computer Music Association, 108-116.

[19] Wessel, D. and Wright, M. (2002) Problems and Prospects for Intimate Musical Control of Computers. Computer Music Journal, 26 (3), 11-22.

[20] Wessel, D.L. (1973) Psychoacoustics and music: A report from Michigan State University. Bulletin of the Computer Arts Society,, 1 (30).

[21] Wessel, D.L. (1979) Timbre space as a musical control structure. Computer Music Journal, 3 (2), 45-52.

[22] Wright, M. and Freed, A., Open Sound Control: A New Protocol for Communicating with Sound Synthesizers. in International Computer Music Conference, (Thessaloniki, Hellas, 1997), International Computer Music Association, 101-104.

[23] Wright, M., Freed, A., Lee, A., Madden, T. and Momeni, A., Managing Complexity with Explicit Mapping of Gestures to Sound Control with OSC. in International Computer Music Conference, (Habana, Cuba, 2001), 314-317.

# Composing for the (dis)Embodied Ensemble: Notational Systems in *(dis)Appearances*

Matthew Burtner
University of Virginia, McIntire Department of Music
Virginia Center for Computer Music (VCCM)
Charlottesville, Virginia, USA 22903
mburtner@virginia.edu

## ABSTRACT
This paper explores compositional and notational approaches for working with controllers. The notational systems devised for the composition *(dis)Appearances* are discussed in depth in an attempt to formulate a new approach to composition using ensembles that navigates a performative space between reality and virtuality.

## Keywords
Composition, notation systems, virtual reality, controllers, physical modeling, string, violin.

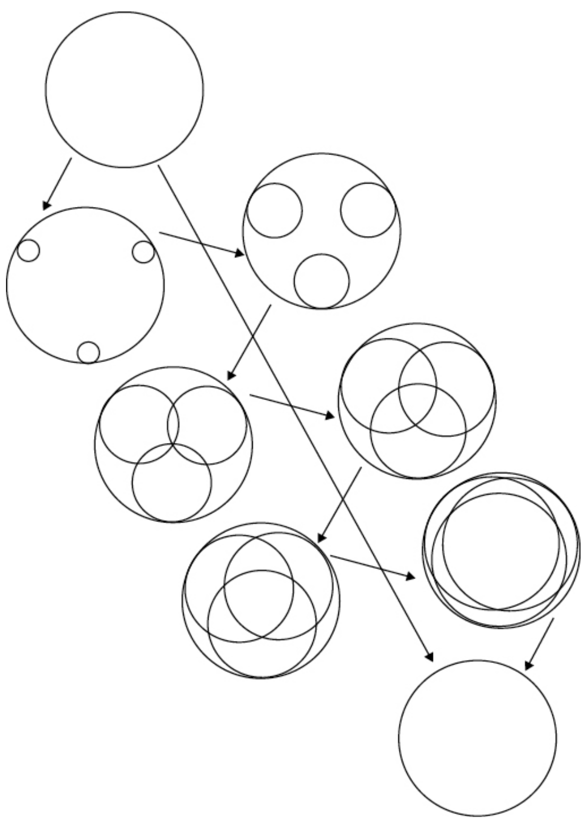


**Figure 1: Musical form of *(dis)Appearances* showing the generation, emergence and disappearance of the independent instrumental elements.**

## 1. BACKGROUND

### 1.1 Notations for NIME Y1.9K to Y2K

New interfaces for musical expression (NIME) and new approaches to sound synthesis provide composers opportunities to draw from an ever richer orchestra of expressive instruments and tools (Cook 2002, Paradiso, Hunt 1999, Wanderley 2001). In recent years, the renewed interest in the area of musical controllers reflects the origins of electronic music in the early 20th Century when pioneers such as Cahill, Theremin, Trautwein, Martenot and Hammond carved out the beginning of electronic music as a performance-based field (Roads 1996, Holmes 2002). But these brave-new-century interfaces left little trace in the way of compositions or even notational systems. Recordings or scores of original music for these instruments would be valuable because it would point to key aspects of how artists perceived the expressivity of the new instruments.

In today's brave-new-century there is a strong tradition of electronic and experimental music composition, embracing a wide range of styles and forms. However, we see little research into designing systems to describe repeatable sequences of control change, ie. notation. Composers are often performers of their own inventions, or a demonstration-improvisation is created for a new instrument showing the facilities of the controller more than exploring any real musical depth the instrument may possess.

A work such as Stockhausen's *Mikrophonie I* from 1965, for a large tam tam, microphones and mixer remains a part of the performed classical music canon precisely because the score describes a sequence of control changes over time and therefore it *can* be performed (Burns 2001). The notational system was invented by Stockhausen, and if it did not exist, our understanding of the piece and certainly of the rich expressive potential of the extended tam-tam would be impoverished. (Figure 2) .

While a tam tam is not a NIME per-se, Stockhausen's extended exploration of the instrument combined with the unique network interaction between the ensemble provides a good metaphor for the type of relationship we see between instrument, synthesis, performance and composition in the field of computer music. Additionally, the approach to the interface is itself an excellent example of how recombining control parameters in performance and composition can redefine and augment traditional interfaces.

**Figure 2: Original performance of Mikrophonie 1, 1965 in Cologne. Photo by Klaus Barisch**

## 1.2 Formative Work

### *1.2.1 Noisegate 67 and the Metasaxophone*

This paper stems from a more general interest in investigating notational systems for new musical interfaces. Frequent performances with the Metasaxophone (Burtner 2002) and a desire to create repeatable sonic states with that controller inspired the development of a notation for *Noisegate 67* (1999). The notation system for that piece has been discussed in detail previously and presented to the NIME community (Burtner 2002).

### *1.2.2 MinMax and the Scanned Synthesis System*

Working with Max Mathews on his Radio Baton Scanned Synthesis system led to another notational approach for controllers for the composition *MinMax* (2000). In this notation, the performer is given detailed information about aspects of the system such as time in seconds, display feedback from the Scanned Synthesis window display on the computer monitor (visual feedback cues), movement of the two batons across the baton antenna surface, movement in the Z plane, other controller aspects of the Radio Baton such as the potentiometer settings, sounding pitches (audio feedback), and programming instructions for setting up the synthesis algorithms such as timbre, pitch system settings, hammer position and hammer force, hammer spacing, string tension and mass centering.

These notational constructs are totally idiosyncratic to the Max Mathews Scanned Synthesis System. But a performer given that system can recreate *MinMax* from the score. The score was made because Mathews was traveling to ICMC 2000 in Berlin to present scanned synthesis with Bill Verplank (Mathews, Verplank, Shaw 2001). As part of the demo Max had requested a short piece that he then could play as a first example of compositional uses of the synthesis technique. The score was made so that Mathews could present the piece without the presence of the composer. Figure 3 shows a page from the score of *MinMax*.

### *1.2.3 S-Trance-S and S-Morphe-S: Morphological Instruments*

The potential for morphological instruments arises when control and synthesis instrument parameters become separated (Chadabe 2002). In a project with Stefania Serafin, the acoustics and artistic possibilities of this disassociation has been explored (Burtner and Serafin 2000, 2001). The compositional outcome of this research project is expressed musically in the compositions *S-Trance-S* (2001) for a bowed string tenor saxopohone, and *S-Morphe-S* (2002) for a soprano saxophone singing bowl.

In *S-Trance-S*, the Metasaxophone was used as a controller for bowed string physical models. By controlling the string from within the gestural space of a wind instrument, new expressive potentialities of the model are opened. The disembodied nature of physical models becomes a means of recombining it with other interfaces, creating extended techniques for physical models that would not be possible for the real instrument. This piece has been discussed in detail in a previous article (Burtner and Serafin 2002).

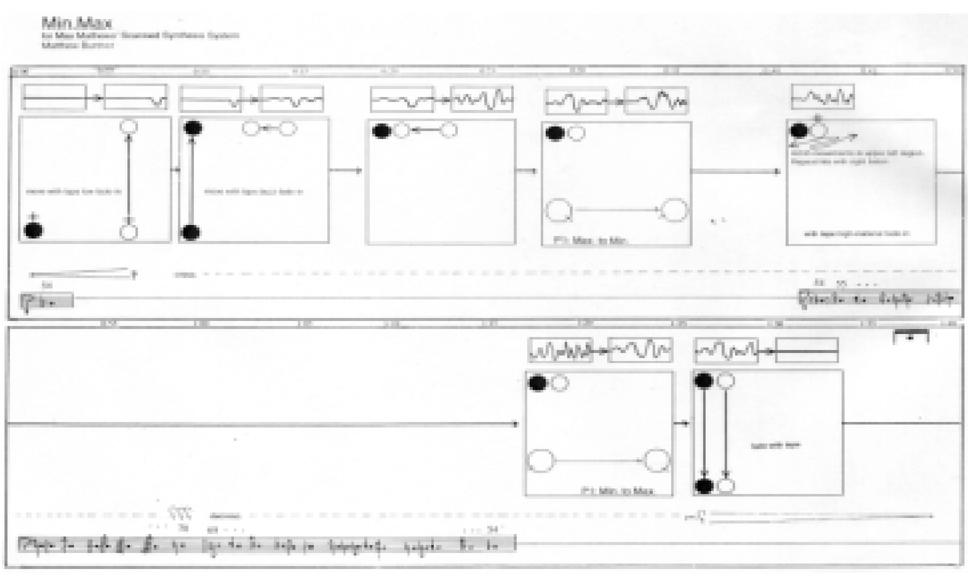


**Figure 3: Score of *MinMax* for Max Mathews' Scanned Synthesis System**

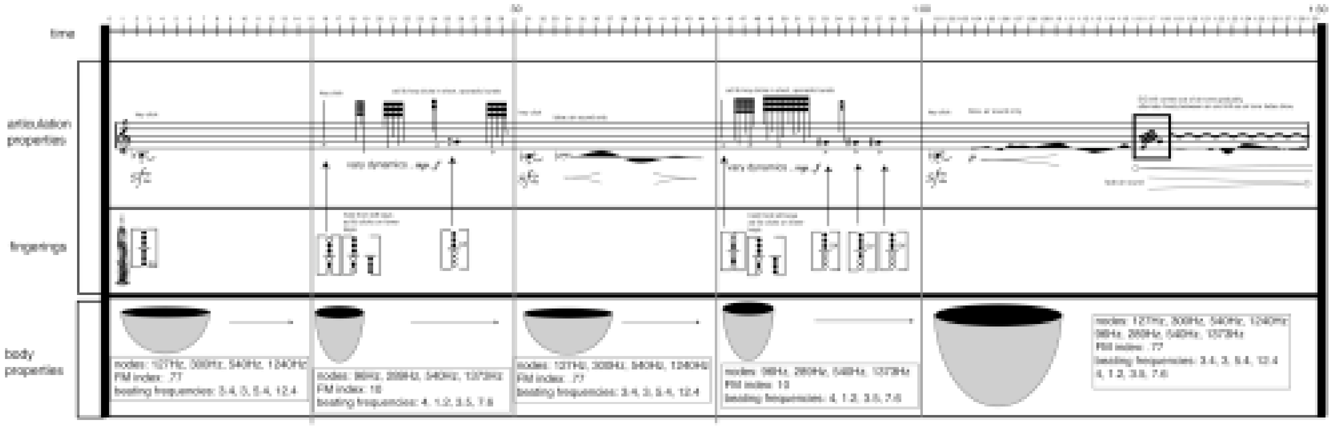



**Figure 4: *S-Morphe-S* soprano saxophone performance interface**

In *S-Morphe-S*, a real soprano saxophone is reembodied within a virtual bowl by sending the saxophone through a physically modeled bowl as an impulse to the model. The result is a hybrid instrument with the articulatory characteristics of a soprano saxophone but the body of a singing bowl. The saxophone uses varied articulations such as key clicks, breath, trills and sustained tones. The shape and material properties of the bowl are varied in real time creating a continuously transforming body. Figure 4 shows a page from the performance score of *S-Morphe-S*.

The working paradigm of these pieces investigates virtual reality by placing a performed physical instrument outside the realm of physical reality. In live performance this is compelling because the audience perceives something that should be impossible happening in real time.

The titles of these compositions reflect the philosophy behind them. *S-Trance-S* refers to the series of dream or hallucinations represented by the different morphological forms generated as the energy of the controller is transfused into the medium of sound. These hybrid forms then act as the extensions of their archetypes, exploring states of metamorphoses. The title *S-Morphe-S* comes from the Greek word for *form*, and in Greek mythology Morpheus was the god of sleep, of disembodied forms. The english word commonly used for a transformation between two objects is morph, a shortening of metamorphosis, derived from the Greek. The title of this piece is meant to evoke all of these meanings -- dreamed images, transformative bodies, and disembodied forms.

It occurred that this process of transformative reality based on the combination of embodied and disembodied instruments could be explored further and that interesting performative states between reality and virtual reality could be navigated musically.

### *1.2.4 Somata/Asomata and a Concentration on the String*

In recent years the string has been a focus of much development. The violin controller technology (Nichols 2001, Young 2002, Trueman and Cook 1999) and the research on physical models for strings (Serafin and Smith) are both at a sophisticated state and continue to grow. This allows for interesting compositional opportunities for combining synthesized strings and string controllers in different ways.

*Somata/Asomata* (2002) for electric string quartet and computer string quartet explores this approach to the hybrid string ensemble. In this work the computer is used to separate the sounding instrument from the instrumental controller. In this way, physical properties are remapped in different ways

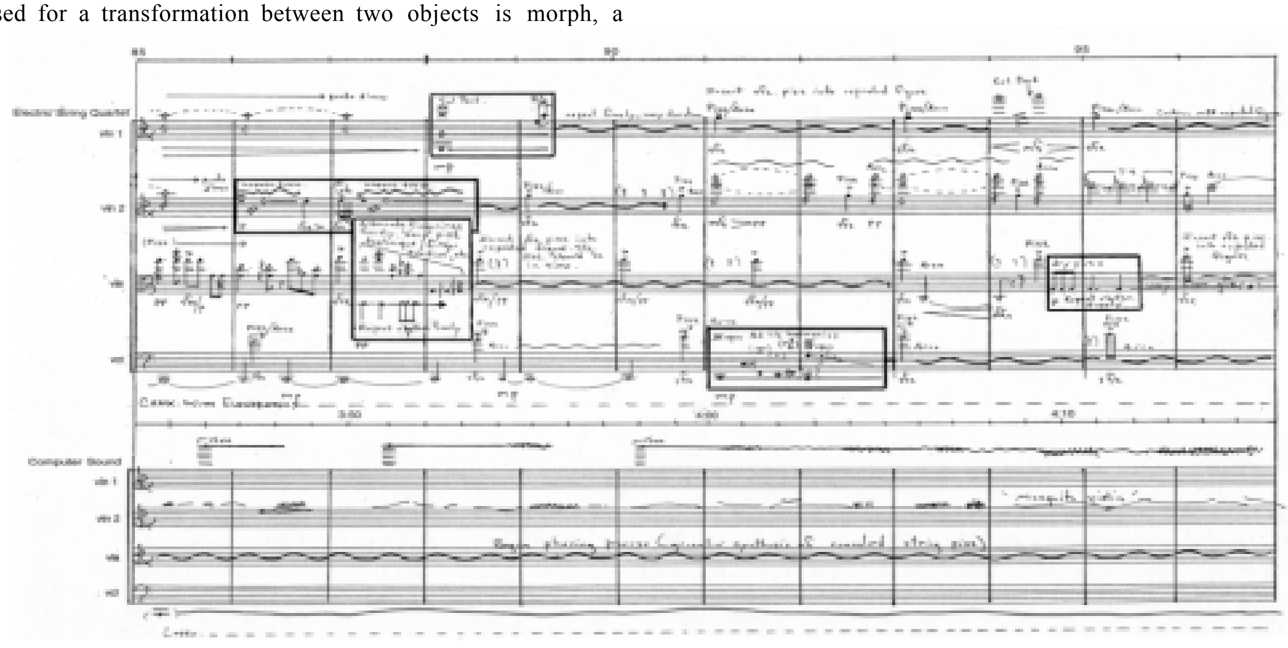



**Figure 5: Measures 85-97 of *Somata/Asomata* for electric string quartet and computer quartet**

to create a hybrid set of virtual instruments having selective properties of entirely different instruments. Notions of body and control are then explored through the recombinatory properties of these elements.

The concept of instrumental reality is explored through the cross-fertilization of acoustic and electroacoustic instruments in the four "digital prints" of the string quartet presented in the electronics. The violin prints are computer-generated physical model strings, controlled with extended, non-string controllers. The viola print is a computer processed acoustic string sample, predominated by convolved, granulated and phase shifted pizzicato sounds. The cello print is an unprocessed recorded string, the sound made using extended techniques such as bowing on the bridge, overbowing, and multiphonics.

## 2. *(DIS)APPEARANCES*: A (DIS)EMBODIED TRIO

*(dis)Appearances* (2003), a musical composition for a trio of amplified acoustic violin, electric violin, and computer violin/multicontroller, explores the nature of disembodiment and physical acoustic reality through the use of computer controllers and physical modeling synthesis. The piece is scored for a string trio in which the ensemble is not defined by register (as with a traditional string trio) but by states of embodiment/disembodiment.

### 2.1 Overview of the Instrumentation

An acoustic violin controlled with a real bow substantiates a basis in resonating real-world acoustics. An electric violin controlled with a bow outfitted with sensors that also acts as a real time controller for audio processing of the electric signal, mediates between the physical body as resonating space and the nonphysical computer-generated reality. The electric violin presents the physical presence of a controlled violin that is in fact an electric instrument using human-computer interface technology. Finally, the physical model violin, controlled by a multicontroller interface, presents a completely virtual, modeled, then extended violin. This trio mediates a space between embodiment and disembodiment as illustrated in the example below.

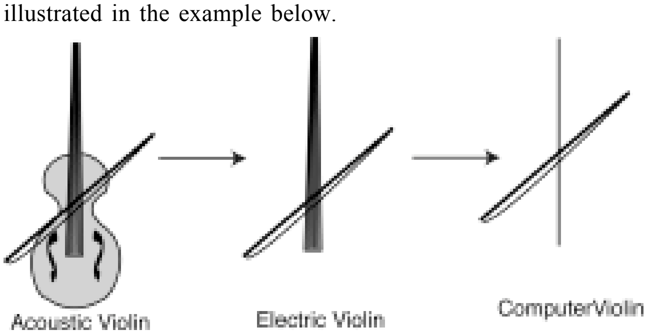



**Figure 6: Instrumentation disembodiment in *(dis)Appearances***

The musical form of *(dis)Appearances*, illustrated in the graphic in Figure 1, is derived from a musical idea based on the idea of appearance and disappearance.

Figure 7 illustrates the technical configuration of the piece. Each instrument's sound comes from a separate speaker located near the performer. In *(dis)Appearances* the electric violin bow is used to control signal processing of a hard-body electric violin instrument, creating a complexly variable electric instrument. The computer violin instrument, a physically modeled and extended violin controlled by a modular multi-controller system with a Peavey PC1600x multi-slider controller at its core.

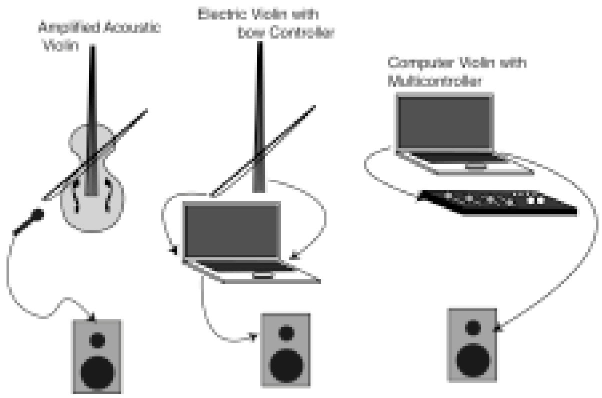



**Figure 7:Controller configuration in *(dis)Appearances***

### 2.2 Formal Overview

The form is a transformation of identity, simultaneously in and out of reality as illustrated in Figure 1. Each instrument develops a characteristic identity that both grows and disappears simultaneously. The identities interrelate, replacing and feeding off of one another. The acoustic violin anchors the form of the piece which is divided into 25 expanding pulses. The pulses are articulated by a novel technique of the performner holding the violin to her/his face and blowing across the *F* holes of the instrument. The electric violin identity is formed of natural harmonics, processed and pushed towards breath or noise. The computer violin/multicontroller identity is a machine-like glissando and buzzing that becomes increasingly unstable and multilayered, vanishing out of the range of hearing.

As the formal graph in Figure 1 illustrates, the three elements grow and overlap until the overlapping is complete and they have eventually occupied the same musical space.

## 3. NOTATION

The following sections present aspects of the notation for each of the three instruments. The notational approaches are in some manner quite different reflecting the degree of difference between the controllers.

### 3.1 The Acoustic Violin

The movements across the violin strings are notated in the score using the graphic notation shown in Figure 8. The representation shows the four strings, the bridge, the fingerboard, and the pegs.

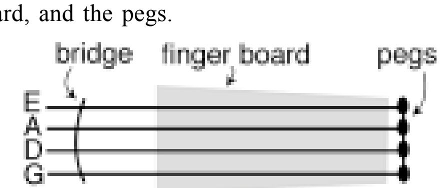



**Figure 8: graphic representations of the bowed violin**

The performer can orient the bowing action according to instructions such as shown in Figure 9. The left side of the figure shows the manner in which a normal down-bow would be scored. The right gesture shows a bowing action starting on the G string near the bridge and moving vertically across the strings to the A string while moving horizontally from the bowing area to a position near the top of the finger board. The curve of the line reveals a simultaneous gradual slowing of this movement.

**Figure 9: left: normal down bow, right: altered bowing motion**

Bowing types described in the score include up bow/down bow, bow pressure changes, and bow speed changes from stopped bowing to very fast bow speed. The notation of bowing in this manner was influenced by working with the physical model violin in which every paramater of the physically-based instrument needs to be accounted for and carefully structured in a controller mapping.

Pitch and dynamics are notated in a traditional fashion on their own staff.

In addition to being bowed, the acoustic violin is blown by the performer. Inspired by the ability to apply different types of impulses to physical models such as blowing the physical model string or bowl, the acoustic violin is also articulated here by blowing. The blown acoustic violin is a key aspect of the piece because the form is generated from 25 blown breaths, each one augmented by X (+1, 2, 3…25).

The blowing is accomplished by holding the violin to the mouth sideways and blowing across one of the "F" holes. The violin can be both blown and bowed simultaneously if held correctly. The performer blows across the hole with an "f", "h", or "sh" sound to fit with the timbre of the bowing. Blowing pressure is shown graphically on its own staff.

Figure 10 shows a single system of the acoustic violin score. In the example, a down bow motion moves from the G string gradually across the strings to the E string while simultaneously moving up the finger board horizontally. The bowing movement is slow, and becomes slower. The pressure increases for the first part then decreases to very light pressure. The left hand does not touch the strings. The dynamics of the gesture crescendo and decrescendo. Simultaneously the performer blows across the F hole.

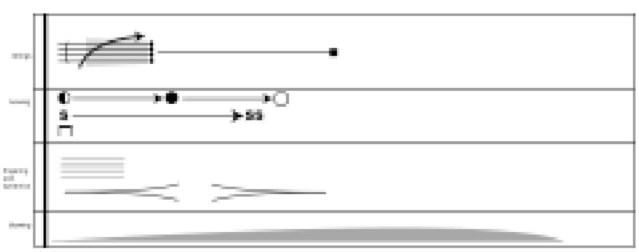


**Figure 10: Acoustic violin score example**

## 3.2 The Electric Violin

The hard body electric violin part is made entirely of harmonic nodes on the open strings. The score has four systems: density, articulations, fingering/dynamics/rhythmic figuration, and signal processing instructions.

The density staff shows the approximate density of the articulated harmonics. The vertical axis shows pitch bandwidth (high or low nodes). Vertical size indicates dynamics and horizontal size indicates duration. The performer follows the overall movement of harmonic grains but can be very free with the actual interpretation.

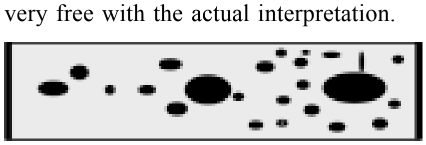


**Figure 11: Density staff**

The types of articulations the performer can use are given in the articulation staff. The performer freely alternates between articulations appearing in the staff over the given duration. The types of articulation described are tenuto, accent, detached tenuto, stacatto and jete.

The available types of articulation are combined together in a box over a particular duration. Global variables such as bow pressure and vibrato are added as modifiers of the articulation type. Figure 18 shows an example of alternating articulation types with increasing bow pressure over a defined time.

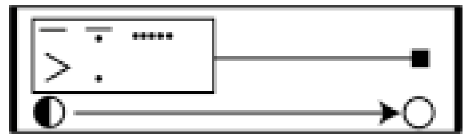


**Figure 12: Articulation staff**

Harmonic node fingerings are notated on a traditional pitch staff. The performer can play any of the notated harmonics in any order or the type of motion is described. Dynamics are also notated in a traditional manner, below the pitch staff.

The signal processing control staff gives instructions to the performer about tpes of signal processing applied to the amplified signal.

## 3.3 The Computer Violin/Multicontroller

The exbow computer violin is played with the modular multicontroller. In composing the piece, a PC1600X slider box was used. This controller offers enough continuous control parameters to dynamically alter the numerious physical model violin parameters.

The exbow interface (Figure 13) is programmed in Max/MSP. The score provides a single staff for all parameters of control with each control variable being assigned a number.

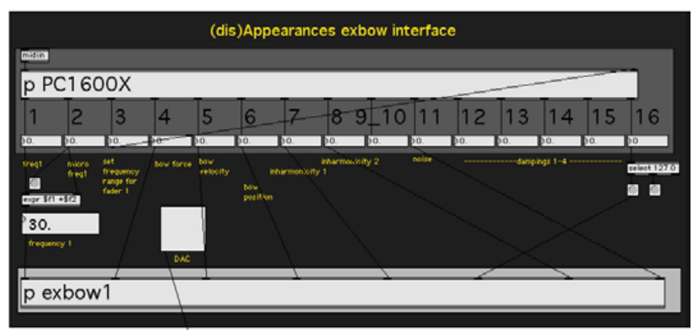



**Figure 13: Exbow performance interface**

In the PC1600X configuration, each number represents a slider from left to right. The top of the staff represents the slider-at-top position, and the bottom represents slider-at-bottom. The control parameters of the Exbow are:

1) Frequency: This determines the total frequency range of the model, set by slider 3, Frequency Range. The frequency range defaults to 5000Hz.

2) Micro-Frequency: This is a 15 Hz plus or minus deviation from the frequency. The performer can bend the pitch using this slider.

3) Frequency Range: The range of slider 1 is set between 5000 and 15 Hz. All the way up is 5000Hz.

4) Bow Force: Extreme force is at the top and no bow force is the bottom.

5) Bow Position: sul ponticello is the top and sul tasto is the bottom

6-7) Inharmonicity 1 and 2: Inharmonicity is increased by moving the sliders up.

8) Noise: Noise is introduced by moving the slider up.

The computer violin notation shows a control parameter only when it is being set or changed. In Figure 14 the initial position settings for sliders 1, 2, 3, 4 and 5 are given. Slider 5 is moving at the beginning of the measure, and slider 6 starts moving down halfway through that motion. The performer resolves the need for specific sliders to be controlled with the left or right hand.

Because of the potential for overwhelming the performer with performance data, it is important to understand that positioning of the sliders is approximate. The performer will need to use her/his ear to tune the specific setting to a desireable sound that fits into the context of the sounding music.

The detached nature of the multislider control interface was appealing when working with the extended physical model string because specific parameters could be isolated and set against very slow moving parameters. For example, the slightest change of bow pressure can become a compositional parameter because of the ability to isolate the micro-timbral effects of this otherwise dynamic property.

The change of parameters notated in the computer violin part would be impossible to play on an acoustic violin. Parameter configurations that would not make a sound on an acoustic string create interesting virtual acoustic states on the modeled string. For example, by maintaining all parameters in a steady state and only changing bow force, very interesting sounds can be obtained. On a real string it is very hard to isolate bow force from bow speed because any increase in pressure coincides with a change in bow velocity which the performer imediately tries to compensate for. Combining this type of parameter isolation with the possibility of dynamically changing the inharmonic properties of the string generates entirely new possibilities for composing music for strings.

The Graphic overview and Dynamics Staff provides an approximation of the overall changing sound.

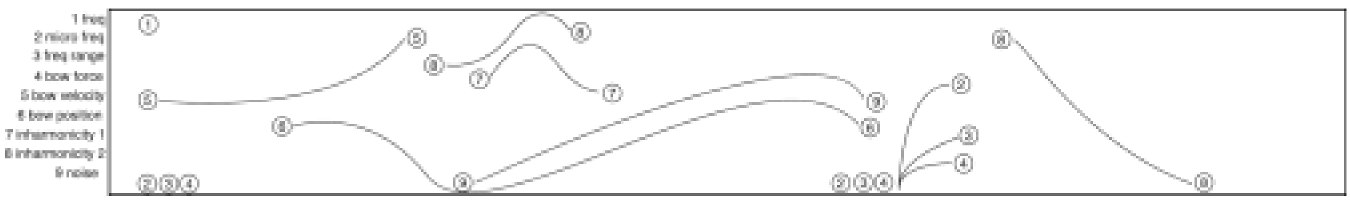



**Figure 14: Computer violin score example**

## 4. THE SCORE

Figure 15 shows a page from the score of *(dis)Appearances*. The notational elements described above have been combined into a single system, now forming an orchestrated musical ensemble.

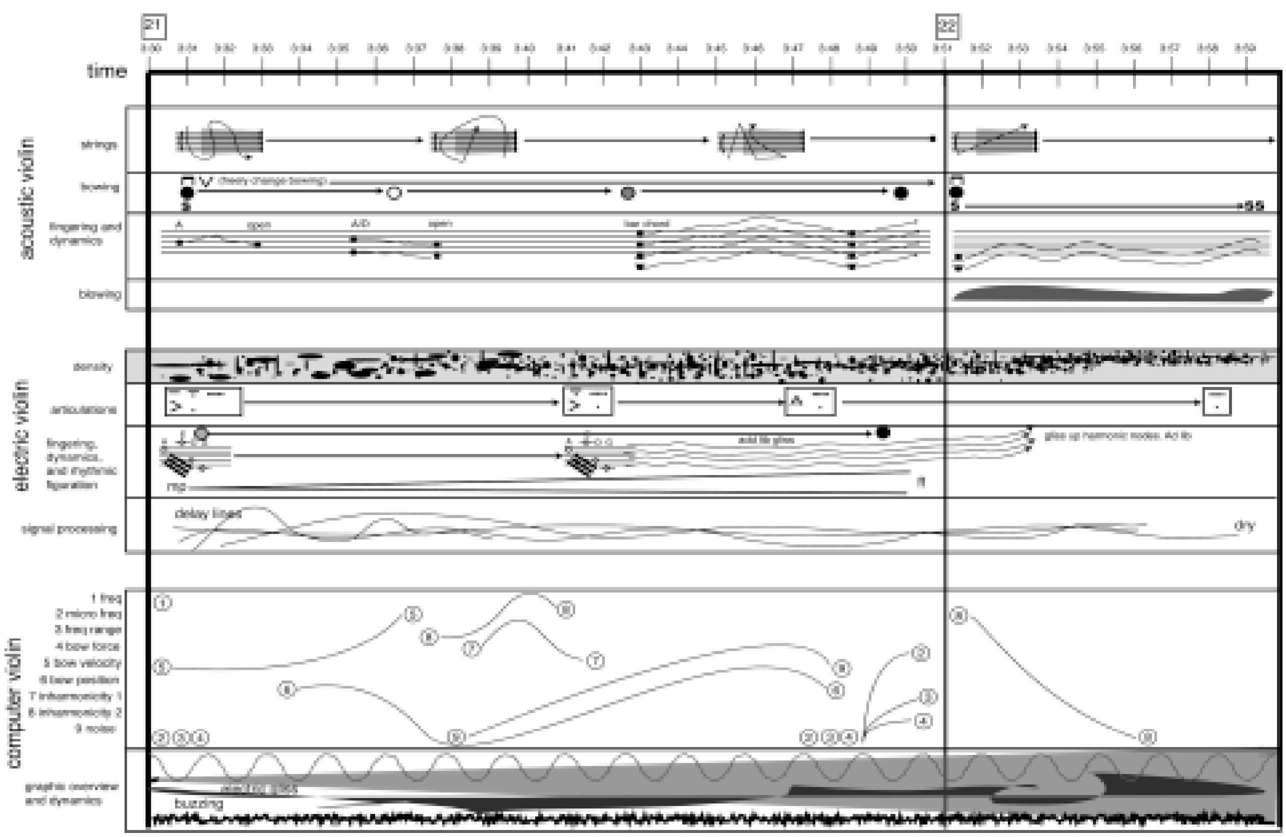



**Figure 15: Score Excerpt from *(dis)Appearance***

## 5. REFERENCES


[1] Burtner, M. “The Metasaxophone: Concept, Implementation, and Mapping Strategies for a New Computer Music Controller.” Organized Sound, vol. 7, n.2. 2002.

[2] Burtner, M. “Noisegate 67 for Metasaxophone”: Proceedings of NIME 2002, Dublin, Ireland. 2002.

[3] Burtner, M., and Serafin, S.. The Exbow Metasax: Compositional Applications of Physical Models Using

[4] Instrument Controller Substitution. *Journal of New Music Research*, vol. 31, June 2002

[5] Burtner, M., and Serafin, S.. Extended Techniques for Physical Models Using Instrument Controller Substitution. Proc. ICMA, Perugia, Italy, 2000.

[6] Burtner, M., and Serafin, S. Real time Extended Physical Models for the Performer and Composer. Proc. ICMC, Havana, Cuba, 2001.

[7] Burns, C. 2001. "Realizing Lucier and Stockhausen: Case studies in the performance practice of electroacoustic music." Proceedings, International Computer Music Conference 2001, pp. 40-43. San Francisco: International Computer Music Association.

[8] Chadabe, J. “The Limitations of Mapping as a Structural Descriptive in Electronic Instruments. Proceedings of NIME 2002. Dublin, Ireland. 2002.

[9] Cook, P. Principles for Designing Computer Music Controllers. Proc. of NIME-01, Seattle, USA, 2001.

[10] Hunt, A. 1999. "Radical User Interfaces for Real-time Musical Control." D. Phil Thesis, University of York, UK.

[11] Nichols, C. “The vBow: A Haptic Musical Controller Human-Computer Interface”, Proceedings of the International Computer Music Conference, Berlin. 2000

[12] Serafin, Stefania, J.O. Smith III, and J. Woodhouse. "An investigation of the impact of torsion waves and friction characteristics on the playability of virtual bowed strings," in IEEE Workshop on Signal Processing to Audio and Acoustics, New Paltz, NY, New York, Oct. 1999, IEEE Press.

[13] Orio, N., Schnell, N., Wanderley, M. Input Devices for Musical Expression: Borrowing Tools from HCI. Proc of the NIME-01, Seattle, USA, 2001.

[14] Paradiso, J. New ways to play: Electronic Music interfaces. IEEE Spectrum 34, 1997.

[15] Mathews, M., Verplank, B., and Shaw, R. Scanned Synthesis, A New Synthesis Technique. Proc. ICMC, Berlin, Germany, 2000.

[16] Roads, C. *The Computer Music Tutorial*. 1996, MIT Press, Cambridge, MA, 1996.

[17] Holmes, T. *Electronic and Experimental Music: Pioneers in Technology and Composition*. 2002, Routledge, New York, 2nd Edition. 2002.

[18] Trueman, D. and P.R. Cook. “Bossa: The Deconstruced Violin Reconstructed.” In Proeedings of the ICMC, Beijing. 1999.

[19] Wanderley, M. 2001. "Performer-Instrument Interaction: Applications to Gestural Control of Music." PhD Thesis. Paris, France: University Pierre et Marie Curie - Paris VI.

[20] Young, D. “The Hyperbow Controller” Procedings of NIME 2002, Dublin, Ireland. 2002.

# Sonigraphical Instruments: From FMOL to the *reacTable**

Sergi Jordà
Music Technology Group
Pompeu Fabra University
Ocata 1, 08003 Barcelona, Spain
+34 93 542 2104
sergi.jorda@iua.upf.es

## ABSTRACT
This paper first introduces two previous software-based music instruments designed by the author, and analyses the crucial importance of the visual feedback introduced by their interfaces. A quick taxonomy and analysis of the visual components in current trends of interactive music software is then proposed, before introducing the *reacTable**, a new project that is currently under development. The *reacTable** is a collaborative music instrument, aimed both at novices and advanced musicians, which employs computer vision and tangible interfaces technologies, and pushes further the visual feedback interface ideas and techniques aforementioned.

## Keywords
Interactive music instruments, audio visualization, visual interfaces, visual feedback, tangible interfaces, computer vision, augmented reality, music instruments for novices, collaborative music.

## 1. INTRODUCTION
For the last ten years, my main area of interest and research has focused around the possibilities for bringing new musical creative facilities to non-musicians, without degrading neither the music potentially producible, nor the users' interactive experiences and control possibilities. Moreover, and because of my penchant for free-jazz and improvisation, I have chosen to concentrate on real-time interactive solutions, which I also feel can be more suitable (i.e. more easily encouraging, exciting and rewarding) for the non musicians than the more thought demanding non-real-time compositional tools.

New musical tools or instruments designed for trained musicians, or even for specific performers, can be quite complex and challenging; as a counterpart they may offer a great amount of creative freedom and control possibilities to their players. On the other hand, instruments designed for amateur musicians or for public audiences in interactive sound installations, tend to be quite simple, trying in the best case, to bring the illusion of control and interaction to their users, while still producing "satisfactory" outputs. Logically, these two classes of instruments are often mutually exclusive. Musicians become easily bored with "popular" tools, while casual users get lost with sophisticated ones. But is this trend compulsory? Wouldn't it be possible to design instruments that can appeal to both sectors: tools that like many traditional acoustical instruments, can offer a *low entry fee with no ceiling on virtuosity*? [53] With these questions in mind I started in 1997 the conception and development of FMOL, a path that has recently taken us to the *reacTable**.

## 2. SONIGRAPHICAL PRELIMINARIES

### 2.1 Epizoo (1994-1995)
Several years before FMOL, together with the visual artist and performer Marcel.lí Antúnez I had developed the computer-based interactive performance *Epizoo* (1994-1995).

The project was not a *musical instrument*; at least not only. Integrating elements of body art, videogames and multimedia applications, it allowed volunteers from the audience to play with (or "tele-torture") the performer's (i.e. Antúnez's) naked body via a graphical interface [29][34][45]. *Epizoo*'s graphical interfaces, as seen in figure 1, could seem to come from a weird videogame designed by the likes of Hieronymus Bosch or Giuseppe Archimboldo, but the fact is, that these GUIs still stick to the typical, hypertextual multimedia cd-rom or web approach: buttons (albeit very hidden) for discrete selections, and sliders (or hot-spots that evaluate mouse activity) for continuous controllers.

*Epizoo* musical output was mostly based on wavefile loops and MIDI sequences; loops were often layerable and pitch-changeable, and sequences could be sometimes manipulated in several ways, but each of *Epizoo*'s screens (there are about 15 screens in the complete performance) can be considered in fact more as a musical piece or composition, which happens to have different performances every show, than a true musical instrument. Besides, volunteers did really conduct all the show development, including the music and the lightshow, and they did so through its quite peculiar mouse-driven GUI, but the opportunity to manipulate a real human body seemed to mask all other "banal" interaction possibilities. This, combined with the fact that these users (which could typically have many different "mouse-skills") were being exposed to the interface for the first time, but were still responsible of conducting a show in a cathartic atmosphere, closer to a rock concert or a techno rave than to a *typical* interactive installation, turned *Epizoo* (at least its musical part) into a perfect example for the category earlier exposed: interactive sound installations which promote the user's illusion of control while guarantying their musical output. Whatever the user did, s/he could feel the control over the whole show, but at the same time the output, especially the musical one, would never be "too bad". FMOL was not going to be about that.

**Figure 1. In EAX, one of Epizoo's screens, the eyes follow the mouse. The eyes, mouth, ears and legs are hot spots that can be touched and clicked**

## 2.2 Reintroducing FMOL (1997-2002)

FMOL, a project I started in 1997 when the Catalan theatre group La Fura dels Baus proposed to me the conception and development of an Internet-based music composition system that could allow cybercomposers to participate in the creation of the music for La Fura's next show, supersedes most of *Epizoo*'s musical limitations. The FMOL project has evolved since its debut, and several articles have been written that should not be repeated here. The fact is that FMOL exemplifies several paradigms which can be treated independently. It is primarily a tool for collaborative musical composition on the Internet. This feature that was the motto of the initial project is better exposed in [37], which deals with the social and aesthetic implications of net-music, and [38] which cover more technical aspects of the implementation. Furthermore, implications of computer and web based collective or collaborative music composition and performance, starting with *the League of Automatic Composers* in the late 70s [9] have been widely studied and published in these last years in papers and thesis such as [8] and [27].

Technical aspects of the FMOL software (real-time synthesis engine, etc.) are covered in [35]. The didactical, intuitive and proselytizing aspects of FMOL as a tool for introducing newcomers into experimental electronic music are deeply treated in [41], while [39] or [40] also cover its use as a professional instrument and its attempt at dealing simultaneously with *micro-sonic* and *macro-musical* compositional ideas.

In this paper I want to focus only on the peculiar aspects brought by FMOL's unique user interface, which presents a closed feedback loop between the sound and the graphics: in FMOL, the same GUI works both as the input for sound control and as an output that intuitively displays all the sound and music activity. After explaining deeper this idea, I will discuss different ways where these sonic-graphic relations are present in recent audiovisual software, and the path that has led us to the conception of our new project, the *reacTable**.

## 2.3 FMOL Musical Output

With FMOL I wanted to introduce newcomers to experimental electronic music making. Therefore, for obvious availability reasons, the instrument had to be a mouse-driven software (it can still be freely downloaded at [22]). I also wanted to create a simple and complex tool all at once; a tool that would not dishearten hobbyist musicians, but would still be able to produce completely diverse music, allowing a rich and intricate control and offering various stages of training and different learning curves.

Both goals have been, in my opinion, quite well attained. During the two Internet calls for musical contributions for two of La Fura's shows (January-April 1998 for F@ust 3.0, and September-October 2000 for the opera DQ) more than 1,700 compositions were received in the database [23]. We know now that many of the participants had no prior contact with experimental electronic music and that a few were even composing or playing for the first time, but the final quality of the contributions (which can be heard online, as well as on the the Fura dels Baus' F@ust 3.0-FMOLCD published in 1998 [37], and on the more recent CMJ 2002 companion CD [51]) was quite impressive.

Moreover, I have given several FMOL workshops usually with a mix of musicians and non-musicians, and if the feeling is positive they usually end with public concerts. An improvisation fragment recorded after one of these workshops can also be heard in [51]. The intuitiveness acid test took place in March 2003 during a one-day workshop with 5 to 8-year old kids from Galicia (Spain), which ended with surprising collective improvisations.

It takes about half-hour to start having fun with the instrument, and several hours to acquire some confidence and produce controllable results. However, after five years of playing it, I am still learning it and do often discover hidden features. Because, and that is another important point, it happens that the instrument I originally designed as a cheap and freely available system for "experimental electronic music proselytism", turned to be, to my own surprise, my favorite instrument for live concerts. Since 1999, the FMOL Trio (Cristina Casanova and me on FMOL computers, plus Pelayo F. Arrizabalaga on saxophones/bass clarinet and turntables) performs free-form improvised electronic music and has produced several live CDs [21][24][25][26].

## 2.4 FMOL Visual Feedback

Arguably, visual feedback is not very important for playing traditional instruments, as the list of first rank blind musicians and instrumentalists (e.g. Ray Charles, Roland Kirk, Tete Montoliu, Joaquín Rodrigo, Stevie Wonder …) may suggest. But traditional instruments usually bring other kinds of feedback, like haptic feedback [12] [30], which is not so often present in digital instruments, at least in the "cheap" ones. Besides, why should not digital instruments use at their advantage anything that could broaden the communication channel with its player? I am convinced that in the case of FMOL, its unique visual feedback has been a fundamental component for its success as a powerful and at the same time intuitive and enjoyable instrument.

FMOL mouse-controlled GUI is so tightly related to the synthesis engine architecture, that almost every feature of the synthesizer is reflected in a symbolic, dynamic and non-technical way in the interface. In its rest position the screen looks like a simple 6x6 grid or lattice. Each of the six vertical lines is associated with one voice generator (FMOL's sound engine supports six real-time synthesized stereo audio tracks or channels), while the horizontal lines are associated with the effects processors (filters, reverbs, delays, resonators, frequency, amplitude or ring modulators, etc.), embedded in each track. All of these lines work both as input devices (controllers) that can be picked and dragged with the mouse, and as output devices that give dynamic visual and "sonic" feedback.

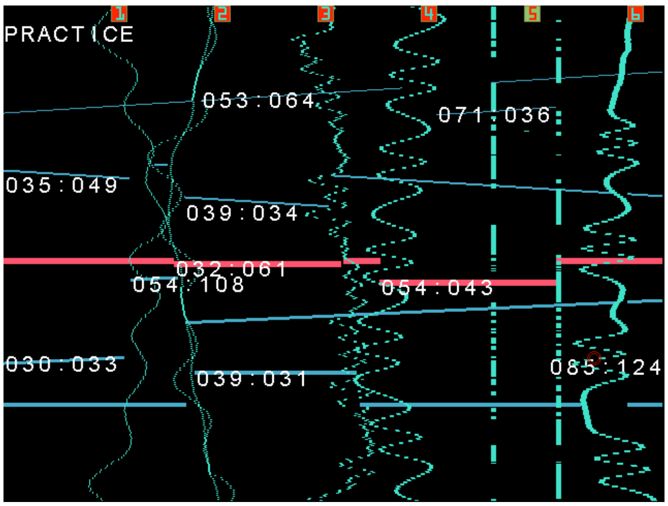



**Figure 2. FMOL in action**

Mappings and detailed control mechanisms are explained better in [41]. The key point is that when multiple oscillators or segments are active (FMOL engine includes 24 LFOs and 96 parameters to control), the resulting geometric "dance", combined with the six-channel oscilloscope information given by the strings, tightly reflects the temporal activity and intensity of the piece and gives multidimensional cues to the player. Looking at a screen like figure 2 (which is taken from a quite dense FMOL fragment), the player can intuitively feel the loudness, frequency and timbrical content of every channel, the amount of different applied effects, and the activity of each of the 24 LFOs. Besides, no indirection is needed to modify any of these parameters, as anything in the screen behaves simultaneously as an output and as an input.

## 3. SONIGRAPHICAL TOOLS

### 3.1 Media players and VJ Tools

In order to show the secular catacomb stage of visual music, Bernard Klein affirmed in his 1927 book *Color-Music: the Art of Light* that "it is an odd fact that almost everyone who develops a color-organ is under the misapprehension that he or she, is the first mortal to attempt to do so" [42]. This assessment could surely not be pronounced anymore nowadays. While since its beginning, digital technologies have boosted multimodality and any kind of parallelism between image and sound in any of their two directions, the truth is that in the last few years, we have seen the flourishing of many software programs or environments that deal with this duality in several ways, even creating distinct families of tools each with its well defined idiosyncrasy.

Following the trend started with the popular music visualization freeware program *Cthugha* released around 1994 and described on its birth as *an oscilloscope on acid* [19], current software music players, like *WinAmp* or *MS Media Player*, come with dozens of fancy visualization plug-ins, that allow the user to view the results of the music in many different ways. These systems can be generally described with the following scheme.


**Figure 3. Elements of a standard music player with visualization**

However, these systems are not very interactive, except that users can decide to change the visualization plug-in, applying thus a discrete change to block D. When an audiovisualizer of this kind becomes interactive, we have what we could call a VJ Tool [6] [52]. Such tools exist as stand-alone programs such as Jaromil's *FreeJ* [28], *Arkaos* [4] or *Resolume* [48], or can be easily built using visual programming environments like MAX + (Nato or Jitter), or PD + (GEM or Framestein), to name a few of the more popular software combinations.

In this new case, depending on the particular system design, the user could interact at any step of the chain.

Using the aforementioned programming environments, one can also decide to take the complementary approach, and build a musical instrument or a sound synthesizer which can be directly controlled by the analysis of some dynamic visuals. These image input can be of any kind (synthetic, abstract, video, etc.), and can come from any source (stored movies, real-time generated animations, live video input, etc.) Although different in concept, this alternative scheme could also include computer vision based musical controllers.


**Figure 4. Elements of an image controlled music generator**

However, none of these two approaches (Sound→Visuals or Visuals→Sound) does generally close the control loop as FMOL does (i.e. in VJ tools, the way the user modifies the graphics does not affect the music). Besides, they usually present two windows at least: one for the visual output (or input, depending on the chosen approach) and an additional one (the "control panel") for parameter modification; they do not allow to modify the *visuals window* by playing directly on it.

### 3.2 Golan Levin's Work

To my knowledge, only Golan Levin's work follows an audiovisual approach comparable to the one I've presented in FMOL. The fact is that although we unfortunately did not know about each other until quite recently, I believe our goals and approaches share many common aspects. In his master thesis "Painterly Interfaces for Audiovisual Performance" he proposes a system for the creation and performance of dynamic imaginery and sound, simultaneously, in real-time, and with basic principles of operation easy to deduce, while at the same time capable of sophisticated expressions and indefinitely masterable [43].

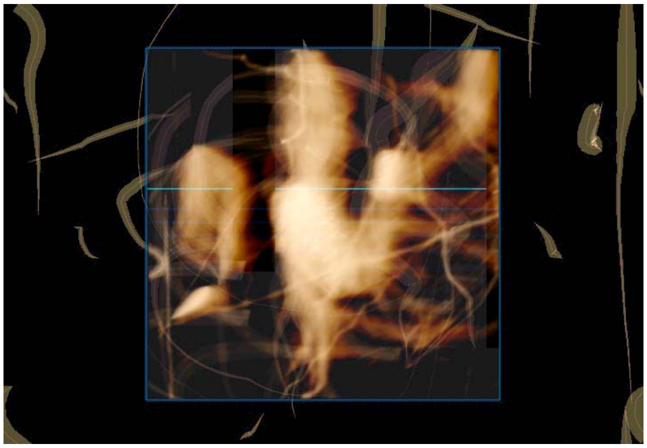


**Figure 5. A screenshot of Golan Levin's Yellowtail**

Levin talks about an inexhaustible, extremely variable, dynamic, audiovisual substance that can be freely painted, and he has developed many audiovisual tools, like *Yellotail*, *Loom*, *Warbo*, *Aurora* and *Floo*, which follow this path. Perhaps, the major difference in our approaches may be the fact that Levin, like Oskar Fischinger, the animator that in the 40s invented the *Luminograph* color organ [44], is willing to play light while playing sound. For him, image is therefore an end in itself. While for me, it is only the means to an end: a more intuitive interface for creating music.

# 4. THE *REACTABLE**

## 4.1 Preliminary

Last year, together with the doctorate students Alvaro Barbosa, Gunter Geiger, Rubén Hinojosa, Martin Kaltenbrunner and José Lozano, and the undergraduate students Carlos Manias and Xavier Rubio, we constituted the Interactive Systems Team, inside the Music Technology Group led by Xavier Serra at the Pompeu Fabra University of Barcelona. One of the initial projects was to port FMOL to Linux and make it open-source, which seemed also a good opportunity for revamping the system [1].

Looking at the way people have used FMOL, and using it myself for improvisation in different contexts and with different musicians, has raised ideas new features and modifications. But we also felt that this control complexity could not be permanently increased; there are limits to what can be efficiently achieved in real-time by means of a mouse and a computer keyboard. Building an external FMOL controller for a faster and more precise multiparametric control seemed therefore a tempting idea. Designing a video detection or ultrasound system that would allow musicians to interact on a big projection screen, grabbing and moving strings with their hands, was the first idea we had. This could surely add a lot of visual impact to live concerts, although we also felt that musical control and performance may not necessarily improve with it. These and other considerations took us to a completely new path, which should profit the knowledge gained during this years and bring it to a much more ambitious project: The *reacTable**.

## 4.2 Intentions

We aim at the creation of a state-of-the-art interactive music instrument, which should be collaborative (off and on-line), intuitive (zero manual, zero instructions), sonically challenging and interesting, learnable, suitable for complete novices (in installations), suitable for advanced electronic musicians (in concerts) and totally controllable (no random, no hidden presets…). The *reacTable** should use no mouse, no keyboard, no cables, no wearables. It should allow a flexible number of users, and these should be able to enter or leave the instrument-installation without previous announcements. The technology involved should be, in one word, completely transparent.

## 4.3 Computer Vision and Tangible Objects

As the Tangible Media Group directed by Professor Hiroshi Ishii at the MIT Media Lab states, "People have developed sophisticated skills for sensing and manipulating our physical environments. However, most of these skills are not employed by traditional GUI…. The goal is to change the *painted bits* of GUIs to *tangible bits*, taking advantage of the richness of multimodal human senses and skills developed through our lifetime of interaction with the physical world." [32][50]. Several tangible systems have been constructed based on this philosophy. Some for musical applications, like *SmallFish* [49], the *Jam-O-Drum* [10] [11] [33], the *Musical Trinkets* [46], *Augmented Groove* [7] [47] or the *Audiopad* [5], but we believe that no one attempts the level of integration, power and flexibility we propose.

***reacTable* = FMOL + MAX + JamODrum***

Substitute if you want *MAX* with *PD* or *JMax* or even *AudioMulch*. Substitute the *Jam-O-Drum* with the table version of *Small Fish*. You can even substitute FMOL, but only with Levin's systems , and you will get an initial idea about what the *reacTable** is all about: a table-based collaborative music instrument that uses computer vision and tangible user interfaces technologies, within a MAX-like architecture and scheduler, and with FMOL-inspired HCI models and visual feedback.

The *reacTable** is a musical instrument based on a round table, which has no sensors, no cables, no graphics or drawings. A video camera permanently analyses the surface of the table, while a projector draws a dynamic and interactive interface on it.

Many interesting and promising computer vision tools, mostly based on body motion capture, are being developed for musical applications [15] [16]. However, many of us do not feel too comfortable "dancing" in front o a video camera (some even without camera!), while we all work and socialize around tables. For this reason our computer vision system does not attempt to track body motion. Instead, it focuses on tracking the hand movements over the table, and on detecting the nature, position and orientation of the objects that are distributed on its surface.

These objects are mostly passive and made out of plastic or wood of different shapes. Users interact with them by moving them, changing their orientation on the table plane or changing their faces (in the case of volumetric objects). More complex objects include (but are not limited to) flexible plastic tubes for continuous multiparametric control, little wooden dummy 1-octave keyboards, combs (for comb-filters), or other everyday objects. In case an object needs sensors, its communication with the host computer will be wireless.

## 4.4 Visuals (1)

The projection follows the objects on the table wrapping them with auras or drawing figures on top of them. The projection covers also the whole table surface with dynamic and abstract elements that reflect all the system's activity, and depend on the hands' movements and trajectories, the objects' types and positions, and the relations between them all. The projection never shows buttons, sliders or widgets of any kind.

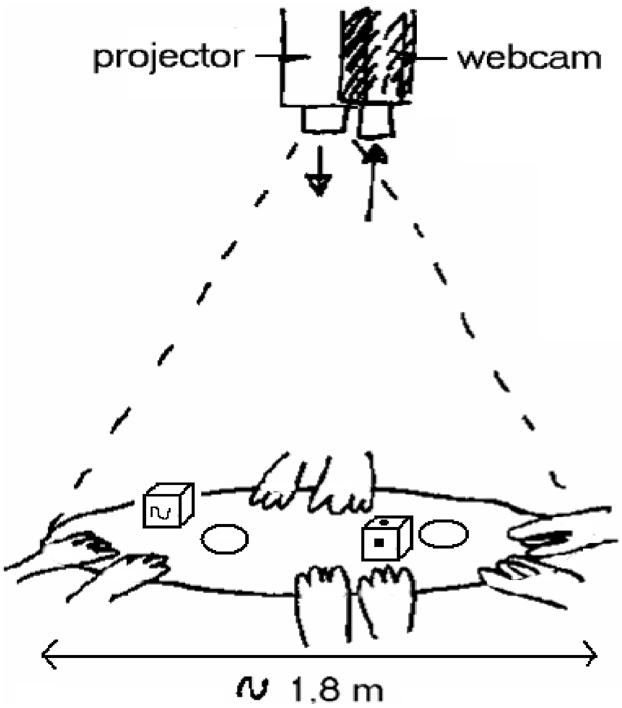



**Figure 6. The *reacTable** simplified scheme**

## 4.5 But where is MAX?

FMOL has proven to be quite flexible. Its palette of sound generators and processors includes more than 20 algorithms that (with internal configuration variations) constitute a bank of 127 presets the user can select and apply to any of the strings. This process of "building an orchestra" is not done in real-time while playing, but in a different, more conventional window. Besides, all FMOL macro-control of form is done like in traditional analog synthesizers, by means of LFOs and arpegiators. More sophisticated control sources, such as algorithmic generators, pitch filters, etc. cannot fit coherently into the FMOL interface. The *reacTable** overcomes these restrictions by adapting one of the more powerful real-time computer music software paradigms implemented in the last decades.

Like MAX and all of its cousins, the *reacTable** distinguishes between control and sound objects, and between control and sound connections. Unlike MAX, and more like Audiomulch (which however has no explicit control flux), the *reacTable** objects are more high-leveled; the *reacTable** is an ambitious project but it is an instrument, not a programming language!

When a control flow is established between two objects, a thick straight line is drawn between them, showing by means of dynamic animations, the flux direction, its rate and its intensity. Visual feedback will also guarantee that LFOs and other macrotemporal values will be perceived as blinking animations projected on top of the related objects, showing frequency and shape (e.g. square vs. sinusoidal).

## 4.6 Visuals (2): Audio flow

Where control flow lines are straight and simple, audio flow lines are *organic* and complex. Their dynamic shapes will show the macrotemporal audio variations (vibratos, tremolos, tempo and rhythms…) and their interior (colors, intensities…) will depend on their spectral audio content.

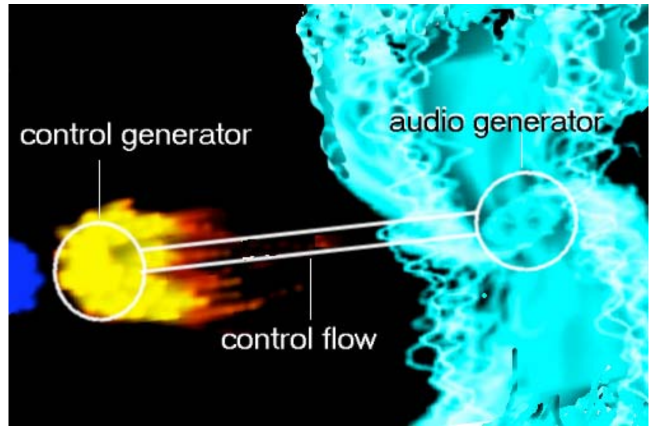



**Figure 7. Control and audio flow simulation**

Users will also be able to control, modify or fork audio flows without using additional objects, but just by waving their hands, as if they were digging water channels in the beach sand.

## 4.7 Avoid user's frustration at any cost

To avoid frustrations, a system does not necessarily have to be completely understandable, but it has to be coherent and responsible. Unlike MAX, the *reacTable** has to work "by default" and any gesture has to produce audible results. Here are some of its laws:

- There is not anything like an *editing mode* and *running mode* (at least for *installation users*); the *reacTable** is always running and always being edited!
- Objects are not active until they are touched
- Active objects have a dynamic visual *aura*
- Objects are interconnected by proximity
- If on start-up, a user activates an object that does not sound (i.e. a control object) the closest audio object is automatically linked to it (and the link is visualized)
- Moving an object on the table can change the relations with the other objects
- Relations can also be "fixed" touching two objects with the two hands. Fixed links are shown with a thicker line or a different color.

Perry Cook, in an informal music controllers design decalogue, ironically points that "smart instruments are often not smart." [18]. Although we basically agree with him, we have come to the conclusion that a system like the *reacTable** must show some kind of intelligent behavior. For example, as most of the control objects are adimensional (some, like the dummy keyboard, are not), when one adimensional control flux is sent to an object that can accept different inputs, the system chooses what the best parameters to control in every case are. In another demonstration of intelligent behavior, the system may suggest interesting candidates for a given configuration, by highlighting the appropriate objects (in a manner not to be confused with LFOs).

The *reacTable** wants to be *user-proof*. For instance, it seems natural that after some minutes, people will start stressing the system in different ways, like placing personal objects onto the table. Although it is no possible to anticipate all objects that people may use, some of the more common could be detected (cigarette packets, mobile phones, keys, pens…) and a "funny" functionality could be added to them (e.g. mobiles could generate pitch in a "mobile-fashion").

## 5. CURRENT IMPLEMENTATION

The *reacTable** project has started in December 2002 coinciding with the foundation of the Interactivity Team within the Music Technology Group (MTG). We are currently working and researching all the main threads in parallel (computer vision and objects recognition, sound engine architecture, interactivity logic, sound visualization, etc.) while designing the core and the integration of all these branches. Computer vision and objects recognition is being carried using both *Eyesweb* [13][14][20] and the Intel ® Image Processing (IPL) and Open Computer Vision (OpenCV) libraries [31]. We will not describe here any of these issues, as we soon plan to devote a whole paper to them. The synthesis engine is being implemented using the CLAM libraries, the open-source, multiplatform C++ libraries for real-time audio being developed at the MTG [2][3][17].

In parallel with these two main productions threads, we are working with a *reacTable** software-only simulator (that runs on both Linux and Windows), which is an essential workbench for defining and refining all of the system laws, evaluating user interaction and objects' connectivity rules, as well as determining the panoply of sound and music objects, their roles, behaviors, the way they synchronize between them, etc. The simulator GUI has been implemented in Java by Martin Kaltenbrunner, while Gunter Geiger is working on its sound engine using PD for quick prototyping. Both modules communicate via TCP/IP, a flexible architecture which also permits multi-user simulation by running different instances of the GUI in different computers.

Figure 8 shows a *reacTable** simulator screenshot, with only four kinds of sound objects: High Frequency Oscillators (circles), Low Frequency Oscillators (triangles), filters (smoothed squares) and time-based effects (squares). Visual feedback is yet very simple; all connection lines are straight and do not suggest therefore any of the information they transmit, except that to distinguish between the two types of connections, audio lines are drawn in dark, while control lines are light grey. Each dark line flowing into the audio sink represented by the black central circle corresponds therefore to an independent audio thread.

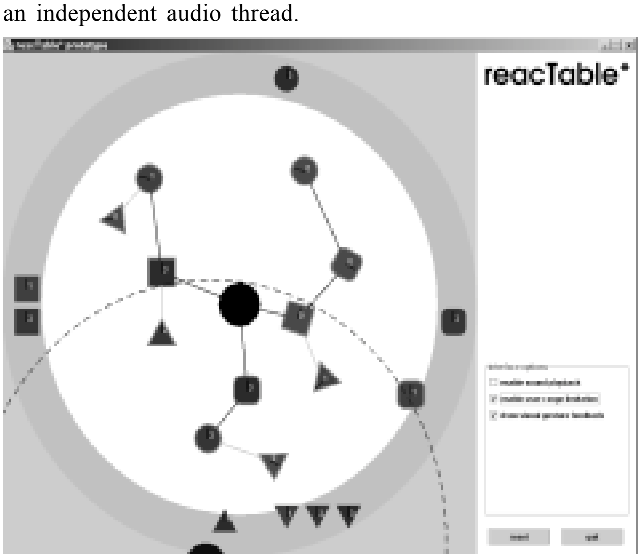



**Figure 8. *reacTable** simulator snapshot**

At this early stage, what we have is a sort of higher level MAX in which users can drag objects that dynamically interconnect between them according to the rules defined in the system. Using the right mouse button objects can also spin around and connection lines can be broken. Parameters are calculated from the rotation angle of the objects as well as from the length and the orientation of their connections. As simple as it still is, this flexible and dynamic architecture already permits for some fast sound changes that seem impossible to attain in an analog modular synthesizer, which it somehow evokes.

## 6. FUTURE WORK AND CONCLUSION

The *reacTable** is an ambitious project. Unlike many new designed instruments, its origin does not come from approaching its creation by exploring the possibilities of a specific technology, nor from the perspective of mimicking a known instrumental model. The *reacTable** comes from our experience designing instruments, making music with them, and listening and watching the way others have played them. Needless to say, we have deposited a great hope and expectation on it. We plan to have the first integrated by autumn 2003 and a first full working version by spring 2004.

## 7. ACKNOWLEDGMENTS

I would like to thank all the members of the Interactive Systems Team, and specially Martin Kaltenbrunner, for their suggestions on this paper, and Xavier Serra for his support on this project.

## 8. REFERENCES

[1] Agnula homepage: http://www.agnula.org

[2] Amatriain, X et al. "CLAM: An OO Framework for Developing Audio and Music Applications", Proceedings of 17th Annual ACM Conference on Object-Oriented Programming, Systems, Languages and Applications. Seattle, WA, USA, 2002.

[3] Amatriain, X., P. Arumi, and M. Ramírez, "CLAM, Yet Another Library for Audio and Music Processing?", Proceedings of 17th Annual ACM Conference on Object-Oriented Programming, Systems, Languages and Applications. Seattle, WA, USA, 2002.

[4] Arkaos homepage : http://www.arkaos.net

[5] Audiopad homepage: http://tangible.media.mit.edu/projects/Audiopad/Audiopad.htm

[6] Audiovisualizers homepage: http://audiovisualizers.com/

[7] Augmented Groove Homepage: http://www.csl.sony.co.jp/~poup/research/agroove/

[8] Barbosa, A., 2002. *Computer-Supported Cooperative Work for Music Applications*. Ph.D. Thesis proposal, Pompeu Fabra University, Barcelona.

[9] Bischoff, J., Gold, R., and Horton, J. 1978. "Music for an interactive Network of Computers". Computer Music Journal, Vol. 2, No.3, pp 24-29.

[10] Blaine, T. and T. Perkis, "Jam-O-Drum, A Study in Interaction Design," Proceedings of the ACM DIS 2000 Conference, ACM Press, NY, August 2000.

[11] Blaine, T. and C. Forlines, "Jam-O-World: Evolution of the Jam-O-drum Multi-player Musical Controller into the Jam-O-Whirl Gaming Interface," Proceedings of the 2002 Conference on New Interfaces for Musical Expression (NIME-02), Dublin Ireland, May 2002.

[12] Bongers, B. "The Use of Active Tactile and Force Feedback in Timbre Controlling Musical Instruments." *Proceedings of the 1994 International Computer Music Conference.* San Francisco: International Computer Music Association, pp. 171-174.

[13] Camurri, A., M. Ricchetti and R. Trocca, "EyesWeb - toward gesture and affect recognition in dance/music interactive systems", Proceedings IEEE Multimedia Sustems '99.

[14] Camurri, A. et al., "A real-time platform for interactive performance", *Proceedings of the 2000 International Computer Music Conference.* San Francisco: International Computer Music Association.

[15] Camurri, A., G.De Poli, M.Leman, G.Volpe (2001) A Multi-layered Conceptual Framework for Expressive Gesture Applications, Proc. Intl MOSART Workshop, Barcelona, Nov. 2001.

[16] Camurri, A., R.Trocca, G.Volpe, 2002. Interactive Systems Design: A KANSEI-based Approach, Proc. NIME2002, Dublin, May 2002.

[17] CLAM Homepage: http://www.iua.upf.es/mtg/clam/

[18] Cook, P., "Principles for Designing Computer Music Controllers", Proceedings of the 2001 New Instruments for Musical Expression Workshop. Seattle: CHI 2001.

[19] Ctugha homepage http://www.afn.org/~cthugha/index.html

[20] Eyesweb Home page: http://www.eyesweb.org/

[21] Feller, R. "FMOL Trio: Live at Metronom." Computer Music Journal, Vol. 26, No.2., pp 110-112.

[22] FMOL Home page: http://www.iua.upf.es/~sergi/FMOL

[23] FMOL-DQ Compositions Database: http://teatredigital.fib.upc.es/dq/eng/fmol/database.htm.

[24] FMOL Trio. 2000. *Live at Metronom* (CD Audio). Barcelona: Hazard Records 010.

[25] FMOL Trio. 2002. *Night in the Chapel* (CD Audio). Barcelona: Hazard Records 025.

[26] FMOL Trio. 2002. *The Köln Concert* (CD Audio). Barcelona: Hazard Records 028.

[27] Föllmer, G. Soft Music. 2001. San Francisco Museum of Modern Art; ZKM – Center for Art and Media – Karlsruhe. http://www.sfmona.org/crossfade

[28] FreeJ Homepage: http://freej.dyne.org/

[29] Gianetti, C., *Marcel.lí Antúnez Roca: Performances,Objects and Drawings*, MECAD, Barcelona (1998).

[30] Gillespie, B. 1999. "Haptic Manipulation". In P. Cook, eds. *Music Cognition and Computerized Sound: An Introduction to Psychoacoustics*. Cambridge, Massachusetts: MIT Press, pp. 247-260.

[31] Intel ® OpenCV libraries for Computer Vision and IPL libraries for image processing: http://www.intel.com/research/mrl/research/opencv/ and http://sourceforge.net/projects/opencvlibrary

[32] Ishii, H and B. Ullmer, "Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms", Proceedings of CHI 97 Conference on Human Factors in Computing systems, Atlanta, Georgia USA, 22-27 March 1997.

[33] JamODrum homepage: http://www.jamodrum.net/

[34] Jordà, S., "EPIZOO: Cruelty and Gratuitousness in Real Virtuality." Proceedings of 5CYBERCONF, fifth International Conference on Cyberspace (1996), at http://www.telefonica.es/fat/ejorda.html.

[35] Jordà, S., Aguilar, T. 1998. "A graphical and net oriented approach to interactive sonic composition and real-time synthesis for low cost computer systems." Digital Audio Effects Workshop Proceedings, pp. 207-210.

[36] Jordà, S. and La Fura dels Baus. 1998. F@ust 3.0 - FMOL (CD Audio). Madrid: Fundación Autor 02801.

[37] Jordà, S. 1999. "Faust music On Line: An Approach to Real-Time Collective Composition on the Internet." Leonardo Music Journal, Vol. 9, pp. 5-12.

[38] Jordà, S., Wüst, O. 2001. "Architectural Overview of a System for Collaborative Music Composition Over the Web." Proceedings of the 2001 International Computer Music Conference. International Computer Music Association.

[39] Jordà, S. "Improvising with Computers: A personal Survey (1989-2001)." Proceedings of the 2001 International Computer Music Conference. San Francisco: International Computer Music Association.

[40] Jordà, S. 2002. "Improvising with Computers: A personal Survey (1989-2001)." Journal of New Music Research, Vol. 31, No.1, pp. 1-10.

[41] Jordà, S. 2002. "FMOL: Toward User-Friendly, Sophisticated New Musical Instruments." Computer Music Journal, Vol. 26, No.3. pp 23-39.

[42] Klein, A. B., Colour-Music: The Art of Light. Crosby, Lockwood & Son, London, 1927.

[43] Levin, G. 2000. "Painterly Interfaces for Audiovisual Performance", Master Thesis, Massachusets Institute of Technology, pp. 53-55. http://acg.media.mit.edu/people/golan/thesis/

[44] Moritz, M., "The Dream of Color Music, And Machines That Made it Possible", Animation World Magazine, Issue 2.1, April 1997. http://www.awn.com/mag/issue2.1/articles/moritz2.1.htm l

[45] Lozano-Hemmer, R. 1996. "Perverting Technological Correctness". Leonardo, Vol. 29, No. 1.

[46] Paradiso, J. & Hsiao, K., "Musical Trinkets: New Pieces to Play", SIGGRAPH 2000 Conference Abstracts and Applications, ACM Press, NY, July 2000, p. 90.

[47] Poupyrev, I, Augmented Groove: Collaborative Jamming in Augmented Reality, ACM SIGGRAPH 2000 Conference Abstracts and Applications, p. 77.

[48] Resolume homepage: http://www.resolume.com

[49] SmallFish homepage: http://hosting.zkm.de/wmuench/small_fish

[50] Tangible Media Group: http://tangible.media.mit.edu/index.html

[51] Various Authors, 2002. Computer Music Journal companion CD. Computer Music Journal, Vol. 26, No. 4.

[52] VJCentral homepage: http://www.vjcentral.com

[53] Wessel, D. and M. Wright. "Problems and Prospects for Intimate Musical Control of Computers." Proceedings of the 2001 New Instruments for Musical Expression

# Ergonomic Design of A Portable Musical Instrument

Motohide Hatanaka
Center for Design Research, Department of Mechanical Engineering Design Division
Building 560, 424 Panama Mall, Stanford University
Stanford, California 94305, USA
+1-650-723-3286
motohat@cdr.stanford.edu

## ABSTRACT
A handheld electronic musical instrument, named the Bento-Box, was developed. The motivation was to develop an instrument which one can easily carry around and play in moments of free time, for example when riding public transportation or during short breaks at work. The device was designed to enable quick learning by having various scales programmed for different styles of music, and also be expressive by having hand controlled timbral effects which can be manipulated while playing. Design analysis and iteration lead to a compact and ergonomic device. This paper focuses on the ergonomic design process of the hardware.

## Keywords
MIDI controller, electronic musical instrument, musical instrument design, ergonomics, playability, human computer interface.

## 1. INTRODUCTION
The busy modern lifestyle provides few opportunities for music lovers to perform. However, people do have many short intervals of free time during which they can potentially play musical instruments. For example, many people spend time less actively when riding or waiting for public transportation for commuting. It would be a boon for busy music lovers if there were a musical instrument that they could easily carry around and play anytime anywhere without disturbing others, just like a portable audio device such as the Sony Walkman [10]. For instance, a harmonica is a very portable instrument but it can disturb others if played in public. On the other hand, an electronic keyboard can be played and heard through a headphone so as not to disturb people, but it is awkward to carry around and it may also irritate surrounding people because of its size. What would be ideal instead is a compact portable silent musical instrument with electric sound output (see Figure 1).

The Bento-Box was designed and built as a class project for a course on human computer interaction (HCI) theory and practice at Stanford University [11]. Basics of HCI theory as well as electronics and computer sound generation were taught, and students were given approximately a month to build musical instruments of their own interest in groups of two or three.

## 2. CONCEPT
### 2.1 Project Goal, Design Criteria, and Functional Requirements
The general goal of the project was to develop a device that contains maximum musical expressiveness in a limited size while assuring ease and comfort of performance. Some criteria were established to limit certain properties of the device for the user and the people around the performer. Functional requirements were also identified to assure satisfactory performance for the user.

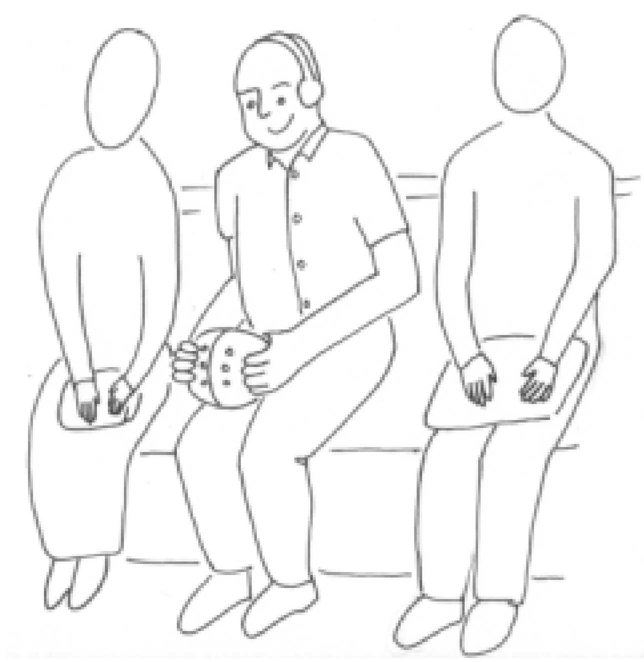


**Figure 1. Compact quiet musical instrument for playing in public places**

The device had to be portable both in terms of size and mass. The design team aimed for a size that can be held with two hands brought fairly close together and with a mass that can be supported comfortably for at least half an hour. The size constraints also result in a unit that the user can easily carry around, hence the name of the device Bento-Box (*bento* is a Japanese word for packed lunch). The performer's range of motion also had to be small enough not to disturb the surrounding people in crowded public transportation where people are seated shoulder to shoulder. So the device was designed to be played without having to move shoulders but only hands, wrists, and forearms. Furthermore, the instrument also had to be quiet during performance. Hence, quiet input devices, such as buttons without clicking noise, were selected to minimize noise generation.

The design team decided that they wanted an instrument that can play melody with at least two octaves of tone range for sufficient musical expression. They also agreed on having at least two timbral effects to control. Other functions were added to the device as required or desired as the design evolved.

### 2.2 Design Strategy
The design takes full advantage of the fact that this is an electronic instrument. An acoustic musical instrument may

have structural design constraints that may result in poor ergonomics. In contrast, electronic instruments have much more freedom in structural design since the sound usually does not depend on the physical properties of the device. Hence, here, the instrument was designed prioritizing ergonomics, while also assuring high expressiveness. Such a design can be done through a combined analysis of the tasks, hardware characteristics of input devices, and human ergonomics (see Figure 2).

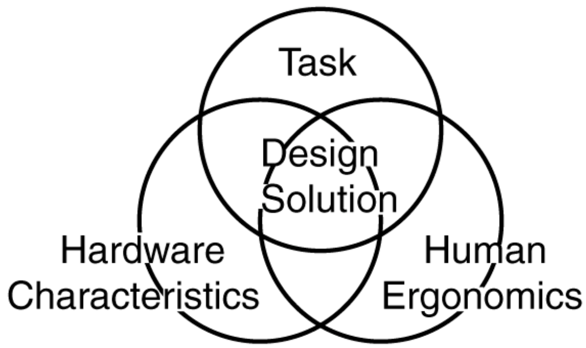



**Figure 2. Design approach involves the analyses of task, hardware characteristics, and human ergonomics**

Many compact handheld and hand-worn musical devices take advantage of being free from physical constraints seen in acoustic instruments. The hardware designs often differ greatly from one to another because of the intended application. The Hands used by Waisvisz is a pair of rather sophisticated hand-worn MIDI-controllers with numerous sensors that allows expressive performance [12]. A family of hand-held instruments developed recently by Weinberg and collaborators generally have simple hardware that are easy to operate and comfortable to hold, probably considering use by children, and their applications have broad goals such as enabling collaboration among performers [13-15]. Since we were developing an instrument for an entirely new setting, design inspiration were sought not only from such existing musical devices but also other electronic devices used in similar settings such as mobile phones and portable video games like the Nintendo Gameboy [7]. However, the greatest source of ideas were various types of brainstorming exercises highlighted by building quick mockups and playing with them, acting out as if they were real instruments.

## 2.3 Initial Design

The initial concept was to have a spherical device that had touch sensors, such as buttons or force sensitive resistor (FSR) pads, on the surface for basic melody and chord performance. There were also to be several effects controlled by twisting and squeezing the device (see Figure 3).

In order to maximize the playability and sound range of the instrument on the limited space of the instrument, a scale (style) selection slider was to be installed. Notes in the selected scale and key are assigned to the melody performing sensors (see Figure 4). This assumes that the performer only needs to play notes that are in the scale. Hence, having a sensor (or a part of a sensor) assigned to an off-scale note would be a waste of space. This has an advantage, especially for beginner musicians trying to improvise, in that the performer will not produce a note that is terribly dissonant.

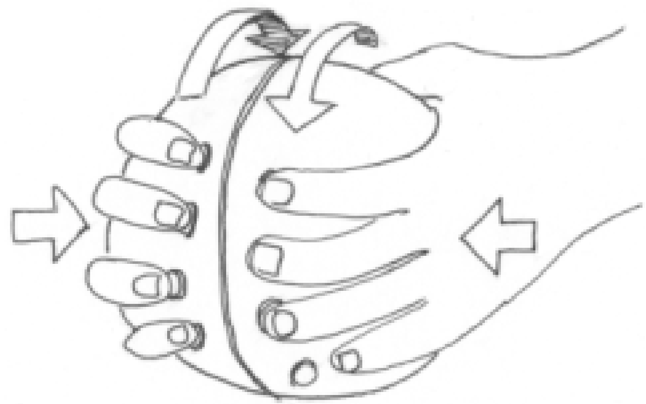


**Figure 3. Controlling effect parameters by twisting and squeezing the instrument was one of the earliest ideas.**

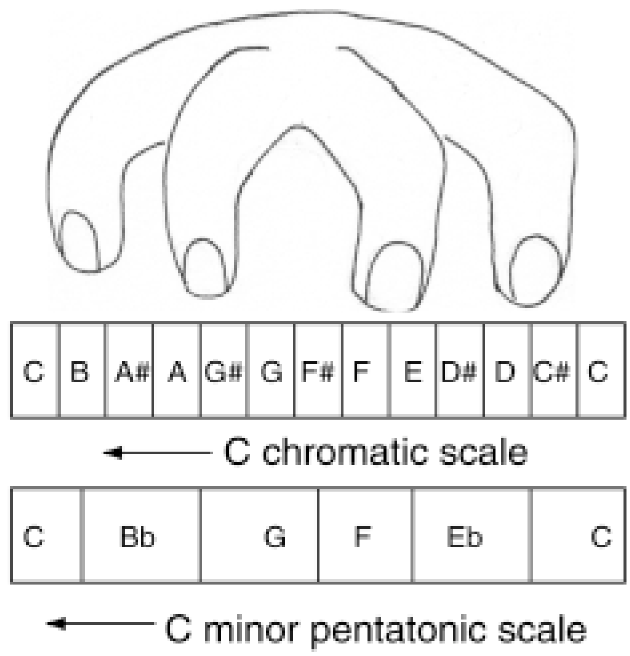



**Figure 4. By changing scale settings, the instrument limits itself to produce only the notes that are in the specified scale.**

# 3. HARDWARE DESIGN

## 3.1 Shape Determination

The hardware design went through ergonomics analysis and testing using mockups to maximize playability and expressiveness. The usefulness of prototyping in early stages is discussed in a book on product design by Kelley [5]. Three mockups were built using rigid urethane foam. One was cylindrical, with tapered ends, 90mm diameter and 115mm length, and the other two were rectangular, one being narrow (80mmx150mm x55mm thick) and the other being almost square in its front profile (150mmx150mmx55mm) (see Figure 5).

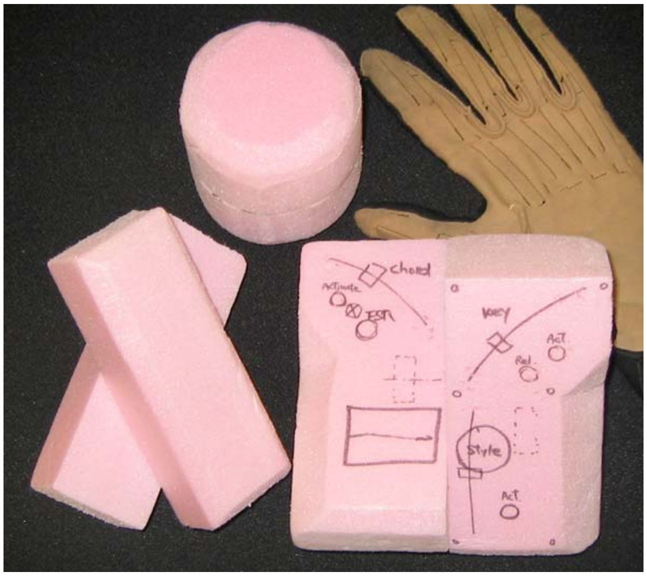



**Figure 5. Foam mockups for initial instrument design. The glove is placed for size reference.**

For the cylindrical shape, it was discovered that the hands conform too nicely to wrap around it such that it left the fingers with little freedom to move (see Figure 6).

**Figure 6. Having no free space between the instrument and the fingers limits the freedom of finger motion.**

The narrow rectangle relied on the fingers and thumb for supporting the instrument. As a consequence, both of the thumbs as well as two fingers from each hand had to stay on the instrument for stable support. This greatly inhibited their freedom for performance (see Figure 7).

The square shape was selected at the end as its wide shape allowed for the user's palms to support the instrument without hindering the dexterity of thumbs and fingers (see Figure 8). The result of the experiment shows that round shape does not necessarily make a device ergonomic. Various ways of grasping are listed and described in books by Cutkosky [3] and MacKenzie and Iberall [6].

## 3.2 Sensor Selection and Layout

Controlling a timbral effect parameter by twisting the instrument was one of the featured ideas of the design. Twisting and bending are motions that could be easily applied by the performer's wrists. Therefore, making use of that motion was perfectly suited to represent our ergonomics consideration approach for expressiveness maximization. The positioning of the twisting pivot was also experimented and determined using the foam mockups. The pivot was first positioned at the center of the device, which worked fine for the cylindrical shape. However, having the pivot at the center of the rectangular mockups required large motions of the forearms, which could lead to elbow motions that disturb surrounding people. The problem was solved by shifting the pivot toward the wrists.

**Figure 7. In a grip using fingertips, thumbs and some fingers must always remain on the instrument to support it.**

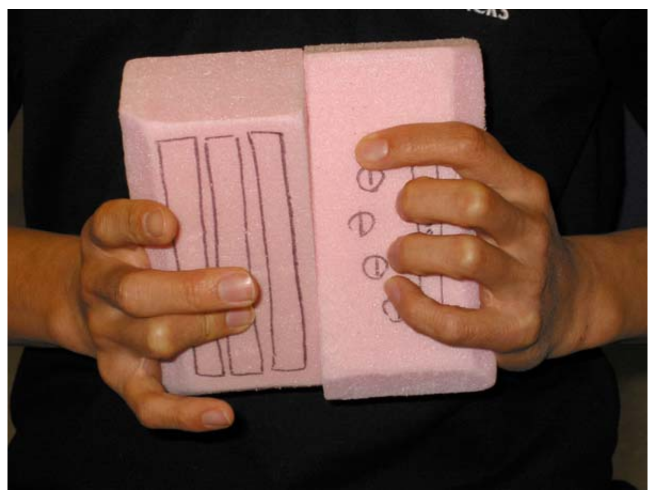


**Figure 8. Palms are used to support the instrument, leaving the thumbs and fingers free to move.**

Another expression parameter was to be controlled with the squeezing force between the two halves of the instrument. The twist and squeeze sensors were integrated in one unit. The rotary potentiometer for sensing the twist was suspended on a linear slider that allows it to translate in the direction parallel to its rotating axis. An FSR was installed behind the potentiometer via a rubber cushion to sense the squeeze.

Design of other input devices was done by analyses and matching of the task, device characteristics, and human ergonomics all together. Predicted interaction frequency as well as the required signal output (continuous or discrete) were listed out for various control parameters (see Table). For example, melody pitch changes are required much more frequently than scale change. Properties of various input devices such as the output signal types and the time required to manipulate them were also listed. The list was used to identify the potential couplings of control parameters and sensors.

**Table 1. Body parts, control parameters, and sensors were matched by analyzing their characteristics.**

| | parameter | No. | control frequency | signal | sensor |
|---|---|---|---|---|---|
| fingers (R) | pitch | 2 | high | continuous | flex pot |
| | volume | 2 | high | continuous | flex FSR |
| Thumb (R) | scale release | 1 | medium | discrete | button |
| | style | 1 | low | continuous | pot |
| | style activate | 1 | low | discrete | button |
| Wrist | octave shift | 1 | high | continuous | pot |
| | compress | 1 | high | continuous | FSR |
| Thumb (L) | key | 1 | low | continuous | slider pot |
| | chord | 1 | medium | continuous | slider pot |
| | key activate | 1 | low | discrete | button |
| | chord activate | 1 | low | discrete | button |
| fingers (L) | sustain | 1 | medium | continuous | slider pot |
| | reverb | 1 | medium | continuous | slider pot |
| | vibrato | 1 | medium | continuous | FSR |
| | harmonics | 1 | medium | continuous | slider pot |

Final selection and physical layout of these input devices on the instrument were done through an experiment to move thumbs and fingers in every possible way on the foam mockup. The objective was to discover how to extract maximum expressiveness out of the hands. The range and speed of motion as well as force and control involved were analysed. Level of comfort was another factor that was tested. The layout determination also considered maintaining enough comfortable room for the thumbs and fingers to rest. It is worth keeping in mind that fingers are good at flexing and extending but they are not so finely controllable nor forceful at abduction or adduction, especially when they are bent. There are also slight but noteworthy differences in the dexterity among the four fingers. The index finger, for example, has much more independence in motion compared to the other three fingers. Meanwhile, a thumb can exhibit a variety of complex motions unrivaled by fingers. Such facts about hands can also be obtained from literature by, for example, Kaplan [4] and Wilson [16], but the effectiveness of mockup experimentation cannot be completely replaced by reading only. However, while experiments can reveal what one can do, literature on ergonomics such as [1] by Cacha can provide more information about what is good or bad for the hands in long term ergonomics so that injuries can be prevented.

Here are some examples from the resulting design. There are two sliders positioned diagonally on the thumb side of the device. They are assigned for key and chord selection. Another slider is placed lower on the device for style change, where it is not as accessible, since it will not be used as frequently as the two others (see Figure 9). The chord/key/style settings change only when an activation button is pressed. This allows the user to control the sliders during performance in preparation for a future change without interfering the ongoing performance.

The right fingers do the melodic performance by pressing buttons (see Figure 10). Fingers are good for selecting and pressing buttons. They are also quite capable of varying pressure on each finger if needed. Fingers on the left hand were used to trigger effects-selection toggle switches.

## 4. SYSTEM CONFIGURATION

Sensor information from the Bento-Box was collected by an Atmel ATMega 163 microcontroller and was sent to a desktop computer as MIDI data. Pd [8] was employed to process the data and generate sound with plugins created from Synthesis ToolKit (STK) physical models [2]. C++ plugins were also employed to ease signal processing (see Figure 11).

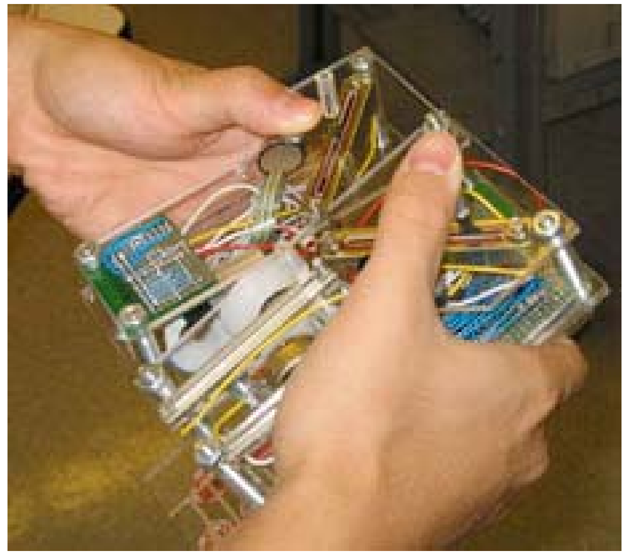


**Figure 9. The Bento-Box (150mmx150mmx45mm, 526g) Frequently used controllers are ergonomically placed.**

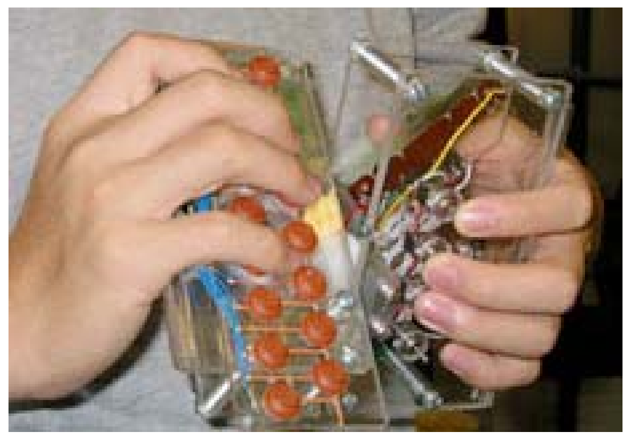


**Figure 10. Noiseless buttons for playing melody.**

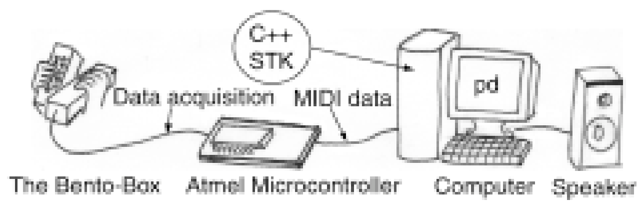



**Figure 11. System configuration**

## 5. SETUP ALTERATION

Once the entire setup was built according to the initial design, some alterations were made in the assignment of sensors to effect parameters after some experimentation. Certain sensors turned out to be ergonomically difficult to use. Some modes of control were avoided because they were confusing for the user. Furthermore, less-used functions were replaced by more desired functions for a richer musical experience.

### 5.1 Initial Setup

The initial setup was as follows (please refer to Figures 12 and 13). There were two rows of eight buttons each for the right fingers, each assigned to individual notes. The note assigned to these buttons changed depending on the style and key setups, which were selected by sliders operated by the right thumb. The style slider was located lower than the key slider, which was positioned diagonally at the top of the back right side (Back refers to the thumb side of the instrument, which faces toward the user). There is a chord selection slider, which was planned to automatically accompany the melody performed, on the left back side symmetrical to the key selection slider. The style, key, and chord changes are only to occur when one of the two small red activation buttons, installed on both right and left just below the two symmetrical sliders, are pressed. A large black button next to the activation button for the right thumb was to be a scale-reset button which enables the user to instantly change the note designation for the right finger buttons to a chromatic scale. Symmetrically on the left is an FSR for a volume effect. It was to decrease volume as it was pressed.

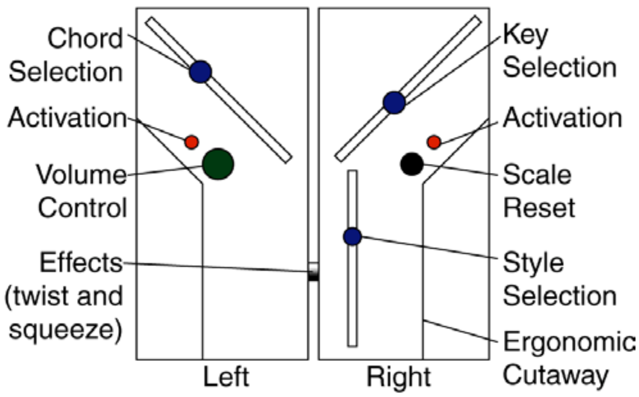



**Figure 12. Thumb (back) side schematic of the initial setup**

On the front side, there are four three-way momentary toggle switches for the left fingers. Each of these were to be assigned a sound effect, and used to select a sensor for controlling the effect; either the twist or the squeeze sensors at the center of the device.

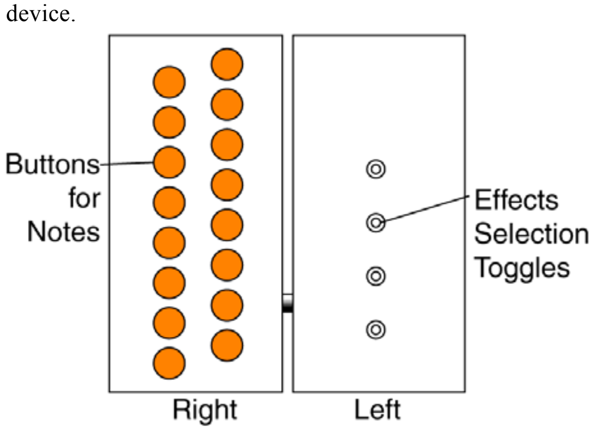



**Figure 13. Finger (front) side schematic of the initial setup**

## 5.2 Altered Setup

These four toggle switches turned out to be difficult to operate with fingers. They had to be pulled toward or pushed away from the palm, and the latter was especially difficult. The switches were also rather stiff. These switches were not employed in the final setup since it was found to be very confusing for the performer to alter the effects assigned to various sensors. Avoiding modes like this is recommended in HCI design in general [9]. The left fingers can instead be used to play notes like the right fingers by installing more buttons.

The key and chord selectors were not used during test trials. The key selector was left unused as no key change took place during performance. The chord selector was not needed since the melody buttons, controlled by the right fingers, could play chords. Instead of the initially intended applications, these two sliders were assigned to control more effect parameters.

As the result of the above changes, the two activation buttons were no longer necessary. Furthermore, the scale-reset button and the volume effect controller were never used. These input devices were not used in the final setup.

# 6. FINDINGS AND FUTURE IMPROVEMENTS

Future improvements are suggested from the experience of playing with the finalized setup. The musical setup, employing the style selection slider and numerous effect controls, was quite a success. Generally, more feedback is ideal in both physical and visual forms. Some mechanical improvement is also required for comfort and robustness. In addition, device-specific functional consideration would improve the quality of performing experience. Furthermore, there is the engineering task to make it a real portable device.

## 6.1 Feedback

Visual information feedback is necessary to show the style and key setup as well as any other parameters that may need indication. (Although the key control was eliminated in the final setup, it would still be a necessary function.) This would simply require a small LCD on the backside of the instrument.

Force feedback would be useful in some of the continuous effect controls. Passive feedback by springs might be sufficient for the application. Such a system can also help bring the effect control back to the neutral position.

The importance of tactile feedback was confirmed when operating the note buttons on the blind side of the instrument. For example, it would be very difficult to locate buttons if they were all flat. Alternatively, hand location with respect to the instrument may be realized by placing an indexing feature where the palm touches the instrument.

## 6.2 Mechanical improvement

The twisting pivot, which connected the right and left halves of the instrument, had a robustness concern. It was a rotary potentiometer and was not necessarily designed to bear load. A load isolation mechanism must be introduced to improve the structural reliability of the instrument.

The square Plexiglas construction is uncomfortable to hold. The material and the fabrication method constrained the instrument surfaces to be designed flat. Three-dimensional freeform manufacturing capability would greatly increase design freedom and the resulting ergonomics.

Sliders also had some usability problems. There was frictional resistance making the operation difficult when fast control was desired. The linear motion was also not very ergonomic for the thumbs. Instead, a curved pressure-sensitive touch-pad would be a nice replacement as there would be virtually no frictional resistance upon operation. It can also make better use of the dexterity of thumbs by added sensing dimensions: two- dimensional coordinates and vertical pressure.

## 6.3 Device-Specific Development

Currently, the various sound effects are arbitrarily assigned to the sensors on the instrument. As a result, the physical motion of the performer does not correlate with the resulting effects in any intuitive way. Hence, custom developed effects that relate intuitively to the physical actions are desired. For example, sound may be "twisted" or "squeezed" by skillful multi-speaker output system.

### 6.4 Stand-alone

Here, the system required three separate components; the Bento-Box as the controller, a micro controller for data acquisition, and a desktop computer for sound generation. It would be desirable to integrate all of it so that everything will be enclosed in a Bento-Box size package for true portability.

## 7. CONCLUSION

A prototype for a portable handheld electronic musical instrument was successfully built. Its unique style selection capability allows the user to easily perform a variety of styles of music, which would otherwise have been difficult to learn. The initial motivation was to create a personal instrument to be heard only by the performer. However, the resulting instrument, with a rich capability to control various interesting sound effects, is fit for performing out loud along with other musicians. The design was done with mockup iterations and functional and ergonomic analyses, and has led to many interesting findings. The experimentation with the completed setup has inspired further improvements.


## 8. ACKNOWLEDGMENTS

I would like to thank my project teammates Seth Nickell and Katia Zarrillo for the collaboration, the class teaching team (Professors Bill Verplank and Max Mathews, TA's Michael Gurevich and Wendy Ju) for giving me the opportunity, and Professor Mark Cutkosky and Weston Griffin for support in writing.


## 9. REFERENCES


[1] Cacha, C.A. Ergonomics and Safety in Hand Tool Design. CRC Press LCC, Florida, 1999.

[2] Cook, P.R. Synthesis ToolKit Algorithms and Instruments. http://www.cs.princeton.edu/~prc/STKAlgorithms.html

[3] Cutkosky, M.R. Robotic Grasping and Fine Manipulation. Kluwer Academic Publishers, Massachusetts, 1985.

[4] Kaplan, E.G. Functional and Surgical Anatomy of the Hand. J. B. Lippincott Company, Philadelphia and Montreal, 1965.

[5] Kelley, T., and Littman, J. The Art of Innovation. Doubleday, New York, 2001.

[6] MacKenzie, C.L. and Iberall, T. The Grasping Hand. Elsevier Science B.V., Amsterdam, 1994.

[7] Nintendo Gameboy. http://www.gameboy.com/

[8] Puckette, M. Pd download site. http://crca.ucsd.edu/~msp/software.html

[9] Raskin, J. The Humane Interface: New Directions for Designing Interactive Systems. Addison-Wesley, Massachusetts, 2000.

[10] Sony Walkman. http://www.sony.net/Products/walkman/

[11] Verplank, B., Sapp, C., and Mathews, M. A Course on Controllers. Proceedings of NIME (Seattle, Washington, USA April 2001)

[12] Waisvisz, M. "The hands, a set of remote midi-controllers," in Proc. Int. Computer Music Conf. (ICMC'85), pp. 313-318, 1985

[13] Weinberg, G., Lackner, T., and Jay, J. "The Musical Fireflies - Learning About Mathematical Patterns in Music Through Expression and Play". *Proceedings of XII Colloquium on Musical Informatics 2000*. A'quila Italy.

[14] Weinberg, G., Orth M., and Russo P. "The Embroidered Musical Ball: A Squeezable Instrument for Expressive Performance," Proceedings of CHI 2000. The Hague: ACM Press.

[15] Weinberg, G., Aimi, R., and Jennings, K. "The Beatbug Network – A Rhythmic System for Interdependent Group Collaboration." Proceedings of NIME 2002. Dublin: MLE.

[16] Wilson, Frank R. The Hand: how its use shapes the brain, language, and human culture. Pantheon Books, New York, 1998.

# Sound Kitchen: Designing a Chemically Controlled Musical Performance

Hiroko Shiraiwa, Rodrigo Segnini, and Vivian Woo
CCRMA
Department of Music, Stanford University
Stanford, CA 94305-8180
{hiroko,rsegnini,vwoo}@ccrma.stanford.edu

## ABSTRACT

This paper presents a novel use of a chemical experiments' framework as a control layer and sound source in a concert situation. Signal fluctuations from electrolytic batteries made out of household chemicals, and acoustic samples obtained from an acid/base reaction are used for musical purposes beyond the standard data sonification role. The batteries are controlled in handy ways such as warming, stirring and pouring that are also visually engaging. Audio mappings include synthetic and sampled sounds completing a recipe that concocts a live performance of computer music.

## Keywords

Chemical music, Applied chemistry, Battery Controller.

## 1. INTRODUCTION

In data sonification[1], sound is the medium to display a process that generates data. While aesthetic principles may be used in the design of a sonification interface, the scope usually remains in the observation of data. The method and sounds selected have a limited role compared with the process at hand. In our project, sound is the goal itself, and the process from which it derives is designed to have a musical and visual outcome deliverable through a live performance. The wide range of chemical reactions allows for the realization that some can be appropriate for musical purposes. To explore this idea a number of household chemicals were considered for their potential in producing interesting visual and acoustic effects while providing signals used to drive a software sound engine.

### 1.1 Chemical Selection Criteria

A chemical process was defined as worthwhile pursuing if it had any combination of the following: controllability, availability, safety of use, visually engaging, wide and predictable dynamic range, or some degree of concordance between expectations about the process and the sound it controls/produces. Another concern was its matching to certain basic time gestures already considered part of the musical outcome: instantaneous changes, as well as slow build ups should be in the 'recipe'. After evaluating a number of elementary processes like acid/base reactions and electrolytic batteries, next came sorting through a list of potential 'ingredients' that would pass the criteria above. Thus colorful orange juice and red wine were chosen as acidic alternatives to the more powerful but transparent vinegar. Direct heating of water gained over the slower temperature build-up of yeast, water and sugar. The stirring of a liquid in a container remained as a must-do for the applicability of radial motion as a control for audio system playback rate. More hardcore chemicals were discarded, despite their potential for great effects, due to safety concerns.

### 1.2 System

Sound Kitchen uses three variations of a electrolytic battery: a stirring, a pouring and a thermally controlled battery. The three batteries generate voltage signals with variations in time, and are easy to manipulate in front of an audience. The voltage signals are captured and conditioned through a microcontroller and sent to a PC where they are mapped to musical parameters. The system also has four test tubes with acid and base. The fizzing sounds of an acid with base reaction are captured by microphones and used directly as sound source (Figure 1).

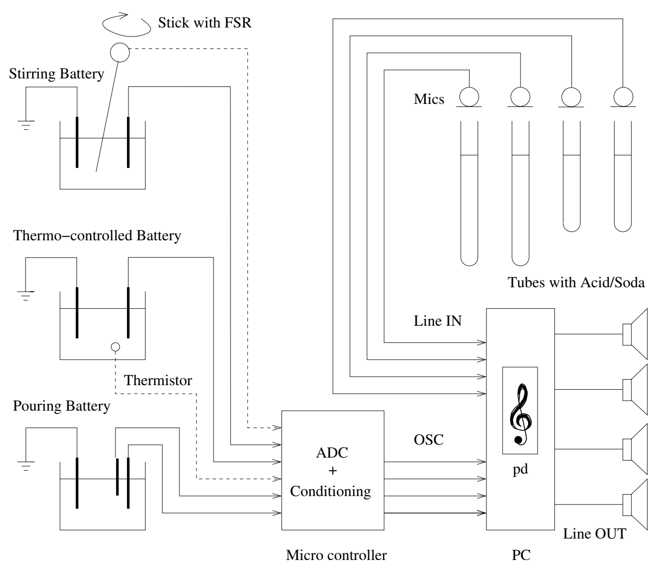



**Figure 1: System overview**

This work was done as a final project for the course "Music 250A: Human Computer Interaction Theory and Practice: Designing New Devices"[2] (instructor: Max Mathews and Bill Verplank) at CCRMA in Stanford University.

## 2. HOUSEHOLD CHEMISTRY

### 2.1 Baking soda and vinegar

Combining 10 ml of vinegar (an acid) and 2g of baking soda (a base) results in an acid/base reaction with fizzing sounds inside a test tube. The products of all acid/base reactions are a salt, water, and carbon dioxide gas:

$$Acid + Base \longrightarrow salt + water + carbon\ dioxide\ gas$$

The specific reaction in this system is:

$$CH_3COOH + NaHCO_3 \longrightarrow NaC_2H_3O_2 + H_2CO_3$$

where $CH_3COOH$ is acetic acid, $NaHCO_3$ is baking soda and $NaC_2H_3O_2$ is sodium acetate. This is followed by

$$H_2CO_3 \longrightarrow H_2O + CO_2$$

where $CO_2$ is a gas, dispersed as bubbles into the atmosphere. This part of the reaction causes a fizzing sound which is captured by microphones and processed as an acoustic signal.

## 2.2 Electrolytes in Batteries

A voltage can be created when two polarized electrodes, e.g. copper and aluminum strips, are inserted in an electrolytic substance. The negative ions in the electrolyte are attracted to the aluminum strip, while positively charged ones are attracted to the copper strip. Temperature increase/decrease and electrolyte volume are used to control the total charge of the system.

### *Citric Acid*

Citric acid, a prominent component of orange juice is represented by the chemical formula $C_6H_8O_7 \cdot H_2O$. When the copper and aluminum (the electrodes) are submerged in the orange juice (the electrolyte), the hydronium ions ($H_3O^+$) of the orange juice are attracted to the copper electrode and the $C_6H_7O_7^-$ ions are attracted to the aluminum electrode (Figure 2). 40 ml of orange juice are used to charge the stirring battery.

### *Red Wine*

Red wine may contain three acid derivatives: benzoic acid derivatives, cinnamic acid derivatives, or flavonoid derivatives (Figure 3). This substance works much like that of orange juice except that the $C_6H_7O_7^-$ is replaced by one of the acid derivatives of the wine. 400 ml of water and 100 ml of red wine are used to charge the pouring battery.

### *Temperature control*

In this system protons ($H^+$) and hydroxide ions ($OH^-$) are the attaching elements. After heating around 60 degrees Celsius, the electrolytic properties of water become more intense, therefore more interesting as a slow paced control signal. Heat increases the agitation of the ions in the electrolyte thereby increasing the flow of the electrons (Figure 4). Our thermo-controlled battery uses a tea pot coil warmer to heat 250 ml of water.

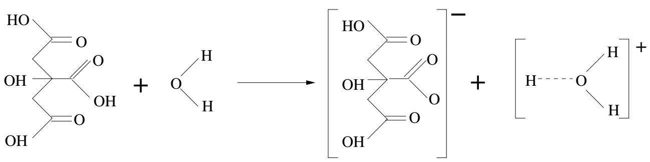



**Figure 2: Citric acid gives up one hydrogen ion to a water molecule resulting in negatively and positively charged molecules**

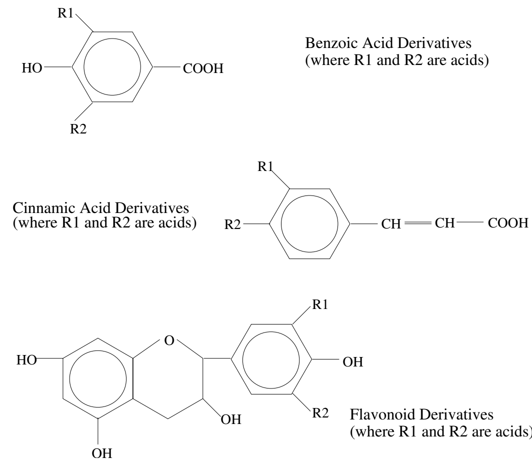



**Figure 3: Three acid derivatives that occur in red wines - benzoic acid derivatives, cinnamic acid derivatives, and flavonoid derivatives**

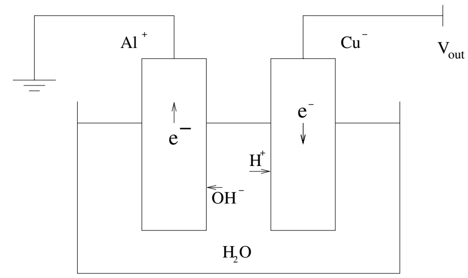



**Figure 4: The water battery system where water is the electrolyte and a copper strip and an aluminum strip act as electrodes**

# 3. ENGINEERING

An ATMega163 controller[3] is used to capture, condition and transmit signals from sensors in the chemical framework. AVRGCC is used for coding. Out of six input signals to the ATMega163, four output signals are sent to a PC after conditioning via the serial port using Open Sound Control (OSC) [4].

## 3.1 Tubes and Microphones

Four small microphones are used to capture the sounds of the reaction of acid and soda in the test tubes. Signals are sent directly to four loud speakers via a spatialization patch in Pd [5], e.g. the first tube to the front right speaker, the second to the front left, the third to the rear right, the fourth to the rear left. Before performance, the portion of vinegar and soda is carefully chosen not to damage the microphones with foamy acidic bubbles. The soldered parts of microphones are coated with hot glue, to prevent rust.

## 3.2 Batteries

### *Stirring Battery*

The Stirring Battery exhibits a chaotic voltage flow when the electrolyte is stirred with a circular motion. For musical reasons, this is turned on and off in a controlled way. An FSR (Force Sensitive Resistor) attached to the stirring stick implements this switching functionality: raw voltage signal when the stick is held, or zeros when not held (Figure 5). Both signals from the FSR and the battery are sent to the micro-controller where the flow control switching takes place.

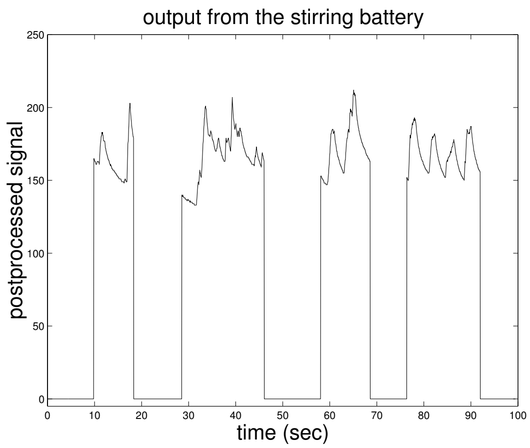



**Figure 5: Example output from the Stirring Battery showing voluntary on and off switching at four instances. Note that fast transients occur during stirring.**

### *Pouring Battery*

The Pouring Battery has two stages during the performance: initially it is filled with plain water; then as some red wine is added, more ions charge the liquid, increasing the voltage output from the battery. By pouring the liquid into a different container, and back into to the battery container this voltage fluctuates (Figure 6). Two copper strips are set in the battery with different but coherent signal outputs. One of them is sent to a Non-Inverting amplifier, made with an Op-amp, as to provide more dynamic range to the signal.

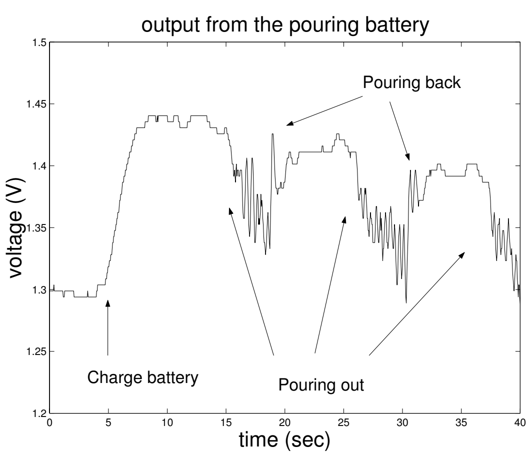



**Figure 6: Example output from the pouring battery showing voltage fluctuations as wine is added and the electrolyte is poured in/out the battery container**

### *Thermo-Controlled Battery*

The Thermo-controlled Battery generates a voltage $v_T$ which increases roughly proportional to the water temperature $T$ within the approximate range of $60 < T < 100$ degrees Celsius. During the performance a coil heater, cold water and ice are used to control temperatures. A thermistor is also used to set a temperature threshold around the boiling point. Both signals from the thermistor and the battery $v_T$ are sent to the micro-controller where the flow control takes place (Figure 7).

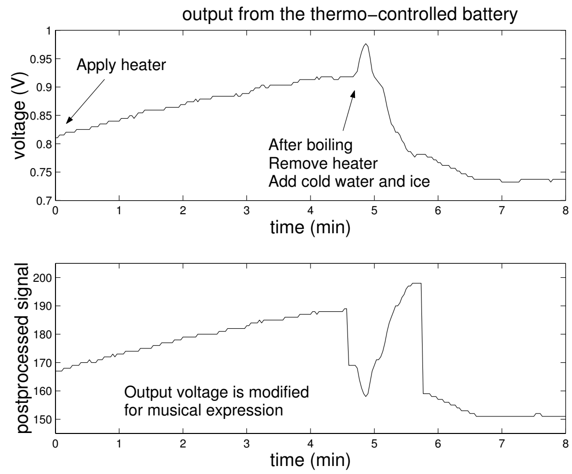



**Figure 7: Signal from the thermo-controlled battery showing slow time transient during 8 minutes. Top : raw voltage output, Bottom : postprocessed signal**

## 4. MUSIC AND PERFORMANCE ASPECTS

The metaphor used when composing music for this project was that of cooking. When preparing a dish, adding separate ingredients following a recipe results in a whole that is more than the sum of its parts. In our framework every chemical reaction contributes to the music as 'composed' on a score designed for live performance.

Since this composition is to be performed live, Pd was chosen as the sound engine where the incoming signals are translated into music.

### 4.1 Signal Mapping

Overall 4 input signals (signals 1 - 4) arrive into the Pd patch via the serial port using OSC. Additionally, 4 microphone signals (acoustic signals 1 - 4) are fed via the audio card and entered into Pd using 4 channels in the adc~ object (analog-to-digital converter). The output of the whole patch is sent out through the audio card using 4 channels in the dac~ object (digital-to-analog converter).

### *Signal 1: Multi-rate Sampler*

Signal 1 is derived from the Stirring Battery and used to control the playback sample rate of two speech samples. While the stick is held, signals pass through, otherwise the input is shut down. When a signal is present, the Pd patch responds by fading in the voice samples, and depending on the motion achieved inside the container, the sample rate for playback varies. Based on tested behavior of this battery, whenever a critical motion speed is achieved inside the container, the signal value raises, otherwise it settles and decays. Because of the non-linearity of the chemical process, it is difficult to achieve the "sweet spot" where both voices are intelligible, otherwise sounding too slow and too fast, respectively. It is used after the climax in the composition.

### *Signal 2: BoiliBass*

Signal 2 comes from the Thermo-Controlled Battery. It raises continuously until the boiling point is reached, at which point it shuts down. It controls a simple additive synthesis patch of two sounds with 4 partials each. The base frequency of one sound is equal to $f1 = x$, while that of the other is $f2 = x - (x/5)$, where $x$ is the signal value. These two sounds produce a beating effect, which increases as both values diverge. This signal is used throughout the piece, providing its overall form (Figure 7, the bottom part) with the top value achieved at $\frac{2}{3}$ s of the duration of the piece, and the remainder $\frac{1}{3}$ showing a different behavior during cooling down.

### *Signal 3 + 4: Morning Bird & Meshscape*

Signals 3 and 4 are derived from the Pouring Battery. It controls a patch producing 50 ms bursts of an additive synthesis array whose bandwidth, base frequency and preeminence of each partial's weight depend on various scaling of signals 3 and 4. It provides what is perceived as a background bird singing sound which tends to be stronger when the quantity and motion of electrolyte inside the battery is higher, that is, during pouring, with an immediate decay in presence (lower pitch height and intensity).

This signal also controls an additive synthesis and frequency modulation patch. Volume is controlled by slow transient envelope taking up to 30 seconds to peak. The additive synthesis patch produces a mesh of 16 partials whose weight is scaled by three oscillators which also modulate the frequency of 3 other partials. The rhythm achieved with the liquid pouring influences the rate of oscillation, inducing higher sidebands that appear to match the movement of the performer.

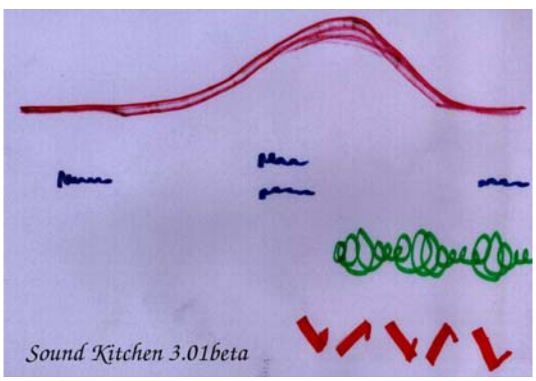



**Figure 8: The score: its duration is 3 to 8 minutes depending on the start temperature of water. Drawings indicate location of group events in a timeline.**

### *Acoustic Signals 1 - 4*

The direct sound of baking soda diluted in vinegar inside test tubes of different sizes is sent across the room in a 4-channel spatialization setup controlled via a pd patch. This process has a short duration and defines the start point every time a scoop of baking soda is added to the tube. It appears along the beginning, reaching the climax, and at the end of the piece.

## 4.2 Score and Performance

The system was tested in December 2002 with a composition for which a simple score (Figure 8) was designed. The three authors participated as performers stirring juice, pouring water + wine in the batteries, baking soda to tubes, and otherwise, supervising that the whole system responded as expected throughout the performance. With practice, a single performer may be sufficient for the whole act.

As to not affect the impression that the visual performance is integrally bounded to the output sound, no manipulation of the Pd patch, nor the microcontroller board is required (except for a few start/stop instructions throughout the performance)

While this system was designed with an specific outcome in mind, there are few constraints for it to be used as a more general performance system. Most of the limitations are posed by the choice of audio synthesis and parameters in the design of the Pd patch, and the capture/conditioning stage in the microcontroller. This choices are completely arbitrary, and respond mostly to the initial compositional idea. A different user with Pd and avrgcc programming experience could modify this framework with a minimum effort, but that also brings the question of whether it would not be more effective to build a different one from scratch.

# 5. CONCLUSION AND FUTURE WORKS

The exploration of music creation with chemical reactions and manipulations is a novel topic in the field of computer music. Through simple chemical experimentations, analog circuitry, computer manipulation, and human control, a new way of creating music has been explored. Since making music using chemicals has not been vastly explored, there remains great potential for further improvement in this area. One method of improvement would be to implement some sort of real time automatic normalization of the chemical signals. Currently the system has to be tuned to prevailing conditions (ambient temperature, electrolyte mixture) before performance. Furthermore, it is possible to explore other chemicals, reactions, and control methods that can allow for other musically inspiring signals and performance situations. Finally, this project is truly a testament of the integration of diverse fields of study to create new music.

# 6. ACKNOWLEDGMENTS

We would like to thank Max Mathews, Bill Verplank, Michael Gurevich and Wendy Ju for their advice and support.

# 7. REFERENCES

[1] *Kramer et al:* Sonification Report: Status of the Field and Research Agenda. Prepared for the National Science Foundation by members of the International Community for Auditory Display. (1996) Available online at http://www.icad.org/websiteV2.0/References/nsf.html

[2] CCRMA, Music 250A home page: http://www-ccrma.stanford.edu/courses/250a/

[3] Atmel AVR home page: http://www.atmel.com/atmel/products/prod23.htm

[4] OSC is a protocol for communication among computers, sound synthesizers and other multimedia devices that is optimized for modern networking technology. OSC home page: http://cnmat.cnmat.berkeley.edu/OSC/

[5] Pd (Pure Data) is a real-time graphical programming environment for live interactive computer music. Pd works on SGI machines, Microsoft Windows, Linux, and Mac OSX. Pd home page: http://crca.ucsd.edu/ msp/software.html

# TGarden: Wearable Instruments and Augmented Physicality

Joel Ryan
STEIM
Achtergracht 19
1017WL Amsterdam
011-31-206243886
jr@xs4all.nl

Christopher Salter
Sponge
Lausitzer Strasse 6
10999 Berlin
011-49-30-814 92 898
csalter@gmx.net

## ABSTRACT
This report details work on the interdisciplinary media project TGarden. The authors discuss the challenges encountered while developing a responsive musical environment for the general public involving wearable, sensor-integrated clothing as the central interface and input device. The project's dramaturgical and technical/implementation background are detailed to provide a framework for the creation of a responsive hardware and software system that reinforces a tangible relationship between the participant's improvised movement and musical response. Finally, the authors take into consideration testing scenarios gathered from public prototypes in two European locales in 2001 to evaluate user experience of the system.

## Keywords
Gesture, interaction, embodied action, enaction, physical model, responsive environment, interactive musical systems, affordance, interface, phenomenology, energy, kinetics, time constant, induced ballistics, wearable computing, accelerometer, audience participation, dynamical system, dynamic compliance, effort, wearable instrument, augmented physicality.

## 1. INTRODUCTION
We report on work done for TGarden, an experimental responsive media environment where small groups of participants from the general public can control and play with real time generated sound and image through improvised movement and gesture. Development on Phase 1 of the project took place during 2000-2001 with support from the Daniel Langlois Foundation and was shown as a work in progress at the Ars Electronica Festival and at V2/Las Palmas in Rotterdam for the European Cultural Capital of the Year in the fall of 2001 [6].

Our focus in this report lies on issues arising in the process of designing a physically responsive musical system activated by the motion and gestures of non-experts, where no predetermined, a priori representation of gesture can be said to exist.

While so-called "audience participation" installations are beginning to take these issues into account, there has still been little work to date, at either the conceptual or technical-implementation levels on how to build a responsive system that is physically engaging *and* learnable within a short period of time *and* musically rich and coherent for the casual, non-expert participant [9].

While literature in the field of gesture-activated musical interaction is well established and voluminous, most of this work has focused on systems designed for trained and expert performers, dancers and musicians, where issues of musical (and movement) nuance, control and expression are assumed from the start. Furthermore, much of this literature assumes traditional performer/spectator relationships, where the behavior of an interactive system is experienced passively by a viewer/listener at a distance.

The work described here focuses less on the specifics of the hardware and software layers in TGarden but rather suggests a novel approach to the total design of a responsive musical system. This system is architected to create a coherent and felt resonance between multiple layers: a participant's improvised movement, sensor input, software and the resulting musical response.

## 2. PROJECT BACKGROUND
TGarden is a responsive environment where visitors can shape media around them through improvised gesture, play and social interaction. The project investigates how people individually and collectively interact with and make sense of a responsive media space by articulating their knowledge non-verbally. More specifically, the project examines the process of meaning making as a complex relationship between the production of gesture and movement and the resulting audio/visual responses that ensue from solo as well as group motion and action.

The TGarden setting consists of a series of private dressing rooms leading to a physically demarcated, curtained off performance space for between two to five visitors over a defined time cycle. The performance area is a large (approximately 12 x 14 m) environment, with real time generated video projected onto floor of the room and multi-channel sound, the number of speakers determined by the number of participants. Besides a number of oversized balloon-like balls that help to catalyze participant's movement and spatial exploration, the performance space is empty. Participants are informed of the time frame set for their play period before they enter the event.

Outside of the actual participants, there are no spectators allowed to watch from the sidelines.

After making a reservation for a specified time slot, visitors arrive at the space and are presented with an array of specially designed garments to choose and subsequently change into. Each of the garments is designed with unusual materials (plastic tubing, Styrofoam balls, springs and wire) and appears at first to be more sculptural than wearable. Additionally, the exaggerated weight, scale and dimension of the garments acts to augment and shift a participant's normal, everyday way of moving.

**Figure 1. TGarden costumes, Ars Electronica festival 2001.**

Loosely attached to some of the garments and woven into others are several ADXL202 2G accelerometers that can measure the degree of acceleration-from small arm gestures to larger, full body movements-of each participant.

Continuous data streams from the accelerometers are transmitted from each of the individual participants by way of a wearable, commercial COMPAQ iPAQ handheld computer running LINUX and sent, via 802.11b to a central dynamics system that forms the core of the TGarden system architecture. This dynamics model, written as a custom C extension to the Max programming environment, models the states of the players and of the entire TGarden performance space as a set of continuous dynamical physical systems.

While a technical description of the system is beyond the scope of this report, the software looks at overall energy input from the multiple data streams, interprets data from the sensors and analyzes what is happening in the environment overall, subsequently sending commands to the sound and video systems based on its judgment. The dynamics system contains the microscopic logic of how the environment responds to visitors' actions, both immediately as well as over time. The different sound and image systems modify their own internal states on the basis of the dynamic system's hints and also on the basis of the continuous output from the sensors themselves [5].

From a user perspective, TGarden is conceived to be an emotionally engaging event, aiming to create a tangibly experienced dynamic between the environment's physical and computational layers. Metaphoric but highly physical notions of digging, excavating, camouflaging, unearthing, marking and skrying form a dramaturgical context for the overall design of the media layers, from music and image to architectural space, "props" and garments.

Furthermore, TGarden is grounded in an explicitly phenomenological approach that sees the production of human agency and meaning making as a process of embodied action, made explicit or *enacted* through improvised, "on the fly" gestures and movements that spontaneously arise from the participants' interaction with each other as well as the proximal environment itself [7].

While the process of participants' improvising or "performing" for each other without a pre-determined script is constrained by the specific "framing" of the event (set length of time of engagement, numbers of simultaneous participants, architectural configuration of the space, ritual of dressing and undressing, the exaggerated bodies of the costumes themselves), TGarden's constraints are most tactilely conveyed through the wearable interface of the garments themselves. Here, the garments function as both an *affordance*-where the participant's gestures and movements are shaped, limited and expanded by the inherent physical attributes and properties of the textiles-and, simultaneously as a wearable *instrument*. Indeed, much of the design of responsiveness in TGarden is based on the vision of a first encounter with a musical instrument. As every novice discovers, a new instrument has constraints where "not everything is possible" as he/she quickly learns to grasp at the instrument's physical affordances to enable the production of sound. It is this approach that places TGarden in a long lineage of gesture-based instrument design that explicitly takes the creation of tactile affordances into account [2].

**Figure 2. TGarden group interaction, Rotterdam 2001.**

# 3. RESPONSIVITY

Given the explicit dramaturgical and interface context of TGarden, we were faced with devising a responsive musical system in hardware/software which could accommodate a wide range of movement possibilities from non-trained participants, while conveying an integrated relationship between music, image, garments and social interaction itself.

## 3.1 From sensing to music

In designing a musical software layer, we began with the understanding that participants interact with music in the TGarden world not as an extraneous object which a person is coached into accepting or "learning," but rather through processes of dancing and "marking" - dancing, in that music arises from a moving body and marking, in that the relation to music is rather than mere triggering, one of inflecting and shaping sonic material whatever its provenance. Joel Ryan's comment that "the physicality of the performance interface gives definition to the (musical) modeling process itself," suggests that there must be an equal physicality between the interface (i.e., sensors) and the software layer [1,3].

In choosing body-mounted sensors as the system's sole input devices, we looked for technologies that were sensitive to the widest possible range of bodily movement. While a host of sensors systems, from the BioMuse that monitors electrical charges in the muscular system, to bend sensors that measure relative position of limbs are available, these systems were passed over due to the amount of control required on the part of the participant and their relatively limited dynamic range [8].

The use of accelerometers to measure degrees of bodily force is a technique that has been previously deployed for responsive musical systems. Indeed, as Sawada, Onoe, and Hashimoto incisively point out in their article "Acceleration sensor as an input device for musical environment," "although most of the reported works to introduce body movement into musical performance treat the shape or position of the body, the most important emotional information in human gestures seems to appear in the forces applied to the body" [4].

Because TGarden participants experience the garments as an instrument/interface, however, we felt the need for an augmented physicality in the software layers of the system. In other words, TGarden instruments begin with accelerometer data, but the responsiveness of the system is built on simulated physical behavior.

**Figure 3. TGarden Rotterdam 2001.**

In the TGarden instruments, players connect to higher-level musical parameters like pitch and beat not directly but via an additional layer of simulated physics written in SuperCollider. This physics, by adding elastic couplings and phantom masses, induces distinctive ballistic behavior in the response of the system. A player listening to the sonic feedback of his/her movement, while perhaps not being able to decode their relation to some intricately woven algorithm or mapping, easily comes to identify wobbles and bounces, recoils and lags as their own; as their personal marking of the sound. This is clearly related to our encounters with real instruments. At any one time in the TGarden system, there are several physical models in action simultaneously, some based on simulated kinetics, some on energy. Each has a set of time constants that can be tuned to bring players closer to the musical time of a phase of their physical play within the environment. This characterization in terms of time constants is perhaps the most direct link between TGarden's system design and its compositional thinking. Though we use a variety of musical sound generating methods, all share this augmented physicality.

The system and player are in a dynamic compliance that varies as the play evolves from state to state. The energy of the players is the primary input that drives the large-scale behavior of the environment. This includes the density and angular momentum of groups and the averaged energies of individuals at several time scales. A great deal of effort has been put into recovering the reduced physicality of a simple, sensor-based interface through a simulated physics, not to create phantom instruments, but because we believe that embodiment and immediacy is as much the medium of music as intentions drawn from musical experience. This is central to reinforcing TGarden's overall dramaturgy of "felt" physicality. In addition to giving us a way of achieving musical coherence, we found that providing the players with an exaggerated body by drawing attention to ballistics helped overcome resistance to enthusiastic physical exploration and play.

## 4. USER TESTING/USABILITY

During TGarden's public showings at the Ars Electronica festival in Linz and in Rotterdam in the Fall of 2001, the development/production team conducted extensive user testing and usability studies of the prototype system. In over 250 videotaped user interviews, participants were asked to evaluate the quality of their experience based on a series of questions asked by team members. The wide range of ages, body types and overall responses to the experience suggest that our early attempts at engendering a dynamic coupling between garments, participant movement and musical response were partially successful, if not sometimes too complex for the general audience.

### 4.1 System response

In general, participants focused primarily on (1) the immediacy of the feedback between gesture and sound as well as (2) the ability to sonically distinguish one participant from another. Indeed, the ability or inability to "parse" out individual sonic agency and distinguish "who was who" was a clear issue for the majority of visitors. The degree of this concern, however, varied widely among participants. Some groups sorted agency-feedback issues out in a round robin manner, letting each individual participant "solo" their own movement and the resulting sounds before beginning any group-related play. Individuals who detailed less successful experiences sometimes failed to acknowledge the necessity of concentration and listening to what other participants were doing.

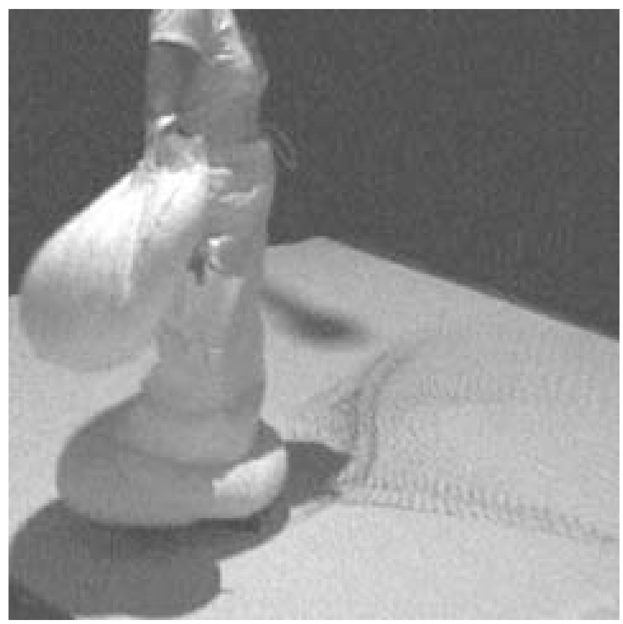


**Figure 4. TGarden participant, Ars Electronica festival, 2001**

While many users commented positively on both the richness and experiential quality of the timbral palette (finding a clear relationship between music, garments, images and the total theatricality of the event) some complained that the sonic complexity of the synthesis models overwhelmed their ability to make sense of the response characteristics. Further, some users grew tired of the system's inability to rapidly change musical/sonic material or to easily find the energy thresholds to make the musical system vary past its initial conditions. This comment, although perceptive, was expected based on lack of adequate time in the software development period, resulting in a limited sonic palette.

### 4.2 Temporal engagement

Time clearly emerged as a critical issue in both sets of prototypes, demonstrating that the amount of time spent inside the environment had a crucial impact on the overall user experience. Some users, perhaps conditioned by other interactive experiences expected an immediate "return" on their time investment, wanting to discover all of the system's response "tricks" almost immediately. Some participants revealed a clear sense of frustration with an experience that required so much time commitment overall. The number of simultaneous visitors within the environment also clearly impacted how long participants were willing to remain in the performance space. While system problems during the rehearsal period in Linz forced the team to cut back on the number of simultaneous participants, visitors in Rotterdam were able to interact with up to six other players. Groups in Rotterdam occasionally remained inside the main room for more than 45 minutes, finding it hard to leave. Some users commented on a complete loss of any sense of time whatsoever, particularly since they were faced with so much perceptual overload between the garments, media and social interaction.

### 4.3 Sensing: physicality and robustness

The use of accelerometers was confirmed to be an appropriate choice yet also revealed unexpected usability and robustness issues. What was immediately apparent was that user experienced hinged on the degree of willingness to explore multiple movement and gesture possibilities as well as different attitudes to bodily acceleration and force. Many visitors quickly discovered the clothing/sensor relationship, playfully exploring a wide variety of movements and gestures to see what the system was capable of. Despite being informed beforehand, some visitors were confused by the idea of a strictly body-based interface, instead moving about the performance space in search of static, environmental sensing. Other users were confused, in general, about what movements to make. Visitors who remained stationary and reserved in their movement exploration were naturally disappointed by the lack of response and mentioned that they quickly grew tired of the experience.

Still other participants remarked on the physical difficulty of setting the system in motion, some even complaining of physical exhaustion during the experience. While such observations certainly reinforce Joel Ryan's comment in his article "Effort and Expression" that "in designing a new instrument, it might be just as interesting to make playing it as difficult as possible," it is clear that TGarden assumes (perhaps too much) an inherent willingness on the part of the participants to explore potential extremes of bodily physicality without knowing a priori what results will occur [3].

Finally, while the integration of accelerometers and the induced ballistics of the costumes worked to a degree, the extraordinary and unexpected abuse wrought by the participants resulted in frequent accelerometer and connector breakage as well as some sensors coming loose from their costumed housing/material. This resulted in the impossible situation of some users trying to excite the system to no avail due to the loss of appropriate axis calibration of the accelerometers. While team members were on hand to handle such breakages, it is clear that abuse from a general audience must seriously be taken into account in the next phase of clothing and sensor design and integration.

## 5. CONCLUSION

We have briefly described work done on TGarden, an interdisciplinary responsive environment for the general public. It should now be evident that the complexity of designing interfaces and musical responsivity which is both intuitive and experientially felt for non-expert users is a daunting task that challenges conventions in the field of gesture-based musical interaction. The success and feedback of the public prototypes in Linz and Rotterdam clearly point the way for further exploration into new kinds of body-based responsive musical environments for the general public as well as providing critical lessons for our ongoing research into these areas. This information is invaluable as we begin work on Phase 3 of the project under the support of the Rockefeller and LEF Foundations, for public showing in North America in late 2004.

## 6. ACKNOWLEDGEMENTS

We would like to thank our collegues and collaborators on the 2001 TGarden team, particularly Sha Xin Wei, Maja Kuzmanovic, Laura Farabough, Nik Gaffney, Yon Visell, Evelina Kusaite, Cocky Eeck, Yifan Shi and other members of the Sponge/FoAM network.

We also wish to acknowledge the financial and philosophical support of the Daniel Langlois Foundation, the Ars Electronica Center, the Banff Center for the Arts, the Graphics, Visualization and Usability Center at the Georgia Institute of Technology and the V2 Institute for Unstable Media.

## 7. REFERENCES

[1] Lewis, G., et al. Discussion in Array: Communications of the ICMA, Volume 13, Winter 1993.

[2] Ryan, J. Waisvisz, M. Collins, N. "STEIM - de zoetgevooisde Bliksem,'"(chief editor: S. Sjollema), STEIM 1993 .

[3] Ryan, J. "Effort and Expression: Some Notes on Instrument Design at STEIM." In *Proceedings of the 1992 International Computer Music Conference*, ed., A. Strange. San Francisco: Computer Music Association, 1992.

[4] Sawada, H, Onoe, N. and Hashimoto, S. "Acceleration sensor as an input device for musical environment," In Proceedings of the. International Computer Music Conference. (ICMC'96), pp. 421-424, 1996.

[5] Sha, X.W., Macintyre, B. and Visell, Y. "TGarden Media Choreography System." Submitted ACM SIGGRAPH 2003.

[6] Sponge. http://www.sponge.org/projects/m3//tgarden. FoAM: http://www.fo.am/tgarden.

[7] Varela, F., Thompson, E., and Rosch, E. The Embodied Mind: Cognitive Science and Human Experience. Cambridge: MIT Press, 1991, pp.172-179.

[8] Winkler, T. "Making Motion Musical: Gesture Mapping Strategies for Interactive Computer Music." In Proceedings for the 1995 International Computer Music Conference. San Francisco, CA. Computer Music Association, 1995.

[9] Winkler, T. "Audience Participation and Response in Movement-Sensing Installations." In Proceedings for 2000 International Computer Music Conference, Berlin, Germany. Computer Music Association, 2000.

# Duet Musical Companion: Improvisational Interfaces for Children

David Ventura
Carnegie Mellon University
Entertainment Technology Center / ATR MIS Labs
5000 Forbes Avenue, 4301 Doherty Hall
Pittsburgh, PA 15213 USA
1-412-268-5351
dventura@cmu.edu

Kenji Mase
ATR MIS Laboratories / Nagoya University
2-2-2, Hikaridai
Seika-cho, Soraku-gun
Kyoto 619-0288 Japan
81-774-95-1440
mase@atr.co.jp

## ABSTRACT
We present a sensor-doll interface as a musical outlet for personal expression. A doll serves the dual role of being both an expressive agent and a playmate by allowing solo and accompanied performance. An internal computer and sensor system allow the doll to receive input from the user and its surroundings, and then respond accordingly with musical feedback. Sets of musical timbres and melodies may be changed by presenting the doll with a series of themed cloth hats, each suggesting a different style of play. The doll may perform by itself and play a number of melodies, or it may collaborate with the user when its limbs are squeezed or bent. Shared play is further encouraged by a basic set of aural tones mimicking conversation.

## Keywords
Musical improvisation, toy interface agent, sensor doll, context awareness.

## 1. INTRODUCTION
We have developed a musical entertainment system for a context-aware sensor doll with the aim of creating a responsive toy companion for children. By encouraging children to play with sound and make-believe, we hope to promote both learning and personal expression. The interface chosen is a bear for its traditional appeal to many ages, as well as having a set of familiar parts for tactile stimulation. The bear supports two roles: as an instrument for solo interaction, and a companion for collaborative performance. This duality abstractly illustrates the principles of both melody and harmony in experimentation. Feedback to physical contact is issued purely in the aural sense by chords, riffs, melodies and harmonic utterances. Varied play is encouraged through different timbre palettes associated with a series of cloth hats that trigger mode changes in the bear. Our overall objective is to provide an entertaining musical toy for children to play and experiment with.

## 2. RELATED WORK
The area of technologically augmented physical companions is a relatively new field as the required hardware begins to become affordable and compact-enough to place inside of children's playthings. There has been a growing offering in recent years both for commercial and research sectors. Embryonics' Munchkin produced an augmented stuffed animal Mozart Magic Orchestra Featuring Wolfgang that performs an instrument patch when one of its paws is squeezed [3]. Learning Curve's Lamaze has a similar doll Octotunes that plays a different note for a touch on each of its tentacles [5].

**Figure 1. Bear dolls and themed hats.**

Strommen and Alexander report on educational and emotional factors for interactive dolls such as ActiMates' Barney [8, 9] and Arthur [1]. There has also been interesting work recently examining methods for adolescent musical expression and learning. Browall and Lindquist have studied children's play and cooperation to develop a multi-user toy sound mixer [2]. Payling created a set of augmented blocks for children to provide an alternate interface for musical expression [6]. Gan studies a similar application but with particular focus to tactile feedback [4].

Our work leverages the hardware and principles of the com-music sensor doll developed by Yonezawa et. al [10] and Kazuyuki et. al [7]. In form, function and expressive capabilities our system is most similar, but we expand on the previous work by adding basic computer vision and combining it with emotional and design factors for children, much like methods described for the work above.

## 3. ENVIRONMENT
### 3.1 Design Issues
When constructing the interaction suite, there were a number of points considered that controlled the scope and design of the project.

#### *3.1.1 Personality and Autonomity*
To adequately serve the purposes of a play *thing* and a play *mate*, we added enough functional constraint to enforce basic musical principles and themes. Also, great amounts personality or intelligence were not invested to prevent the expectation of an autonomous figure. The doll does not speak,

but it does respond with musical murmurings if neglected or treated harshly. Input for the doll is received directly from physical contact with various parts of the body, and indirectly through a USB camera in its nose that processes images of the environment. The doll may play discrete audio samples and effects, or perform a backing track for the player to collaborate with. Aside from this simple feedback, however, much of the actual play is required from the child.

To add to the doll's charm, subtle characterizations are embedded. We have produced two dolls: first, a brown bear, designated as a boy, warbles and sings in a deep and gruff tone, secondly we have a more colorful blue bear, which has a higher pitched and more feminine vocal timbre. Both, however, contain the same melodies for play and respond vocally to stimuli such as pulling on their mouths.

### *3.1.2 Props and Color Choice*

Each bear may play four different "games", where a game is defined as six complementary samples and a melody. Musical themes associated with the games are linked to the hat each bear uses to activate them. A further degree of iconization is encapsulated in the hat's color. Using basic color psychology principles, different adolescent emotional archetypes are bound to the music. A baseball cap signifies sports, so it triggers patches associated with a brass band and a tune resembling "Take Me Out to the Ballgame". The hat is colored red, which commonly is associated with aggression and excitability. A nightcap is connected to a lilting lullaby and soporific, long decaying major chords of wind instruments. The nightcap is made from quilted, yellow fabric, suggesting infancy and a strong dependence on parental protection. With a painter's beret, the bear supports creative expression, inner reflection and emotional balance. The aural scheme consists of a tonally balanced set of string instruments and a meditative melody. Lastly, a blue knight's helmet formed in the shape of a dragon's head is used to represent daydreaming and imagination, and emotional maturity. A wistful celestial song is employed in conjunction with a set of surreal warm pads and bells. With these separations and groupings, we unobtrusively reinforce the connections and roles between several levels of music and the subconscious. (See table 1.)

**Table 1. Themed color and musical separations**

| Hat Type | Color | Audio Patches |
|---|---|---|
| Baseball cap | Red | trumpet, tuba, drums |
| Nightcap | Yellow | flutes, air vocals |
| Painter's beret | Green | violin, cello |
| Knight's helmet | Blue | pads, bells |

# 4. IMPLEMENTATION

## 4.1 Physical Components

We have constructed two dolls, each a stuffed bear approximately 14 inches in height. In the bear's core we have installed an off-the shelf 400MHz Celeron CPU, an A/D PCMCIA converter card to process sensor data, and a USB speaker. Interaction is supported by an USB CCD camera in the nose, a G-Force sensor in the belly, proximity sensors in the face and hips, and touch/bend sensors in the arms, legs, back, front and head. The system runs with Windows 2000 and DirectX 8.1, and was developed using Visual C++ 6.0, DirectMusic, the Microsoft Vision SDK, and an open source image library. The system is completely self-contained, and executes automatically following connection to an internal rechargeable battery. The process for sensor input to musical output inside of the doll occurs at a rate of 4Hz.

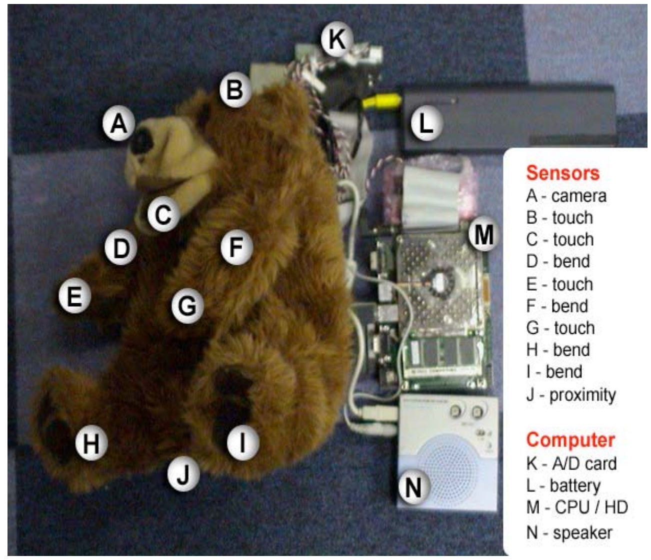



**Figure 2. Hardware components.**

## 4.2 Data Transfer

Our software system builds on existing functionality used with similar projects for the given hardware, but is designed to be reconfigurable and robust, so its overall structure operates with three modules supporting the main component.

The core of the application monitors the bear's current play state and internal attributes, such as sensor stimuli and active sound buffers. A timer polls the sensor array for new data four times a second to prevent backlog of audio feedback events sent to the sound manager. Requests for camera acquisition and analysis are also handled by the main application.

At initialization, a virtual camera object is created by the image processing unit, which then maintains the connection to the USB camera and grabs low-resolution JPEG images when the pressure sensor in the bear's head is activated. This functionality is triggered by the guest when he/she shows the bear a new hat for changing musical themes. The image is then saved to disk with a timestamp. A dominant-hue analysis is performed to determine which color hat is being shown to the bear, and consequently which game is to be played next. Acceptable returns for the hue analysis are extreme amounts of pure red, green, blue or yellow (which the hats are discretely comprised almost entirely of, see figure 2). The algorithm works with a 90% success rate in identifying the correct hat under most indoor lighting conditions. If a dominant color cannot be extracted (if no hat is occupying a majority of the camera's viewing frustum), the bear returns to an idle state of murmuring conversation and touch-activated chattering.

The tactile sensor array polls the hands, arms, legs, base, head and mouth of the bear constantly and sends updated data to the AI's state machine four times a second to keep in sync with sound playback requests. Each request issued maps to a registered audio sample that may be played in a distinct MIDI channel. Samples consist of a riff, one-time hit, or chord, all of which are in key with the main song associated with the musical game. For all four themes, sensor events map to similar tonal components. Squeezing the left or right hand results in a supporting chord, and bending the left or right arm inwards a descending or ascending riff up the scale, respectively. Bending of a leg provides a warm overtone or short melody that backs the bear's main tune. Each sample loops if the corresponding body part is held. Players may experiment with the doll for individual performance by touching the mouth, limbs, and head. Accompaniment mode is invoked when the bear is picked up/held, or laid on its back.

### 4.3 System Flexibility

All of the doll's sounds are general MIDI files, so samples and melodies may be changed by replacing the files and restarting the system. Additionally, since all of the images taken by the camera are stored to disk in JPEG format, the images may be extracted from another computer via a wireless or 10 base-T Ethernet connection, both of which are embedded inside the bear. To provide reliability in sensor on/off thresholds, we have implemented a calibration utility as well that allows the maintainer to physically touch the doll and define for the system what is an acceptable "on" and "off" amount of pressure. All of the data received through the bear's sensors is also capable of being recorded and logged to a Microsoft Access database, which may then be later viewed and played back through another supporting application.

## 5. PRELIMINARY EXPERIMENTS

### 5.1 Reaction Testing

We presented our dolls to a group of 38 7-12 year olds at a local daycare center to evaluate initial reactions from our target demographic. We selected primary school children so that we may observe a wide range of motivations for interacting with the doll, from simple curiousity to structured play. This also gave us the fortuitous opportunity to watch the differently aged children interact and learn from each other. We introduced the bears to children, and explained our intention to design an interactive musical toy to be played with. After demonstrating the basic features of the dolls, the children were allowed to take turns playing with them, first one at a time and then collaboratively. Following this, we asked the each of the children to draw a picture of the bears, write a few sentences about what they did, and then list some other things they would like the bears to be able to do.

### 5.2 Observations and Guest Feedback

We found that the children were very receptive to the dolls, and could interact successfully with little difficulty. Unfortunately, one of the bears was too heavy for some of the children to pick up and play with, as it had a protective metal casing around the computer's core. Some of the children enjoyed the "dress-up" aspect of the dolls more, while others preferred experimenting and testing the limitations of the hardware. Children aged in the median of our demographic appeared to enjoy playing the most, and exhibited the strongest sustained and focused play. They experimented with all of the hats and actuators, in addition to occasionally dancing back and forth when another student would start a new song. These All of the children seemed to appreciate the human-like warbling exhibited when the bears' mouths were pulled on or if the dolls were left alone for a time.

The feedback we received was unilaterally positive and constructive. Many remarks made were that the bears were cute, and their audio feedback was entertaining and interesting. Many children commented they enjoyed touching the bears and wanted to play with them again. Requests for modification included conversational responses from the bears, more hats to dress with, and walking or automated movement of the limbs.

## 6. CONCLUSION

We designed an interactive musical toy application for an augmented sensor doll. Using principles of color psychology and basic music theory, we created an interaction that supports both solo improvisation and assisted performance. Our system is one step towards creating digitally augmented toys that support traditional interaction while encouraging experimentation and personal expression.

There are several areas for further expansion that persuade examination. Increasing the doll's sense of direct awareness in regards to its user and its environment may allow it to respond more intelligently and shift closer to playing the role of being a companion. Enhanced imaging functions and voice recognition methods could be used to control the bear's mood and timbre synthesis, either directly or in a more subtle fashion. We chose to make use of the USB camera-internal PC combination for the physical object recognition and sound processing because we strove to keep the system design as simple, cheap and off-the-shelf as possible. A radio frequency tag system may have been more accurate, but its addition could surpass the physical space limitations inside of the bears, and also significantly raise the total system cost. Similar tradeoffs for performance may arise with the use of a custom hardware microcontroller. Regardless, future investigation of such methods may allow greater freedom in design. The doll may also be enhanced in its role of an instructor, teaching musical concepts through playback games a la a purely audio version of *Simon*. Overall, we hope to see further research in creating toys instructive and compelling enough that they stimulate and promote growth in children, both scholastically and emotionally.

## 7. ACKNOWLEDGMENTS

Our thanks to Hiromi Fukunaga for helping us with our user testing and feedback, and to Rodney Berry for his continued consultation and advice. Most of all, we'd like to thank the children of the Ikoma Gagudo daycare center for trying out our application and giving us such wonderful feedback. This research was made possible in part by the Telecommunication Advancement Organization (TAO).

## 8. REFERENCES

[1] Alexander, K., and Strommen, E. Evolution of the Talking Dinosaur: The (Not So) Natural History of a New Interface for Children. In Proceedings of CHI' 98, ACM Press, 7-8.

[2] Browall, C., and Lindquist K. Our Little Orchestra: The development of an interactive toy. In Proceedings of DARE 2000 on Designing augmented reality environments, ACM Press, 149-150.

[3] Embryonics, Mozart Magic Orchestra Featuring Wolfgang. Munchkin Incorporated. 2000.

[4] Gan, S., Squeezables: Tactile and Expressive Interfaces for Children of All Ages. MS Media Arts and Sciences Dissertation, Massachusetts Institute of Technology School of Architecture and Planning. September 1998.

[5] Lamaze, Octotunes. Learning Curve International. 2003.

[6] Payling, D., Musical Blocks for Children. MSc Music Technology Dissertation, The University of York. August 2001.

[7] Saitoh, K., Yonezawa, T., Mase, K., Awareness Communications by Entertaining Toy Doll Agents. In Proceedings of the International Workshop on Entertainment Computing 2002, Kluwer Publishing, 326-333.

[8] Strommen, E. When the Interface is a Talking Dinosaur: Learning Across Media with ActiMates Barney. In Proceedings of CHI' 98, ACM Press, 288-295.

[9] Strommen, E. Emotional Interfaces for Aardvarks: Designing Affect into Social Interfaces for Children. In Proceedings of CHI' 99, ACM Press, 528-535.

[10] Yonezawa, T., Clarkson, B., Yasumura, M., Mase, K., Context-aware Sensor-Doll as a Music Expression Device. In Extended Abstracts of CHI' 01, ACM Press, 307-308.

# Force Feedback Gesture Controlled Physical Modelling Synthesis

David M Howard
Media Engineering Research Group
Department of Electronics
University of York, Heslington, York
YO10 5DD, UK
+44 1904 432405
dh@ohm.york.ac.uk

Stuart Rimell
Media Engineering Research Group
Department of Electronics
University of York, Heslington, York
YO10 5DD, UK
+44 1904 432407
smr12@ohm.york.ac.uk

Andy D Hunt
Media Engineering Research Group
Department of Electronics
University of York, Heslington, York
YO10 5DD, UK
+44 1904 432375
adh@ohm.york.ac.uk

## ABSTRACT
A physical modelling music synthesis system known as 'Cymatic' is described that enables 'virtual instruments' to be controlled in real-time via a force-feedback joystick and a force-feedback mouse. These serve to provide the user with gestural controllers whilst in addition giving tactile feedback to the user. Cymatic virtual instruments are set up via a graphical user interface in a manner that is highly intuitive. Users design and play these virtual instruments by interacting directly with their physical shape and structure in terms of the physical properties of basic objects such as strings, membranes and solids which can be interconnected to form complex structures. The virtual instrument can be excited at any point mass by the following: bowing, plucking, striking, sine/square/sawtooth/random waveform, or an external sound source. Virtual microphones can be placed at any point masses to deliver the acoustic output. This paper describes the underlying structure and principles upon which Cymatic is based, and illustrates its acoustic output.

## Keywords
Physical modeling, haptic controllers, gesture control, force feedback.

## 1. INTRODUCTION
The incorporation of computer technology in electronic musical instrument has enabled musicians to push back the creative boundaries within which they work, but despite enjoying the resulting freedom from physical constraints, musicians are still searching for virtual instruments that come closer to their physical counterparts. The widespread availability of gestural controllers which now incorporate a tactile element by means of proprioceptive force feedback, following in the wake of PC gaming developments, offers a cost effective route to restoring the musician's sense of working with a true physical instrument in the natural world. The acoustic output from computer instruments is often described as 'cold' or 'lifeless' whereas that from real instruments may be described as 'warm', 'intimate' or 'organic'. These criticisms can be addressed in two ways: (i) by using physical modelling to create organic sounds that more closely resemble those of physical causality, and (ii) by creating new user interfaces that enable musicians to interact with the computer in more intuitive and intimate musical ways.

A new instrument, known as 'Cymatic', is described in this paper that incorporates both of these approaches to create an instrument which provides an immersive, organic and tactile musical experience that is more commonly associated with acoustic instruments and rarely found with computer-based instruments. Cymatic takes inspiration from physical modeling sound synthesis environments such as TAO [1], Mosaic [2] and CORDIS-ANIMA [3], through the use of resonating structures that can be interconnected to create complex and musically versatile virtual instruments playable in real-time using one or more acoustic excitation methods.

Musicians interact with Cymatic's virtual instruments via tactile and gestural interfaces, thereby providing a route to enabling them to interact with the computer in more intuitive and intimate musical ways. In the virtual domain, the player is physically detached from the sound source and therefore is interacting with it indirectly via interface peripherals such as a mouse, MIDI keyboard or musical instrument controller. Second to audition itself, the haptic senses provide the most important means for observing and interacting with the behaviour of musical instruments [4]. Developments in computer-based musical instruments have prioritised visual stimuli over tactile control, with the result that the haptic senses have been left seriously undernourished. It is only possible to realise complex and realistic musical expression when both tactile (vibrational and textural) and proprioceptive cues are available in combination with aural feedback [5, 6]. Previous attempts to rectify this unsatisfactory situation include: electronic keyboards that have a 'feel' close to a real piano [7] the provision of tactile feedback [8], haptic feedback bows that simulate the feel and forces of real bows [9], and the use of finger fitted vibrational devices in open air gestural musical instruments [10].

Existing haptic control devices are generally one-off devices that are restricted to implementation with specific computer systems, and are thereby inaccessible to the musical masses. In contrast, Cymatic exploits the musical interface potential of inexpensive and widely-available PC gaming devices, as its real-time gestural control and haptic feedback is provided by a force feedback joystick and a tactile feedback mouse.

## 2. Overview of Cymatic

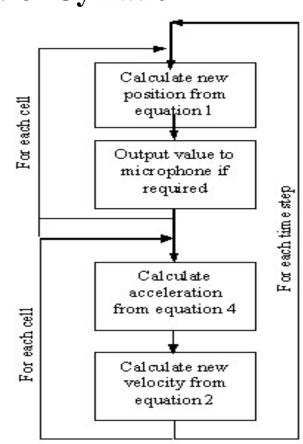



**Figure 1: Schematic depicting the flow of operations in Cymatic's core mechanics functions**

Cymatic is implemented in C under Windows on a standard PC machine. It utilises the mass-spring paradigm of physical modelling to synthesise resonating structures in real time, and the calculations are carried out for each mass cell in terms of updating its position, velocity and acceleration based on the forces acting on that cell. For real-time operation, this set of calculations has to be completed for every cell in the instrument within time *dt* (the reciprocal of the sampling rate selected), otherwise audible output clicks are likely to result since the output waveform will not be fully defined. It can be shown that [11] the position (x), velocity (v) and acceleration (a) can be calculated from equations 1, 2, and 3 respectively.

$$x(t+dt) = x(t) + v(t + dt/2)dt \qquad (1)$$

$$v(t + dt/2) = v(t - dt/2) + a(t)dt \qquad (2)$$

$$a(t) = (1/m)(k\sum (p_n - p_0) - \rho v(t) + F_{external}) \qquad (3)$$

*Where: k is the spring constant, m is the mass of the cell,* $\rho$ *is the viscosity as given by the damping parameter of the cell,* $F_{external}$ *is the force on the cell from any external excitations e.g. plucking or bowing,* $p_n$ *is the position of the* $n^{th}$ *neighbour, and* $p_0$ *is the position of the current cell.*

Cymatic is implemented in C++ on a PC machine. It performs all the calculations required to run the model in a core mechanics function as illustrated in figure 1. The new cell position, velocity and acceleration values are calculated from equations 1, 2, and 8 respectively.

Users design instruments by means of an intuitive graphical user interface (GUI) to create resonant structures of irregular shapes and multiple dimensions. These can be interconnected (any point mass on one structure to any point mass on another structure) to form highly complex virtual instruments. Figure 2 shows an example complex Cymatic instrument. Virtual excitation mechanisms are available (bowing, plucking, oscillators and live-audio input) which can be applied to any point mass within the instrument under real-time or off-line control. Any of the physical parameters can be altered in real time as desired. During synthesis, real-time animation of the instrument is available to provide visual feedback of instrument vibrations. Control over instrument parameters and excitation functions is gestural via a force feedback joystick and force feedback mouse.

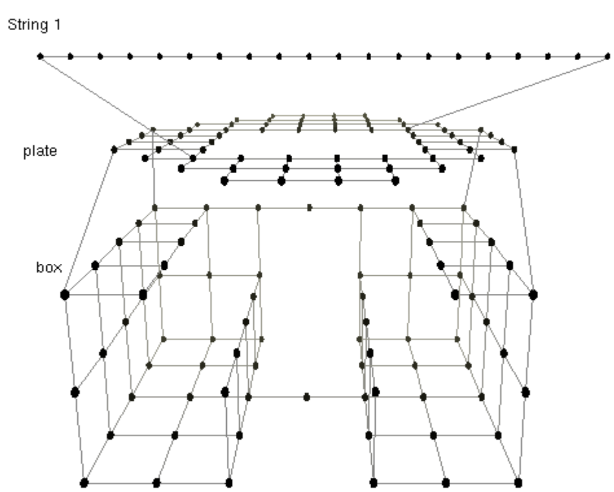



**Figure 2: An example complex Cymatic instrument consisting of an interconnected string, membrane and solid.**

The resulting sound can be heard by placing any number of *virtual microphones* that can be placed at user-defined points on the instrument (see figure 3). The output is the sampled displacement of the mass-spring cell to which it is attached. A selection of standard sampling rates from 96kHz to 8kHz are available, giving the user scope to trade off frequency resolution against the number of point masses that can be incorporated in the instrument for real-time operation.

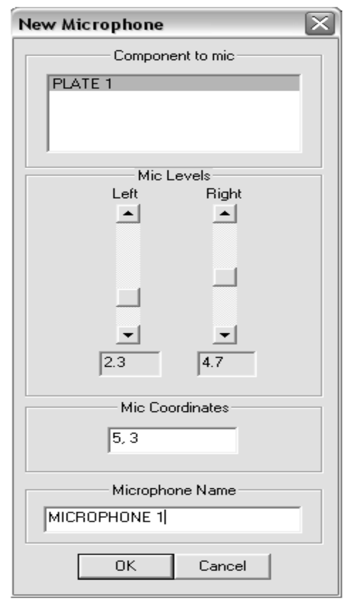



**Figure 3: Cymatic's dialog box enabling placement of virtual microphones at arbitrary cell coordinates**

Figure 4 shows a series of time frames from Cymatic's animated GUI interface to illustrate the motion of a bowed string. The animations provide immediate feedback as to the authenticity of the physically modelled excitations in terms of the design intention and the interaction between various elements.

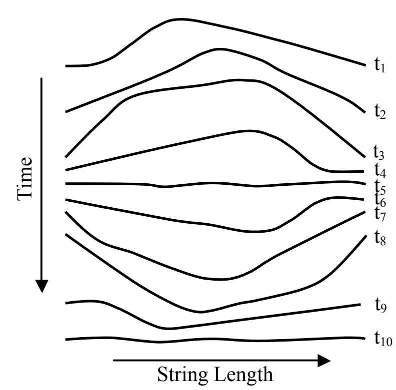



**Figure 4: Time snapshots from a Cymatic animation showing Helmholtz motion of the bowed string model**

Due to the mass-spring nature of the physical modelling process, the instrument can be modified in real-time during synthesis. For example, excitations and virtual microphones can be moved in real-time, 'virtual scissors' can be applied to strings, membranes or solids, and mass or spring parameters can be adjusted.

## 3. Real-Time Control

Real-time control of Cymatic is currently achieved using the joystick and mouse to vary the physical parameters within the instrument components, including mass, tension and damping, excitation force and velocity, excitation point and virtual microphone point.

Cymatic's main controllers are a Microsoft Sidewinder Force Feedback Pro Joystick and a Logitech iFeel mouse. The joystick offers four degrees of freedom (x-movement, y-movement, z-twist movement and a rotary "throttle" controller) and eight buttons. It also provides tactile and proprioceptive feedback with a high degree of customizability, boasting the potential to output six forces simultaneously. Force feedback implementation for Window's devices is normally achieved via DirectX and the 'Immersion Foundation Classes.' The Microsoft Sidewinder Force Feedback Pro joystick receives its force instructions via MIDI through the combined MIDI/joystick port on most PC sound cards. Cymatic can stimulate the joystick's haptic capabilities by simply outputting the appropriate MIDI messages.

The Logitech iFeel mouse is a simple optical mouse which implements Immersion's [12] iFeel technology, containing a vibrotactile device to produce tactile feedback over a range of frequencies and amplitudes. It offers two degrees of freedom and three buttons. Stimulating the iFeel mouse's tactile feedback is achieved through Immersion's, "Immersion Touchsense Entertainment" software which converts any audio signal to tactile sensations on the iFeel mouse. The parameter assigned to each controller function is fully customizable so the controllers can be adapted to the type of instrument or excitation method that Cymatic is running through arbitrary mapping by the user.

The Microsoft joystick can simulate a wide range of time- and position-based haptic sensations. Time-based effects include periodic oscillations of a variety of waveforms and a wide range of amplitudes and frequencies (from 1Hz to approx 300Hz). Constant forces, pulses and recoils also come under this category. Position-based effects include sensations of friction, inertia, solid surfaces and damping as well as spring like forces which increase or decrease as a function of the displacement of the joystick handle. The forces are identified by the joystick as system-exclusive messages and can be played, stopped and altered parametrically in real-time with MIDI control change and aftertouch messages.

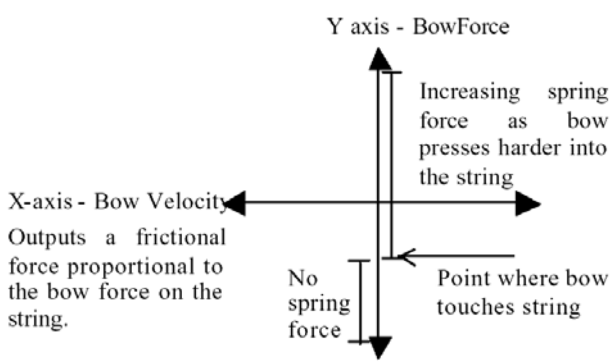



**Figure 5: The haptic mapping of the joystick for Cymatic's bowed string model**

Using these methods it is possible to design haptic sensations to suit the instrument and excitation type running on Cymatic. Figure 5 shows the haptic mapping for a Cymatic bowed string, in which the Y axis of the joystick is mapped to the force of the bow and the rate of change of the x-axis is mapped to the velocity of the bow. The user will feel no force until the virtual bow contacts with the string, after which the more force placed upon the string, the greater the force that is felt. A friction force in the x direction will also increase with respect to increasing the force placed on the string. The iFeel mouse is useful to simulate the velocity of the virtual bow while the joystick Y-axis takes care of the force parameter. This ensures that the user has to provide energy to the instrument via a gestural input in order to achieve an output. Other controller functions can be used to alter different parameters e.g. the 'twist' function of the joystick can change the tension of the string to be used as a vibrato effect and the 'throttle control' can be mapped to the bow position on the string, microphone position etc. A periodic force related to the amplitude and frequency of the audio output is felt through the iFeel mouse and the joystick's handle.

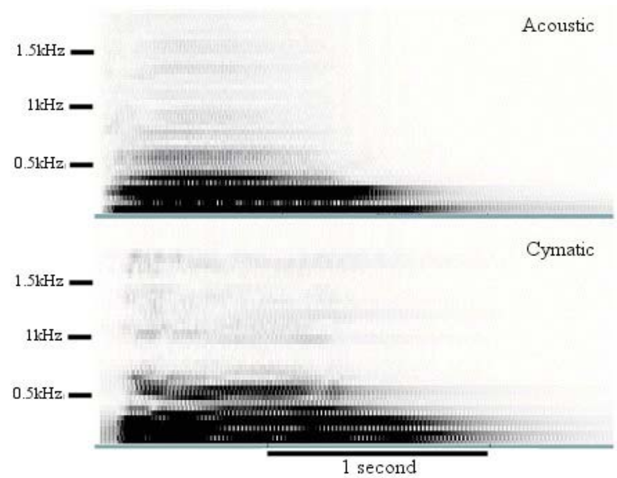



**Figure 6: Spectrograms of the outputs from an acoustic double bass (upper) and a Cymatic bowed virtual instrument tuned to the same pitch.**

Figure 6 enables the acoustic outputs from Cymatic and an acoustic double bass to be compared spectrographically, illustrating something of the potential organic nature of the Cymatic output sound. This is one of the key features of physical modelling. The Cymatic instrument has been bowed and its string set to produce the same fundamental frequency as that obtained from the acoustic double bass to enable comparison. There are clear similarities between the nature of the note onset and offset in each case, and it should be remembered that every note played on a Cymatic virtual instrument will be acoustically subtly different since each will have its origins in a unique gesture. Haptic and gestural mappings can be readily implemented to suit the individual instrument's needs.

## 4. Discussion

Cymatic was conceived with the desire to create new sounds through the use of new instruments, which may not be physically realisable in the real world, and to provide the player with a more engaging musical experience when performing with the instrument. The use of a physical modeling paradigm enables intuitions gained through training and performance with acoustic instruments to be immediately transferred to Cymatic instruments. The addition of tactile feedback reinforces the available visual and aural feedback cues, helping the player to develop internal models that are physically rooted in the manner in whcih the instrument responds to gesture.

Cymatic made its concert debut in December 2002 to universal audience acclaim at a public performance in York in a specially written piece by Stuart Rimell for a small (13 strong) SATB choir and Cymatic. Here the choir provided a backdrop over which a three sheet Cymatic instrument performed an obbligato. Extracts from this piece are available at [13].

Cymatic provides a new environment within which new musical instruments can be implemented, explored, and interacted with in live performance and composition. Only a very small subset of the potential possibilities has been explored to date, and next steps will focus on the creative side in an exploration of Cymatic's wider timbral potential. Cymatic instrument possibilities have the potential to feed musical imagination for a considerable time to come.

# 5. ACKNOWLEDGMENTS

This work was supported by EPSRC grant number GR/M94137.

# 6. REFERENCES

[1] Pearson, M.D., and Howard, D.M. (1996). Recent developments with TAO physical modelling system *Proceedings of the International Computer Music Conference*, ICMC-96, 97-99.

[2] Morrison, J.D. and Adrien, J.M. (1993). MOSAIC: A Framework for Modal Synthesis, *Computer Music Journal,* 17, (1), 45-56.

[3] Cadoz, C., Luciani, A., and Florens, J.L. (1993). CORDIS-ANIMA: A Modelling system for sound and image synthesis, the general formalism, *Computer Music Journal* 17, (1), 19-29.

[4] Cook, P.R. (1999). *Music, Cognition and Computerised Sound: An Introduction to Psychoacoustics,* London: MIT Press, pp. 229.

[5] MacLean, K.E. (2000). Designing With Haptic-Feedback, *www.cs.ubc.ca/~maclean/publics/icra00-DesignWithHaptic-reprint.PDF*

[6] Howard, D.M., Rimell, S., Hunt, A.D., Kirk, P.R., and Tyrrell, A.M. (2002). Tactile feedback in the control of a physical modelling music synthesiser, In: Proceedings of the 7th International Conference on Music Perception and Cognition, Stevens, C., Burnham, D., McPherson, G., Schubert, E., and Renwick, J. (Eds.), Adelaide: Casual Publications, 224-227.

[7] Gillespie, B. (1992). Proc ICMC San Jose, CA. pp. 447-448.

[8] Cadoz, C., Luciani, A. & Florens, J.L. (1984). Responsive Input Devices and Sound Synthesis by Simulation of Instrumental Mechanisms: The Cordis System*. Computer Music Journal* 8, (3), 60-73

[9] Nichols, C. (2001). The vBow: Development of a Virtual Violin Bow Haptic Human Computer Interface, *Proceedings of the NIME-02 Conference on New Interfaces for Musical Expression*, ISBN (CD-ROM): 1-87465365-8, 29-32.

[10] Rovan, J. (2000) Typology of Tactile Sounds and their Synthesis in Gesture-Driven Computer Music Performance. In *Trends in Gestural Control of Music.* Wanderley, M., Battier, M. (eds). Editions IRCAM, Paris.

[11] Rimell, S., and Howard, D.M. (2003). Tactile controlled physical modelling music synthesis, *paper submitted to IEEE Transactions on Speech and Audio Processing*.

[12] www.immersion.com

[13] www-users.york.ac.uk/~dmh8

# Real-time Adaptive Control of Modal Synthesis

Reynald Hoskinson
Department of Computer Science
University of British Columbia
Vancouver, Canada
reynald@cs.ubc.ca

Kees van den Doel
Department of Computer Science
University of British Columbia
Vancouver, Canada
kvdoel@cs.ubc.ca

Sidney Fels
Department of Electrical and Computer Engineering
University of British Columbia
Vancouver, Canada
ssfels@cece.ubc.ca

## ABSTRACT

We describe the design and implementation of an adaptive system to map control parameters to modal audio synthesis parameters in real-time. The modal parameters describe the linear response of a virtual vibrating solid, which is played as a musical instrument by a separate interface. The system uses a three layer feedforward backpropagation neural network which is trained by a discrete set of input-output examples. After training, the network extends the training set, which functions as the specification *by example* of the controller, to a continuous mapping allowing the real-time morphing of synthetic sound models.

We have implemented a prototype application using a controller which collects data from a hand-drawn digital picture. The virtual instrument consists of a bank of modal resonators whose frequencies, dampings, and gains are the parameters we control. We train the system by providing pictorial representations of physical objects such as a bell or a lamp, and associate high quality modal models obtained from measurements on real objects with these inputs. After training, the user can draw pictures interactively and "play" modal models which provide interesting (though unrealistic) interpolations of the models from the training set in real-time.

## 1. INTRODUCTION

Musical instruments are usually selected before a performance and then played in real-time. Occasionally a versatile performer may play several instruments during a piece, sometimes even simultaneously. However, switching instruments is usually not considered to be part of the performance skills of the artist but taken more or less for granted.

This metaphor has been propagated to digital instruments which have elaborate real-time controllers (keyboard, MIDI wind-controller, drum pad, etc.) for playing the instrument, but simple switches to select the instruments or "presets".

Physical musical instruments allow a limited amount of real-time modification of the instrument's behavior, and in the 20th century music composers have moved some of these controls into the performance area. For example requiring a cello player to retune a string while playing, can extend the scope of the instrument.

Synthetic digital instruments using real-time audio synthesis [26] offer the possibility to make the virtual instrument completely flexible and, by changing the synthesis parameters in real-time, allow the morphing of different instruments into each other. This gives the performer the ability to control the nature of the instrument itself in real-time but poses the challenge of finding intuitive and natural interfaces to control these "design parameters".

In this paper we describe a software system which attempts to provide a generic framework to construct real-time controllers for digital synthesis algorithms. Our system uses a backpropagation neural network to map the control variables, which the performer directly controls, to the synthesis variables in a configurable and adaptive way. This is done by training the network on a set of input-output pairs which describe some of the desired properties of the mapping. This can be thought of as defining a collection of instrument presets which are specified by input variables of the performers choice. Once the network is trained, a real-time control map is generated which generalizes the training set to a continuous map allowing continuous control. Because of the neural network's ability to detect features, we believe this mapping is able to generalize the performer's intent in some sense, rather than just provide some arbitrary interpolation.

## 1.1 Related Work

There have been several attempts to create adaptive mappings between gesture and sound. Most notably, [13] used neural networks to map hand gestures to speech formant amplitudes and frequencies, which were excited by a different controller. The neural networks allowed the system to learn the relationship between the speaker's mental model space and the actual sound space. The speaker thus needed only to work in the relatively easy articulatory space instead of formant space.

A combination of neural networks and fuzzy logic software intended for real-time musical instruments control written in the MAX real-time programming environment was described in [15]. An adaptive conductor follower based on neural networks was described in [16].

Of course, many hand-crafted systems to help facilitate learning the mapping between gesture and music have been attempted. For example, refer to [25, 12] for a description of a number of these devices. These mappings strategies all depend upon the intuition of the designer. Several common strategies have been developed to make the mapping easy to learn and understand. One typical strategy is to instrument a pre-existing acoustic instrument such as a cello [17] or saxophone [1]. This approach has the advantage of constraining the player's gesture space to a predefined, already learned space. Unfortunately, the output space may not have any obvious relationship to the gestures. Another technique uses objects that already have clear affordances [21] for control but are not necessarily based on acoustical instruments [2]. Objects such as a coffee mug can be instrumented and interactions with them mapped to sounds. While the mapping may not be clear at the outset, the fun of the interface form encourages a player to begin making sounds and exploring the interface. Other strategies include the use of metaphors [12].

In all the situations above, an adaptive system may be helpful in improving the transparency of the mapping. By carefully choosing the objective space and letting an adaptive algorithm match this to the player's mental model of the gesture-to-sound mapping, improvements should be possible. The role that the mapping plays in determining whether a musical interface is expressive is very complex [23]. The adaptive interface is one technique to help make new interfaces for musical expression.

## 1.2 Overview

Our prototype system has been applied to generate a control strategy for modal synthesis using hand-drawn greyscale pictures. Several pictures are associated with physical models of the objects they are intended to depict, which are linear modal models whose parameters were obtained by fitting them to sound recordings of real objects. Modal models of "everyday" objects such as lamps, kettles, coffee cups, etc. require anywhere from 4 to 100 modes for high quality sounds, which results in $12 - 300$ synthesis parameters to control, which is a very large space. This space contains the linear sound behavior of every imaginable rigid body, from wooden tables to the liberty bell, to the sound of an oddly shaped piece of scrap metal lying on some junkyard! Because of the large size of the sound space it is not possible to manually design the coupling of every synthesis parameter to some physical controller, and the need for a more automated approach to control such as that proposed in this paper becomes apparent.

Because there are so many synthesis parameters, we need a control space which is large enough to reach a substantial portion of the possible sound models. The greyscale level of the pixels of an image provide this large control space. After training the network on the examples, we deploy the trained network in a real-time application where the user can interactively draw a picture and have the modal parameters change in real-time. This simple interface requires no special hardware and is easy to work with, even for non-musicians, and therefore allows us to use it as a good testbed application for our controller design. We believe it also results in an very entertaining sonified drawing application.

The modal model can be excited by any means (or could be embedded in a more complicated synthesis patch) and for testing purposes we use impulses, noise excitations and a live data stream from a contact mike [4] which allows a more direct interaction with the virtual object.

The remainder of this paper is organized as follows. In Section 2 we describe and justify our control model and establish some notation. In Section 3 we describe our instrument model and design and summarize modal synthesis. In Section 4 we describe our prototype application and results obtained, and conclusions and directions for future work are presented in Section 5.

# 2. THE CONTROL MAP

To articulate the problem we find it useful to describe the mapping in a somewhat abstract manner. Let us denote the continuous synthesis parameters describing a virtual instruments by an $N$-dimensional vector $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_N\}$, which we can visualize as a point in "instrument space" $\Theta$. This space consists of all possible virtual instruments that can be modeled by changing parameters of a synthesis algorithm. A "preset" of an algorithm corresponds to a single point in $\Theta$. We can visualize a conventional synthesizer with preset buttons as consisting of a cloud of points in $\Theta$ which we can navigate with buttons (or some other discrete interface).

A continuous interface to instrument selection allows the performer to navigate smoothly between the presets and for example morph a woodblock into a gong while playing. However, its is not clear how to move from one preset to the other in the most natural way. Naively one could interpolate linearly in parameter space but this is arbitrary and does not "sound linear".

For example, let us morph the sound of a bell into the sound of a desk lamp by a linear trajectory in modal space (consisting of the frequencies, dampings, and gains), and control this with a single parameter $\lambda$ which runs from 0 (a metal desk lamp) to 1 (a church bell). An interactive application which runs in most web browsers demonstrating this can be found on the web [6]. If we start at 1 and decrease $\lambda$, we first hear the bell going out of tune. Somewhere around $\lambda = 0.9$ the bell character is lost and from 0.9 to around 0.1 it sounds like "some metal object", but the character of the sound remains fairly constant until we come close to the lamp, around $\lambda = 0.1$ when the sound appears to rapidly "come into focus" and morph into the sound of a desk lamp. This somewhat subjective description illustrates the fact that though the trajectory is linear in parameter space and we move uniformly from one point to the other, what we hear does not sound linear and uniform at all.

Another challenge in designing interfaces is to provide gestural metaphors which are natural to the performer. Controlling motion in $\Theta$ *adaptively* allows the performers to customize the mapping according to their own peculiarities and wishes within the same system. A control interface is a continuous mapping

$$\kappa : \mathcal{C} \longrightarrow \Theta$$

from a control space $\mathcal{C}$ to the instrument model space $\Theta$. The $K$-dimensional space $\mathcal{C}$ consists of all possible settings of the control variables $\boldsymbol{c} = \{c_1, \ldots, c_K\}$. These control variables are obtained from sensors such as Cybergloves, position trackers, etc. and are controlled by the performer in real-time.

Presets are input configurations (points in $\mathcal{C}$) which are mapped to fixed instruments. The preset configuration $\rho$ is defined by specifying $M$ pairs $\rho = \{\{\boldsymbol{c^1}, \boldsymbol{\theta^1}\}, \ldots, \{\boldsymbol{c^M}, \boldsymbol{\theta^M}\}\}$, where $\boldsymbol{c^i} \in \mathcal{C}$ and $\boldsymbol{\theta^i} \in \Theta$. It is a discrete mapping $\rho$ from $\mathcal{C}$ to $\Theta$. We shall notate the preset control set by $\mathcal{C}_p = \{\boldsymbol{c^1}, \ldots, \boldsymbol{c^M}\}$, and the preset instrument set by $\Theta_p = \{\boldsymbol{\theta^1}, \ldots, \boldsymbol{\theta^M}\}$. See Figure 1 for the notation.

A natural framework for constructing the continuous mapping $\kappa$ as a generalization of the discrete mapping $\rho$ is a 3 layer backpropagation feedforward neural network [19] with $K$ inputs and $N$ outputs which, appropriately scaled, provides the mapping $\kappa$. The preset configuration $\rho$ provides a set of $M$ training examples, and training the network on this set results in the desired mapping $\kappa$. An important feature of neural networks is their ability to detect and generalize features [19]. This is very relevant as the preset map $\rho$ captures the performer's metaphor for control. The continuous interpolation of the preset configuration can incorporate features which are detected during the training phase by the neural net and generalize them. The preset configuration can also be seen as the specification by example of the desired behavior of the controller.

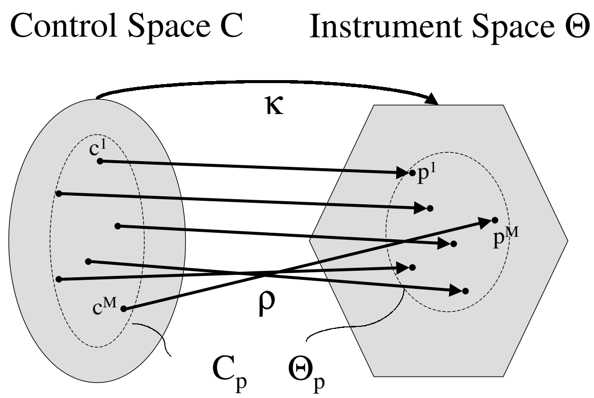



**Figure 1: Control space C and instrument space Θ. The discrete preset mapping $\rho$ is generalized to the continuous mapping $\kappa$ by training a 3 layer back-propagation neural network on $\rho$.**

## 3. MODAL INSTRUMENT SPACE

A good physically motivated synthesis model for vibrating rigid bodies is modal synthesis [28, 14, 20, 3, 8, 9, 7, 22]. Modal synthesis models a vibrating object by a bank of damped harmonic oscillators which are excited by an external stimulus. See Fig. 2 for an illustration. The frequencies and dampings of the oscillators are determined by the geometry and material properties (such as elasticity) of the object and the coupling gains are determined by the location of the force applied to the object. The impulse response $p(t)$ of the modal model with $L$ modes is given by

$$p(t) = \sum_{n=1}^{L} a_n \exp(-d_n t) \sin(2\pi f_n t), \quad (1)$$

for $t \geq 0$ and is zero for $t < 0$, where $p(t)$ denotes the audio signal as a function of time. The modal parameters are the frequencies $f_n$, the dampings $d_n$, and the gains $a_n$. The frequencies and dampings are pure object properties whereas the gains also depend on the location of the interaction point on the surface of the object. The model ignores phase effects.

We create sound models with the FoleyAutomatic [7] system, which allows the creation of realistic sound models based on modal synthesis as well as various contact force models which include striking, scraping, sliding, and rolling. The FoleyAutomatic system is freely available from the web as part of the JASS system [10, 5], a Java based real-time audio synthesis toolkit. The modal models can be acquired by parameter fitting to recorded sounds using the techniques described in [24]. Preliminary user studies [11] have shown that impact sounds constructed with this technique are indistinguishable from the real sound.

## 4. INTERACTIVE DRAWING

We have applied our adaptive controller framework to an interactive drawing application which allows the user to draw pictures on a square window. The picture is down-sampled to $16 \times 16$ greyscale pixels with values in the range $0-1$. The pixels are taken as inputs to a neural net with 256 input units, 32 or 128 hidden units, and 60 output units, allowing for modal models of 20 modes.

The neural network was designed using the Java Object-Oriented Neural Engine (JOONE), an open-source neural net package implemented in Java [18]. JOONE provides a

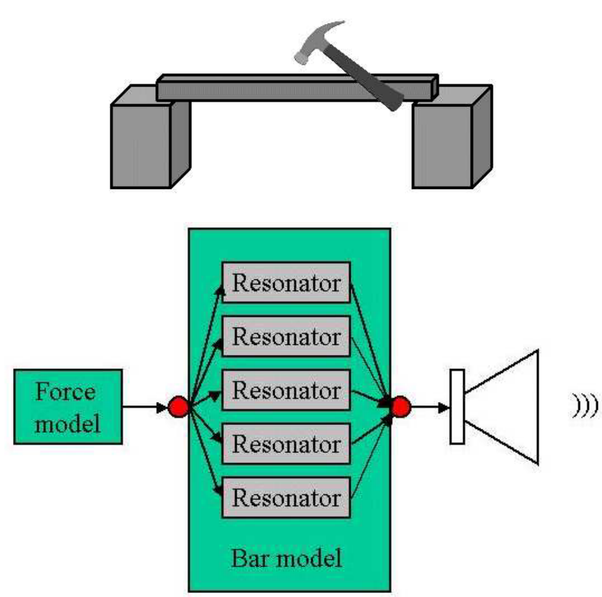



**Figure 2: Modal synthesis of the sound made by hitting a bar with a hammer. The hammer force is modeled by a contact force model, and send to a bank of resonators, which is the modal model of the bar. Each resonator has a characteristic frequency, damping, and gain and the outputs of the resonators are summed and rendered.**

graphical environment to design multilayer neural networks, train them, and export trained networks into real-time applications. All of the neurons are implemented as sigmoid functions $y = 1/(1 + e^{-x})$. The learning rate is set to 0.8, and the momentum factor 0.3.

The 60 outputs of the net are numbers in the range $0-1$. They mapped to the 60 modal synthesis parameters defined in Equation 1, for $L = 20$ modes. For optimum training of the neural net, the range $0-1$ should be mapped as uniformly as possible to perceptually relevant parameters. For instance, frequencies are perceived on a roughly logarithmic scale, so we would like a linear change in outputs to produce a logarithmic change in frequency. The three types of modal parameters are handled separately in order to best take into account the perceptual characteristics of the sounds.

For frequencies, we convert to the Bark scale [27], designed to uniformly cover the human auditory range. It can be expressed as $z = [26.81/(1 + 1960/f)] - 0.53$, with $f$ the frequency in Hz. The result $z$ is then scaled to between 0 and 1. For damping, the conversion is given by $(\log_e(d + 1.0))/5.0$. It covers dampings of up to roughly $150/s$, the most heavily damped modes that occur in the specific physical models we have used. Gains are converted to decibels, and we allow a range of $160dB$, enough for most (non-lethal) applications. The conversion is given by $1 + dB(a)/160$, with $dB(a) = 20 \log_{10}(a)$ the decibel level in the range $-160dB$ to $0dB$.

The preset configuration consists of four hand-drawn pictures depicted in Figure 3. The outputs corresponding to the images are modal models obtained from parameter fitting to recorded sounds of the objects depicted, using the 20 most important modes selected by a perceptual criterion as described in [11], which result in very realistic sounds.

Two neural networks were created, one with 32 hidden

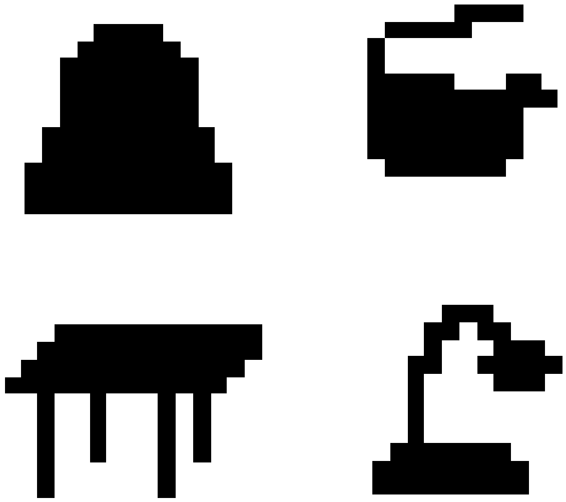


**Figure 3: The four input images to the neural net, depicting a bell, a kettle, a wooden table, and a desk lamp.**

units and one with 128 hidden units. Both were trained until the error in frequencies was below 10 cents (one tenth of a semitone). Errors in the dampings and gains are perceptually much less noticeable, which is why we use the frequencies as a convergence criterion.

Convergence required about 200 iterations, less than one minute on a desktop computer with 733 MHz dual Pentium III processors. In Figure 4 we show the average error of the output as a function of the number of training epochs.

Qualitatively, we listened to the sounds at various stages in the training, obtained by using a picture from the training set as input. After 100 training epochs the results were recognizable as the target sounds but quite distorted, whereas the sound was indistinguishable from the target at 200 training epochs.

After training, we tested our real-time drawing application with fully converged nets containing 32 and 128 hidden nodes, using various excitations. We did not notice any qualitative differences in the behavior of the nets, though there were clear differences between them in sound for pictures we drew which did not resemble any in the training set. The interface allows us to load any of the pictures in the training set and then interactively draw over them. Though the preset configuration with just four presets is very minimal, we were surprised by the richness of the interface. For example, if we start with the bell, when its lower or upper portions are erased, the sound changes dramatically and rapidly loses its bell-like character. But if we erase parts of the picture starting from the middle, the pitch of the bell seems to change, and it is almost possible to etch out a shape inside the bell such that the modes remain in tune and the bell character of the sound is preserved.

If the picture is completely erased or completely black, we do not get a silent model, but rather something which we can only describe as "non-descript". When we draw random shapes, they sound just like that, like random sounds. It is only when features of the input images appear in the drawing that the sounds become "interesting".

We find it very hard to describe the experience with the interface, and intend to convert the application into a Java applet and make it available on the web to interact with through a standard web-browser.

## 5. CONCLUSIONS

This paper has described the design of a general framework to control audio synthesis algorithms with many continuous parameters. The controller maps an input space, which is the space in which the performer manipulates input devices, into the parameter space of a synthesis algorithm using a neural network.

The behavior of the controller is specified by example by specifying a discrete set of input-output pairs, which we have called the "preset configuration". These examples capture the performers intent and a neural network can possibly extract enough features from the examples to generalize it to a "natural" continuous mapping.

Our implementation consists of an interactive drawing application, with the drawing functioning as the controller. Through a neural network the drawing application is controlling parameters of a modal synthesis algorithm. The neural network is trained on a set of images with associated sound models. A real-time synthesis kernel then allows the user to "play" this modal synthesis algorithm by various means. When one of the training examples is drawn, the exact sound model is reproduced, but when a picture outside the training set is drawn the result is not a-priory known, but determined by the neural network's interpolation. Of course, if we draw a realistic image of a real object not in the training set, the resulting sound model will not be realistic, as the modes will depend on the internal structure and other material properties not contained in an image. However, the interpolated models are musically rich and interesting, drawing on features of the objects in the training set.

Our implementation is in an early stage of development and there are several issues which we will address in the near future. First we will extend the training set to include more images to allow the neural net to extract meaningful features. Many similar drawings of the same object should be included in the training set, which can probably achieved by adding noise to the input set. It would be interesting to verify if translation and rotation invariance can easily be learned by including translated and rotated examples in the training set. Next we will incorporate a webcam into the current implementation as an input device, which will provide a very interesting live controller. We are also very interested in applying the controller to live performance, or as a base of an interactive acoustic installation.

## 6. REFERENCES


[1] M. Burtner. Noisegate 67 for Metasaxophone: Composition and Performance Consideration of a New Computer Music Controller. In *Second International Conference on New Interfaces for Musical Expression(NIME02)*, pages 71–76, Dublin, Ireland, 2002.

[2] P. Cook. Principles for Designing Computer Music Controllers. In *First Workshop on New Interfaces for Musical Expression (NIME01), ACM Special Interest Group on Computer-Human Interfaces*, Seattle, USA, 2001.

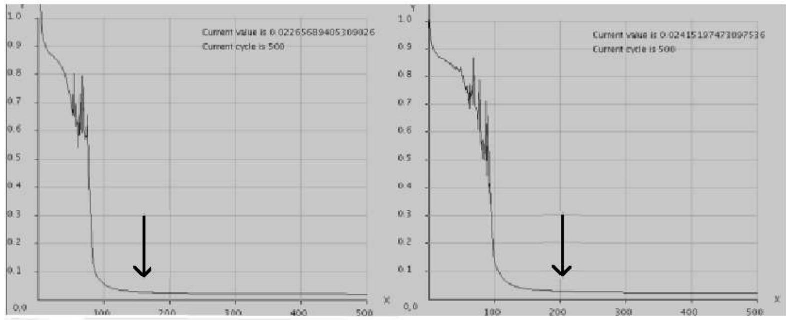



**Figure 4: Convergence graphs of two neural nets we tested. Each has 256 inputs. The first, with 128 hidden shows convergence at under 200 iterations. The second, with 32 hidden nodes, shows convergence a little later, but is still acceptable at the 200-iteration mark.**


[3] P. R. Cook. Physically informed sonic modeling (PhISM): Percussive synthesis. In *Proceedings of the International Computer Music Conference*, pages 228–231, Hong Kong, 1996.

[4] K. v. d. Doel. *Sound Synthesis for Virtual Reality and Computer Games*. PhD thesis, University of British Columbia, 1998.

[5] K. v. d. Doel. JASS Website, http://www.cs.ubc.ca/~kvdoel/jass, 2003.

[6] K. v. d. Doel. JASS Website, Morph Example, http://www.cs.ubc.ca/~kvdoel/jass/morph2/morph2.html, 2003.

[7] K. v. d. Doel, P. G. Kry, and D. K. Pai. FoleyAutomatic: Physically-based Sound Effects for Interactive Simulation and Animation. In *Computer Graphics (ACM SIGGRAPH 01 Conference Proceedings)*, Los Angeles, 2001.

[8] K. v. d. Doel and D. K. Pai. Synthesis of Shape Dependent Sounds with Physical Modeling. In *Proceedings of the International Conference on Auditory Display 1996*, Palo Alto, 1996.

[9] K. v. d. Doel and D. K. Pai. The Sounds of Physical Shapes. *Presence*, 7(4):382–395, 1998.

[10] K. v. d. Doel and D. K. Pai. JASS: A Java Audio Synthesis System for Programmers. In *Proceedings of the International Conference on Auditory Display 2001*, Helsinki, Finland, 2001.

[11] K. v. d. Doel, D. K. Pai, T. Adam, L. Kortchmar, and K. Pichora-Fuller. Measurements of Perceptual Quality of Contact Sound Models. In *Proceedings of the International Conference on Auditory Display 2002*, Kyoto, Japan, 2002.

[12] S. Fels, A. Gadd, and A. Mulder. Mapping transparency through metaphor: towards more expressive musical instruments. In *Organized Sound*, page to appear. Cambridge Press, 2003.

[13] S. S. Fels and G. E. Hinton. Glove-TalkII: A neural network interface which maps gestures to parallel formant speech synthesizer controls. *IEEE Transactions on Neural Networks*, 9(1):205 – 212, 1998.

[14] W. W. Gaver. Synthesizing auditory icons. In *Proceedings of the ACM INTERCHI 1993*, pages 228–235, 1993.

[15] M. Lee, G. Garnett, and D. Wessel. An Adaptive Conductor Follower. In *Proceedings of the International Computer Music Conference*, pages 172–175, San Jose, CA, 1993.

[16] M. Lee and D. Wessel. Neuro-Fuzzy Systems for Adaptive Control of Musical Processes. In *Proceedings of the International Computer Music Conference*, Tokyo, Japan, 1993.

[17] T. Machover. Hyperinstruments: A Composer's Approach to the Evolution of Intelligent Musical Instruments. In *Organized Sound*, pages 67–76, San Francisco, 1991, Cyberarts.

[18] P. Marrone. JOONE Website, http://joone.sourceforge.net, 2003.

[19] J. L. McClelland, D. E. Rumelhart, and the PDP Research Group. *Parallel distributed processing: Explorations in the microstructure of cognition. Volume 1*, volume 1. MIT Press, Cambridge, 1986.

[20] J. D. Morrison and J.-M. Adrien. Mosaic: A framework for modal synthesis. *Computer Music Journal*, 17(1), 1993.

[21] D. Norman. *The Design of Everyday Things*. Currency/Doubleday, 1990.

[22] J. F. O'Brien, C. Chen, and C. M. Gatchalian. Synthesizing Sounds from Rigid-Body Simulations. In *SIGGRAPH 02*, 2002.

[23] N. Orio, N. Schnell, and M. Wanderley. Input Devices for Musical Expression: Borrowing Tools from HCI. In *First Workshop on New Interfaces for Musical Expression (NIME01), ACM Special Interest Group on Computer-Human Interfaces*, Seattle, USA, 2001.

[24] D. K. Pai, K. v. d. Doel, D. L. James, J. Lang, J. E. Lloyd, J. L. Richmond, and S. H. Yau. Scanning physical interaction behavior of 3D objects. In *Computer Graphics (ACM SIGGRAPH 01 Conference Proceedings)*, Los Angeles, 2001.

[25] J. Paradiso. Electronic music interfaces: new ways to play. *IEEE Spectrum Magazine*, 34(12):18–30, 1997.

[26] J. O. Smith. Physical modeling synthesis update. *Computer Music Journal*, 20(2):44–56, 1996.

[27] H. Traunmuller. Analytical expressions for the tonotopic sensory scale. *J. Acoust. Soc. Am.*, 88:97–100, 1990.

[28] J. Wawrzynek. VLSI models for real-time music synthesis. In M. Mathews and J. Pierce, editors, *Current Directions in Computer Music Research*. MIT Press, 1989.

# Playability Evaluation of a Virtual Bowed String Instrument

Diana Young
MIT
Media Lab
Cambridge, MA
diana@media.mit.edu

Stefania Serafin *
CCRMA, Department of Music
Stanford University
Stanford, CA
serafin@ccrma.stanford.edu

## ABSTRACT

Driving a bowed string physical model using a bow controller, we explore the potentials of using the real gestures of a violinist to simulate violin sound using a virtual instrument. After a description of the software and hardware developed, preliminary results and future work are discussed.

## 1. INTRODUCTION

One of the aspects that makes the family of bowed string instruments successful is the incredible degree of nuance and expressivity that a player can obtain with a bow. In building virtual bowed string instruments, the same expressivity is a desirable characteristic. The issue of expressivity in synthetic bowed string research is well known in the computer music field. The importance of controlling a bowed string physical model with input parameters that simulate a physical gesture was first underlined in the work of Chafe [2] and Jaffe and Smith [5]. In this research the combination of the input parameters of a bowed string physical model was used to reproduce different bow strokes such as detaché, legato and spiccato.

Although the goal was to reproduce a particular sound specific to a certain performer's gesture, no real-time input controller was used. To our knowledge, one of the first attempts to control a bowed string physical model using devices other than the traditional mouse and keyboard was proposed in the early 90s by Cadoz and his colleagues at ACROE. Using a device capable of providing tactile feedback the player was able to feel the synthetic bowed string. Recent results of this research are detailed in [3].

Recently, an increasing interest has been shown in controllers for virtual bowed strings developed using physical models.

Preliminary experiments using a Wacom tablet to control a real-time waveguide bowed string model developed in the Max/MPS environment were performed at IRCAM and described in [10]. The Wacom tablet allows a straightforward mapping of the parameters of the bowed string physical model, since it is able to capture the pressure and two axis position of the pen provided with the tablet itself; these values can be easily mapped to the bow pressure, velocity and bow-bridge distance. Moreover, the tablet is able to detect the horizontal and vertical tilt angle of the pen. This immediately shows an advantage of the Wacom tablet versus more traditional input devices such as a mouse and a keyboard, i.e. the possibility of having the same degrees of freedom as a bow in contact with a string. The tablet, however, shows limitations that are mainly illustrated by the difficulties of using it as a performance instrument. These are due also to the lack of tactile feedback (because of the lack of elasticity of the tablet compared to the bow hair) and the dramatic difference between its ergonomics and that of a traditional violin bow.

This lack of force feedback was compensated for when controlling the same bowed string physical model using the Moose, a device built by Sile O'Modhrain and Brent Gillespie. Experiments show that the playability of the bowed string physical model greatly increases when tactile feedback is provided [7]. Researchers simulated the friction of the bowed string using the Moose, thereby providing force feedback associated with bowing gesture.

In order to introduce both force feedback and ergonomics that are reminiscent of a traditional violin interface, Charles Nichols built the vBow [6], a haptic feedback controller. The goal of the vBow is to be able to introduce a new violin interface that addresses the prior limitations of MIDI violins as well as to provide a controller that can also play other real-time synthesis tools.

In this paper, we extend the previous research by proposing an evaluation of the playability of a virtual bowed string instrument when driven by a bow controller that has the same ergonomics of a traditional violin bow.

## 2. ASPECTS OF PLAYABILITY

In virtual musical instruments and musical acoustics, the word *playability* has different definitions. While in this paper we focus on playability of virtual bowed strings, the same issues can be applied also to all other expressive virtual instruments.

According to Jim Woodhouse [14], playability of virtual instruments means that the acoustical analysis of the waveforms produced by the model fall within the region of the multidimensional space given by the parameters of the model. This is the region where *good tone* is obtained. In the case of the bowed string. *good tone* refers to the Helmholtz motion, i.e. the ideal motion of a bowed string that each player is trying to achieve. The Helmholtz motion is given by an alternation of stick-slip-stick-slip, in which the string sticks to the bow hair for the longest part of its period, slipping just once. Experiments show that simulated bowed strings have the same playability as real bowed strings as calculated by Schelleng [9].

Further experiments also show that the playability of virtual bowed strings increases when accurate friction models that account for the thermodynamical properties of rosin are taken into account [12].

This above mentioned definition of playability is interesting from a musical acoustician's point of view but does not

*Current address: VCCM, McIntyre Department of Music, University of Virginia, Charlottesville, VA.

reflect performance issues. In these experiments, in fact, the input parameters that drive the bowed string model corresponding to the right hand of the player are kept constant for each simulation. This is a situation that is clearly not the same as that which occurs in violin performances, as in performance it is the continuous evolution of the input parameters that constitutes the nuances that are the characteristics of an expressive performance. In order to address this issue, Askenfelt [1] studied the contribution of bowing parameters in different bow strokes, trying to determine the physical limits of the input parameters in order to achieve a specific stroke. He determined the maximal duration of the pre-Helmholtz attack allowed in order to judge a particular stroke as acceptable.

In interactive performances, other issues related to playability must be considered. Sile O'Modhrain [7], for example, studied the influence of tactile feedback on the playability of virtual bowed strings.

She discovered that haptic feedback greatly increases the playability of virtual instruments. In this context, playability is referred to as the ability of bowed string players to consistently perform different bow strokes that produce violin sounds that are judged as perceptually acceptable by professional bowed string players and also have waveforms that reside within the playability region as defined by [14].

Another important issue in virtual musical instrument playability refers to the precision and accuracy of both the hardware and the software that comprise the virtual instrument itself.

In the situation where a controller is used to drive a synthetic model, we may say that a controller is precise if it allows the player to control subtle variations of the gesture parameters with great care, and we may say that it is accurate if the data collected by the device may be easily correlated to a real physical values of a gesture. Correspondingly, we may say that the model is precise if it reproduces the nuances in the sonorities that are produced by a real instrument, and it is accurate when the physical input/output data can be matched to measurements on real acoustic instruments. An issue of importance in considering both of these aspects is that of latency, which may be adversely affected by limitations of either precision or accuracy. Of course, in musical performance, minimizing latency is a priority.

Obviously, the acceptance level of responsiveness varies according to the instrument played. For example, percussion instruments require a higher responsiveness than woodwind instruments. In general, it is reasonable to assume that transient instruments require a higher level of responsiveness that sustained instruments.

In the case of the bowed string, the issue of responsiveness is crucial when fast bow strokes such as staccato, balzato, martellato are performed.

Another more general definition of playability is the ability of the virtual instrument to be ergonomically playable. That is, the player should be able to physically manipulate the interface freely and with ease.

A chart that summarizes all the playability issues mentioned above is shown in figure 1.

## 3. NEW EVALUATION PLAYABILITY

In this paper we are interested in exploring all the previous definitions of playability and extending them by driving a violin bowed string physical model using a bow controller.

Rather than building or adopting an alternate controller or working with a haptic device such as in the examples

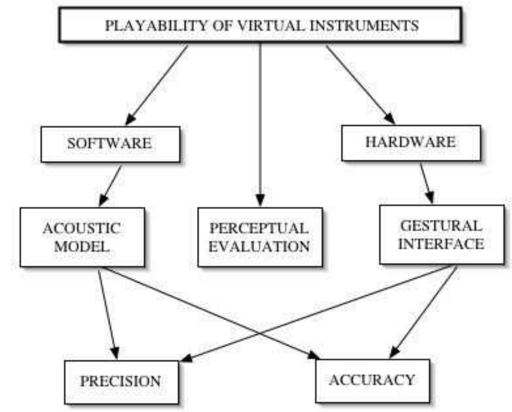



**Figure 1: Playability chart of a virtual musical instrument**

mentioned above, we are interested in exploring the possibility of reproducing traditional bowing techniques using a bow controller that behaves in a manner as closely related to that of a traditional violin bow as possible.

This allows us to validate both the model and the controller by comparing it to the behavior of the traditional instrument.

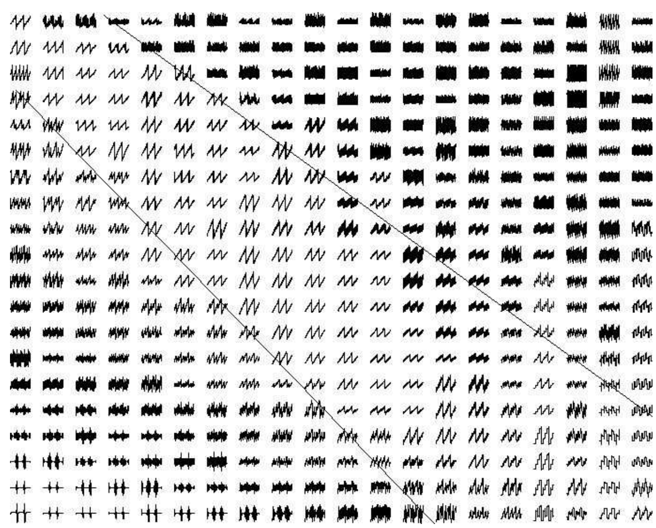


**Figure 2: Playability plot obtained by capturing the waveforms after steady-state motion is obtain. Horizontal axis: bow force. Vertical axis: bow position.**

### 3.1 A bowed string physical model

We built a bowed string physical model that combines waveguide synthesis [13] with latest results on bowed string interaction modeling [8].

A schematic representation of the model is shown in figure 3. In this model, the bow excites the string in a finite number of points, which represent the bow width. The frictional interaction between the bow and the string is modeled considering the thermodynamical properties of rosin [12]. The bow width is modeled by discretizing the region of the string in contact with the bow using finite differences and calculating the coupling between the waves propagating along the string and the frictional interaction between the bow and the string at each point. Once the velocity of the string at the contact point has been calculated, the waves propagating along the string are modeled using digital waveguides. More precisely, transversal and torsional waves propagating toward the bridge and the fingerboard are modeled as pairs of one dimensional digital waveguides.

The outgoing velocity at the bridge is filtered through the violin body's resonances and corresponds to the output waveforms perceived by the listener.

For a detailed description of the physical model see, for example, [11].

The input parameters of the model corresponding to the right hand of the player are bow position relative to the bridge (normalized between 0 and 1, where 0 corresponds to the bridge, 1 corresponds to the nut, 0.5 corresponds to the middle of the string), bow pressure, bow velocity, and amount of bow hair in contact with the string. The model has been implemented as an external object in the Max/MSP environment.

Figure 2 shows the waveforms obtained by running the model with a constant bow velocity of 0.05 m/s, and varying the bow force and bow position between 0 and 5 N and 0.01 and 0.4 respectively. The waveforms are captured after a steady-state motion is achieved. Inside the two straight lines appears the playability region as measured by Schelleng with the same parameter configuration. Note how the synthetic model and the real instrument are in good agreement concerning the playability region's results.

## 3.2 The bow controller

The bow controller provides three different types of measurement that reflect the nuances of gesture that may be observed in traditional string instrument bowing technique. The controller's sensing system employs commercial MEMS accelerometers to measure three axes of acceleration, electric field sensing to track bow position and bridge distance from the bowing point, and foil strain gauges to detect changes in the downward strain of the bow stick as well as in the orthogonal direction (toward the scroll).

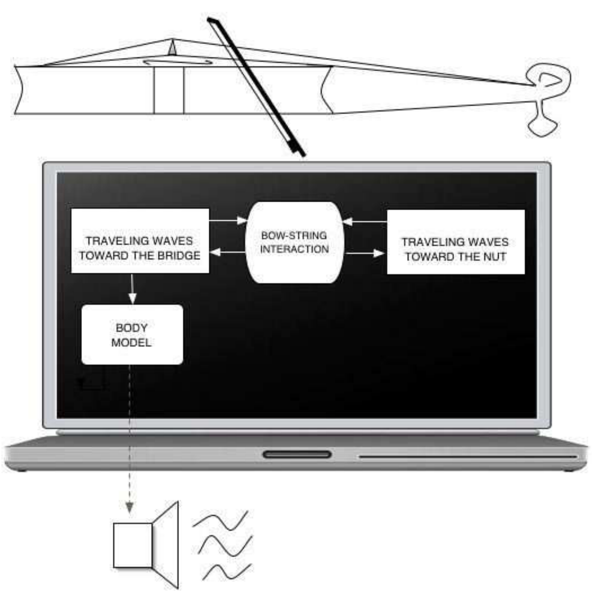



**Figure 3: A bowed string instrument and the corresponding simplfied block diagram of its model.**

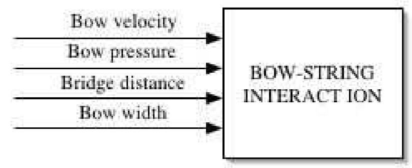



**Figure 4: Input parameters of the bowed string physical model.**

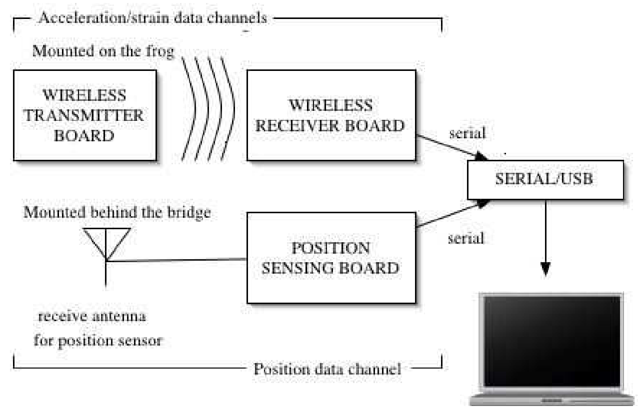



**Figure 5: Data flow for the violin controller.**

These sensors were chosen so as to offer a player and composer the ability to capture data concerning all of the parameters of bowing that contribute to the interaction between the bow and the string: bow speed and bowing point (from the position tracking), downward force and bow width (both reflected by strain sensors). Additionally, the accelerometers were included as a means of observing with great precision changes in bowing direction as well as the differences in characteristics of various kinds of string attacks.

The placement of the sensors and the accompanying electronics were carefully designed so as to provide good results in measurement, while also maintaining as completely as possible the balance, weight, and feel of the bow. Also a priority in this implementation is the protection of the hardware itself and robustness of the overall system, as the interface was intended for use in live performance and rig-

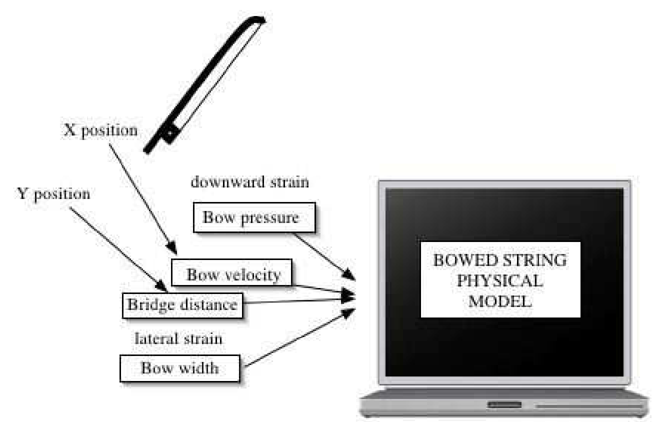



**Figure 6: One to one mapping of the bow controller to the bowed string physical model.**

orous laboratory experiments.

The strain sensors are effectively integrated onto the composite material of the bow, as they are permanently adhered around the midpoint of the stick. They are protected from wear by a thin layer of flexible tubing, and their connecting wires run down the lower length of the bow to the electronics board mounted on the frog (while still allowing the frog to slide freely in order to adjust the tension of the bow hair). This electronics board houses the accelerometers, the microcontroller, and the wireless transmitter, which sends the acceleration and strain data to a remote receiver board containing a serial port. This board also sends two separate signals to either end of a resistive strip that runs the length of the bow stick, acting as an antenna for the position measurement. The resultant mixed signal is received by an antenna mounted behind the bridge of the violin, and this signal is connected to another electronics board that determines the different amplitudes of the two received signals.

The mechanical layout of the electronics on the bow allows the player to use traditional right hand bowing technique, while keeping the hardware out of harm's way from the players hands and the strings of the violin. The electronics add about 30g of weight to the original carbon bow, but because of the careful distribution of the weight along the length of the bow the balance point of the bow is still well within the normal range for a traditional violin bow. In addition, the remote electronics boards are small and light, and the bow is wireless and runs on a camera-style battery with a lifetime of over 20 hours, and so the interface is highly portable.

The data from the bow controller is carried by two separate serial buses: one for the acceleration and strain data and one for the position data. In this experiment, a commercial serial/USB converter is used to connect the two serial lines with the input of a Macintosh computer [15].

## 4. MAPPINGS

In order to build an expressive virtual musical instrument, the capture of the gesture of the performance is as important as the manner in which the mapping of gestural data onto synthesis parameters is done. In the case of physical modeling synthesis, a one-to-one mapping approach of control values to synthesis parameters makes sense, due to the fact that the relation between gesture input and sound production is often hard-coded inside the synthesis model [4]. Because both the physical model and the bow controller are developed according to physical input and output parameters, the mapping between the two is straightforward. Figure 6 shows how the data sent by the bow controller is mapped to the input parameters of the physical model. Downward bow strain of the controller is directly mapped into bow force in the physical model. Bow velocity and bow-bridge distance are captured by measuring the horizontal and vertical position of the bow respectively. Moreover, lateral strain sensors are mapped into the amount of bow hair in contact with the bow.

## 5. PRELIMINARY EXPERIMENTS

The first experiments completed to begin the integration between the bow controller and the violin physical model were encouraging. With the model parameters of bow-bridge distance, bow velocity, and bow width for a fixed frequency held constant, we mapped the downward strain data from the bow controller to the downward force parameter. While applying pressure on the strings of our test violin (without actually drawing the bow across the strings) using the bow controller, we were able to quickly produce sonorities from the model that sounded appropriate for the amount of pressure we applied to the bow.

We also experimented by drawing the bow across damped strings to determine the response for changes in downward force that occur during different bow strokes. Again, the response of the model and the feel of this simple mapping produced a satisfying interaction and was interpreted by us as extremely promising for continued work.

## 6. CONCLUSION

The initial experiments for this research project yielded positive results. Players were able to produce bow strokes that felt natural and were judged as acceptable from bowed string players. Furthermore, the mapping between the amount of force and the change in amplitude of the sound seemed intuitive.

Mapping each of the remaining input parameters of the physical model to the appropriate sensor data and aim to iteratively build a convincing link between physical controller and virtual violin. This allows us to evaluate each component of the system as well as the overall system by examining all aspects of playability discussed.

Using this methodology we are able to validate not only the acoustical properties of our system, but also the performance capabilities, issues which are often treated separately.

## 7. REFERENCES

[1] A. Askenfelt. Measurement of the bowing parameters in violin playing. *Journal of the Acoustical Society of America*, 86(2):503–516, August 1989.

[2] C. Chafe. Simulating performance on a bowed string. *Technical Report*, STAN-M-48, May 1988.

[3] J.-L. Florens and C. Henry. Bowed string synthesis with force feedback gesture interaction. In *Proc. International Computer Music Conference*, pages 115–118. ICMA, 2001.

[4] A. D. Hunt, M. Paradis, and M. M. Wanderley. The importance of parameter mappings in electronic instrument design. In *Proc. NIME*, 2002.

[5] D. Jaffe and J. Smith. Performance expression in commuted waveguide synthesis of bowed strings. In *Proc. International Computer Music Conference*, pages 343–346. ICMA, 1995.

[6] C. Nichols. The vbow: A virtual violin bow controller for mapping gesture to synthesis with haptic feedback. *Organized Sound*. In press.

[7] S. O'Modhrain, S. Serafin, C. Chafe, and J. Smith. Qualitative and quantitative assessment on of a bowed string instrument. In *Proc. International Computer Music Conference*. ICMA, Aug. 2000.

[8] R. Pitteroff. Mechanics of the contact area between a violin bow and a string. part i:reflection and transmission behaviour. part ii: Simulating the bowed string. part iii: Parameter dependance. In *Acustica-Acta Acustica*, pages 543–562, 1998.

[9] J. C. Schelleng. The bowed string and the player. *Journal of the Acoustical Society of America*, 53(1):26–41, Jan. 1973.

[10] S. Serafin, R. Dudas, M. M. Wanderley, and X. Rodet. Gestural control of a real-time model of a bowed string instrument. In *Proc. International Computer Music Conference*. ICMA, Oct. 1999.

[11] S. Serafin, J. O. Smith, III, and J. Woodhouse. An investigation of the impact of torsion waves and friction characteristics on the playability of virtual bowed strings. New York, Oct. 1999. IEEE Press.

[12] J. H. Smith and J. Woodhouse. The tribology of rosin. *J. Mech Phys.Solids*, 48:1633–1681.

[13] J. O. Smith. Physical modeling using digital waveguides. 16(4):74–91, Winter 1992.

[14] J. Woodhouse. On the playability of violins. Part I: Reflection functions. Part II: Minimum bow force and transients. *Acustica*, 78:125–136,137–153, 1993.

[15] D. Young. The hyperbow controller: Real-time dynamics measurement of violin performance. In *Proc. NIME*, 2001.

# Sonic City: The Urban Environment as a Musical Interface

Lalya Gaye
Future Applications Lab
Viktoria Institute
Box 620, 405 30
Göteborg, Sweden
+46-(0)31-7735562
lalya@viktoria.se

Ramia Mazé
Play Studio
Interactive Institute
Hugo Grauers gata 3, 411 33
Göteborg, Sweden
+46-(0)705942932
ramia.maze@tii.se

Lars Erik Holmquist
Future Applications Lab
Viktoria Institute
Box 620, 405 30
Göteborg, Sweden
+46-(0)31-7735533
leh@viktoria.se

## ABSTRACT

In the project Sonic City, we have developed a system that enables users to create electronic music in real time by walking through and interacting with the urban environment. We explore the use of public space and everyday behaviours for creative purposes, in particular the city as an interface and mobility as an interaction model for electronic music making. A multi-disciplinary design process resulted in the implementation of a wearable, context-aware prototype. The system produces music by retrieving information about context and user action and mapping it to real-time processing of urban sounds. Potentials, constraints, and implications of this type of music creation are discussed.

## Keywords

Interactive music, interaction design, urban environment, wearable computing, context-awareness, mobility

## 1. INTRODUCTION

*Sonic City* is a novel interface for musical expression through interplay with the urban environment. Unlike the majority of work in this domain, which tends to focus on concert-based performance, this project promotes musical creativity integrated into everyday life, familiar places and natural behaviours [19].

We describe the development and first implementation of a wearable system that creates electronic music in real time, based on sensing bodily and environmental parameters. Context and user action are mapped to sound processing parameters and turn live concrete sounds into music. Thus, a personal soundscape is co-produced by a user's body and the local environment simply by walking through the city. Considering the city as an interface and mobility as musical interaction, everyday experiences become an aesthetic practice. Encounters, events, architecture, weather, gesture, (mis)behaviours – all become means of interacting with, appropriating, or 'playing the city'.

In this paper, we first introduce our approach to the city and musical interaction. We then outline the development process together with design methods and issues and describe the resulting implementation of a first prototype. Finally, we reflect on the project and discuss potentials and constraints inherent in the system and this type of music creation.

**Figure 1: Sonic City enables users to interactively create music by walking through a city**

## 2. THE CITY AS INTERFACE

The city has long been an inspiration and site for musical expression, whether as a metaphor in classical composition, a source of rhythms and sounds in jazz and electronic music, a stage for street performance or the cradle of a walkman generation. Music is inextricable from the lifestyles and textures of daily urban life. It is also an accessible and well-established form of public aesthetic expression available to everyone. Indeed, making or playing music has been a means of re-appropriating public space for localised concerns, whether as community expression or even as a form of protest (e.g. [22]). Therefore, we believe that the city offers tremendous possibility for personal musical expression and creativity.

Everyday urban experience involves active interpretation and impels creative response – consider the meaning of a screeching noise, the smell of burning rubber and a car headed your way! As a 'physical interface', the city provides a built infrastructure and established ways of using it creatively. Even the mundane act of taking a walk involves the complex co-production of bodily movement in relation to obstacles. Along the way, there are always elements of serendipity: an unexpected view, surprising encounters or fleeting ambiances. Built and transient conditions require continual tactical choices and inspire possibilities along the way. Whether a pleasant stroll or a mundane commute, being in the city involves dynamic creative improvisation.

Use of the physical city is conditioned by our own perceptions, habits, histories, and emotions. Terms such as *mental map* and *psycho-geography* in urban theory describe the special image of the city we each have, characterised by informal landmarks, subjective distances and sizes, and intuitive way-finding (e.g. [1], [13]). Activities such as skateboarding and parkour exemplify the highly personal ways in which we perceive and use the city, in this case physical or acoustic appropriation of the built environment for personal expression [2, 18]. The built, narrative, and emotional landscape of the city is an established topic in everyday as well as aesthetic practices such as performance and sound art, and soundscape composition [25].

In this project, we take the simple act of walking to explore the city as an interface and opportunity for personal creativity. Everyday behaviours, personal (mis)uses, and aesthetic practices suggest the inventive ways in which people already use the physical city. As a new platform for personal expression and urban experience, Sonic City explores public space as a site for private performances and emerging behaviours, and the city as an interface for personal musical expression.

## 3. MOBILITY AS INTERACTION

Urban environments are often places of transit, where people are constantly mobile. They adopt appropriate behaviours for public situations, use portable technologies, navigate and make decisions on the fly. Being in the city implies a dynamic shifting among heterogeneous contexts and behaviours. This, and recent developments in mobile and context-aware computing, prompted us to consider the use of mobility itself as a means of interaction in electronic music making. We see mobility in the city as a large-scale version of gesture-based interaction combined with context-awareness, which can be exploited for music creation.

### 3.1 Gesture and Context

Gesture is generally defined as "a specific movement from part of the body, executed or not in a conscious way, applied or not to a device, that can accompany a discourse or have a meaning by itself" [24], leading – in the context of musical interface research – to the musical output of an interactive system. A large amount of work has been done in the field of electronic music and dance technology (e.g. [11, 23]).

Context-awareness has been defined as the ability of a device or application to "adapt according to its location of use, the collection of nearby people and objects, as well as changes to those objects over time" [20] or to "monitor changes in the environment and adapt [its] operation according to predefined or user-defined guidelines" [9]. Primarily applied human-computer interaction, mobile and ubiquitous computing, this property can also be used to output music: in ATR's Sensor-Doll project [26], the musical results of a user's interaction with a doll differs depending on context.

### 3.2 Mobility

From our point of view, mobility can be seen as physical movement extended spatially, over time, and through multiple contexts. This is exemplified in the simple act of walking through the city as a sequence of contexts experienced over time and shaped by dynamic urban conditions and personal choices of route.

If we consider the act of walking in the same way as we do gesture – as a deliberate and creative action – then the movement of a pedestrian through the surrounding environment can be modelled as a combination of gesture-based interaction and context-awareness. Rather than applying both of these in parallel, we are applying a model that correlates them (see Figure 2), such that it is the interaction between the user and the city that generates music.

**gesture** user action → output

**context-awareness** context → output

**vs.**

**mobility** user action + context → output

**Figure 2: Mobility as interaction**

## 4. RELATED WORK

Sonic City is related to other projects dealing with urban settings and sound or musical expression. Projects involving the city in interaction include the *Citywide Performance* project [6], an urban mixed-reality game event, and *Sound Mapping* [16], a site-specific outdoor interactive music event with portable sensor-based devices. The *Touring Machine* [8] uses location-awareness to supplement real space with a virtual information overlay. *Pirates!* [3] uses proximity and location in real space as interaction elements in virtual game play. *Noiseman* [7], *Sonic Interface* [14] and *Nomadic Audio* [15] propose new interactions with urban sound: Noiseman and Sonic Interface filter and mix urban sounds on the move, and Nomadic Audio creates a dynamic soundscape from local radio frequencies. *CosTune* [17], a networked wearable musical instrument, and *Sensor-Doll* [26] have been developed by ATR for communicative and social purposes.

The goals of Sonic City vary from these projects in terms of expressive genre, sound qualities, and interaction due to differences in intention and research questions. Sonic City belongs to neither a performance, game or communication genre, focussing instead on personal expression and everyday creative use of public space. In this way, it differs from the aforementioned projects of ATR that use music to support social communication and the gaming examples that specify rules, goals, and duration of the experience. Where the focus of the Sound Mapping project is on a performance event, taking place within a restricted area, Sonic City is meant for everyday use by anyone, anywhere.

Sonically, our project is greatly inspired by soundscape composition [25]. However, where this relies on pre-recorded sounds (as most sound-based projects mentioned above) and is listened to out-of-context, sound content in Sonic City is linked directly to the physical location where it is being produced and heard. Urban sounds are transformed into a real-time personal soundscape, as an overlay to the actual acoustic surroundings.

In terms of interaction, music in Sonic City is co-produced by both the listener and the city. Citywide Performance, Costune and Touring Machine projects treat the city is a setting for rather than a participant in interaction. In Noiseman, Sonic Interface and Nomadic Audio, a listener interacts with the urban soundscape using a tangible or visual interface on a handheld device. In Sonic City, conditions of the body and the environment contribute jointly to music creation, shifting the focus to the city itself as an interface and direct physical engagement with the city as interaction.

# 5. DESIGN PROCESS & DEVELOPMENT

The goal of Sonic City is to provide a mobile musical experience for a wide range of users in a variety of urban environments. Thus, the system developed in the project needs to be location-independent and robust enough to be used outdoors and on a daily basis. Rather than relying on a fixed infrastructure, such as sensors deployed in the environment, we opted for an entirely wearable solution in order to support user mobility. Starting with these premises, we then explored a variety of possibilities for what Sonic City could be like.

Essential questions we dealt with were:

- What from the user and the city would be interesting as input?
- How should the music sound?
- What is the amount and nature of user control?
- How should inputs be mapped to music output?

In order to address these and to gain insight in an ongoing manner from users and relevant experts, we have applied a multi-disciplinary and iterative development process. Backgrounds of core team-members include engineering, interaction design and architecture. During the project, we have collaborated with a sound artist, a sociologist, a product designer, and a cognitive scientist. User-centred design methods such as ethnographic studies, scenarios, and workshops have provided insight into the user experience, enabling us to develop a system and sounds that would be interesting for extended use and a wide range of musical expertise.

## 5.1 Input Parameters

In order to determine interesting input parameters, we re-examined characteristics of walking in the city and carried out some limited ethnographic studies. We conducted stationary observations of specific sites and documented paths of pedestrians with action logs (see Figure 3). This gave us insight into relevant and interesting aspects to sense and helped to imagine sequences of actions, events and ambiances along a walk as a potential composition. Observations of specific sites uncovered essential patterns of action, for example behavioural sequences at crosswalks (e.g. glancing, changing course and speeds). Obstacles such as stairways were interesting conjunctions of fixed and mobile elements, including structural elements (step patterns and railings) and pedestrian behaviour (styles of climbing stairs, congestion, and turn-taking).

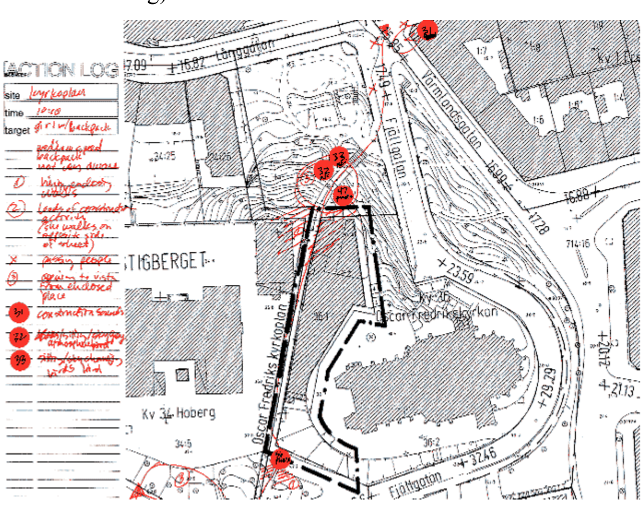



**Figure 3: Example of an action log**

From the observations, characteristics of pedestrians and surroundings were categorised in terms of action and context. High-level descriptions, such as 'indoors' and 'crossing the street', were broken down into measurable cues that the system could use for context and action recognition. From this, possible input parameters from sensors emerged:

- *Body-related input:* heart rate, arm motion, speed, pace, compass heading, ascension/descent, proximity to others/objects, stopping and starting
- *Environment-related input:* light level, noise level, pollution level, temperature, electromagnetic activity, enclosure, slope, presence of metal

Some types of input involved a range of continuous values fluctuating over time, e.g. the outside temperature or a pedestrian's heart rate. Other types, for instance a car horn, only occurred momentarily, in a way that could be described as discrete (see Table 1).

This gave us a framework for making decisions about sensing and retrieval in Sonic City. In terms of choosing sensors, some seemed more relevant than others when confronted with the opinion of potential users and were therefore prioritised in the implementation phase.

**Table 1. Characteristics of low-level input parameters**

| | **Body** | **Environment** |
|---|---|---|
| **Discrete factors** | sudden change in user action (ex: stopping) | localised events (ex: a car passing) |
| **Continuous factors** | physiological state (ex: heart rate)<br>actions over time (ex: compass heading) | evident ambiances (ex: level of light)<br>invisible ambiances (ex: pollution level) |

## 5.2 Sound Design

The sound design needed to be consistent with how people already perceive and experience the environment of the city. With this in mind, we worked with a sociologist (Magnus Johansson) to develop hypothetical scenarios of user experiences, values, and taste. The scenarios were based on potential users that we knew or interviewed. They were deliberately extreme in order to represent a wide range of possibilities and design implications. Besides helping to determine the amount and nature of user control supported by the system (see the paragraph on control), they revealed differing personal relationships with the city. Specifically, we considered peripheral versus foreground aspects of the experience and musical possibilities ranging from ambient to rhythmical. Based on the scenarios, we defined the boundaries of the sound design space (see Figure 4).

We were interested in maintaining a close experiential relationship between the sound content and the context of music creation – namely the existing city soundscape. Thus, we decided to use real-time audio processing of urban sounds as a basis for the sound design. In order to develop interesting sound content from an artistic point of view, we have been working in close collaboration with the sound artist Daniel Skoglund of 8tunnel2 [27], and have been inspired by the musical genres of soundscape composition [25] and glitch [5]. Designed possibilities cover all four quadrants of the design space.

Interesting processing parameters emerging from the sound design process were abstracted according to the kind of musical impact they would have on the output. They were classified into:

- *Structural composition variables,* relative to the number of sound layers and the temporal structure of the music (f. ex., if making an analogy with pop music, a change from a couplet to a chorus)
- *Spectral variables,* which determine the quality of each sound (their timbre, envelope, etc.)
- *Triggering* of short musical events

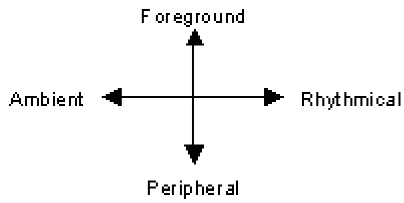



**Figure 4: Sound design space**

## 5.3 Control

When considering questions of user perception and control over the music, we asked ourselves how 'in charge' of the experience a user should feel:

- Should there be means for *explicit control* over the sound, such as buttons, in case a user would not obtain the desired music just through interaction with the city?
- What degree of *randomness* could be built in the system to maintain interest? In situations of unvarying sensor input values over long periods of time, should the music remain exactly the same? For everyday use of Sonic City, how similar could the same walk sound day after day without becoming boring?
- What should the *balance* be between the influence of user and environmental factors? How would 'invisible' factors (whether sensor-based such as pollution or processing-based such as randomness) be perceived?

The same scenarios of use as those mentioned in the preceding section were used to explore potential design directions. Then, we were able to define a control space (see Figure 5) that described the territory of possibilities and located the scenarios in relation to one another.

Two axes describe the predominant factors influencing the music. The vertical axis shows the balance of body or user input versus environmental or city input. To illustrate, *Jonas* is a sound engineer and thinks about music in a highly structured and systematic way. He would want a high degree of control over the music and its sound qualities and even be able to add or customise means of input. *Agnes*, in contrast, would only want the system to monitor tiny variations in the environment and is not interested in controlling the sounds herself.

The horizontal axis describes the span of possibilities from unpredictability to user-deterministic control. To illustrate, *Maria* roams the streets of her city at night as a form of escape. She does not go far and often takes the same path, but would want the music to modulate dramatically and vary each time, implying the introduction of randomness on system level. *Jean*, on the other hand, is a participant in the extreme sport of climbing urban structures. Each climb is like a conquest and happens only once. He would use Sonic City to monitor his body's engagement with each unique environment in a very direct way.

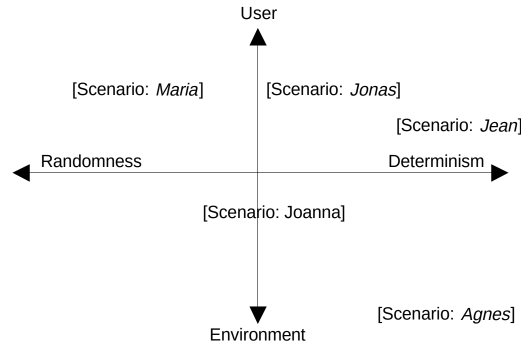



**Figure 5: Control space with user scenarios**

The scenario that we chose to implement was *Joanna*. Balancing both active engagement and urban discovery, Joanna would use Sonic City to re-discover her environment as a poetic and aesthetic practice. Representing the essence of our intentions with Sonic City, this scenario provides a foundation for testing other variables and possible experiences and is reflected in the mapping strategy.

## 5.4 Mapping Strategy

The mapping had to be both transparent to the user and complex enough to sustain interest if the system were to be used day after day. In our process, we took a top-down approach to mapping, starting with the essential concept of context. It has an intrinsically layered nature since a context can consist of several different levels of abstraction. This led us to the development of a layered mapping strategy similar to the "multiple layers" model [10]. In that model, input and output are each abstracted on a high level and are then linked together by a straightforward one-to-one mapping, while the low-level parameters that constitute these abstractions are actually cross-coupled.

We considered it essential that the mapping would reflect scales of time and distances covered while walking in the city and maintain the distinction between continuity and discreteness. Using the categories and abstractions of input and output described in previous sections, we developed the following mapping. The *high-level* abstraction of context and actions is mapped to structural composition parameters. The *low-level* discrete and continuous factors that make up the abstractions are also mapped directly according to their discreteness versus continuity, thus discrete factors trigger short musical events and continuous factors are mapped to spectral variables (see Figure 6). Within this general framework, decisions about details of the mapping were carefully made one at a time to insure coherence and pertinence.

We determined that the time it takes to go two steps (one pace) was a good updating period for context-recognition. If the tempo follows a user's steps, then the length of this pattern of action is comparable to that of half a bar, and structural composition variables reflect the natural rhythm of a walk. At the context and action recognition level, only changes lasting longer than this period of time are considered significant enough to be taken into account by the algorithms, differentiating, for example, a general rise in noise level from temporary noises such as car horns.

Generally speaking, an input value can affect both layers of abstraction (for instance, lighting intensity is a continuous factor that also impacts context), and its effect on the output depends on its length in relation to an update period. Many of the low-level sound processing parameters also belong to more than one abstraction. Therefore, as in the "multiple layers" strategy, values are highly cross-coupled and the mapping is fairly complex, which we hope will sustain interest.

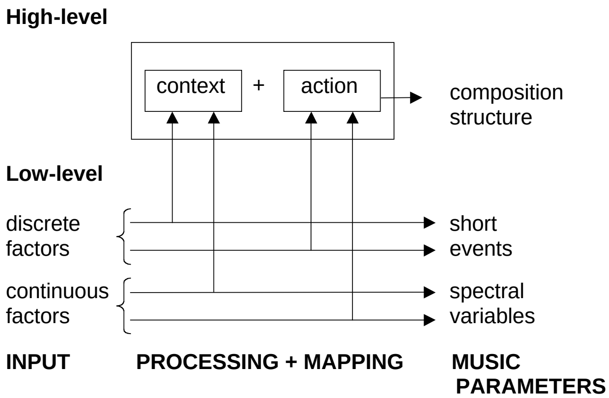



**Figure 6: Mapping strategy**

# 6. PROTOTYPE IMPLEMENTATION

We have implemented a prototype as a platform for iterative development and testing with users. For the sake of simplicity, only the most illustrative aspects of the design have been included in this current version. The prototype employs inexpensive and widely available components.

## 6.1 System Overview

The prototype consists of a set of environmental and biometric sensors, a BasicX-24 micro-controller, a USB-MIDI converter, a small laptop running the interactive music programming environment PD, a stereo microphone, and headphones (see Figure 7).

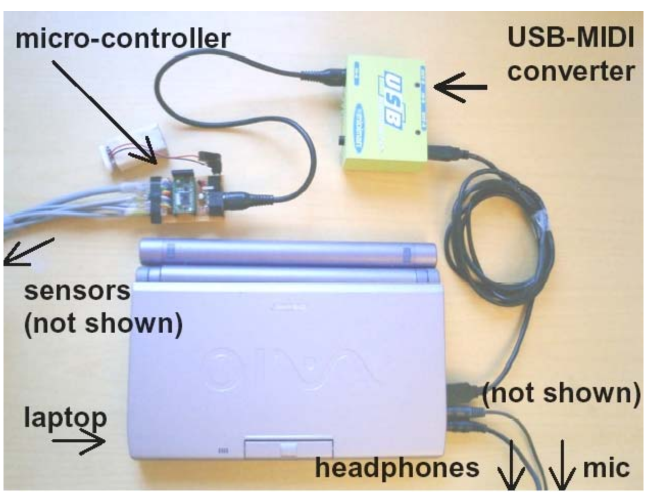



**Figure 7: Components of the prototype**

Sensor data is collected by the micro-controller and sent in MIDI format via the converter to the laptop. It is then converted and processed for context and action recognition. Resulting interaction parameters are mapped to musical parameters that shape the real-time audio processing of urban sounds captured by the microphone (see Figure 8).

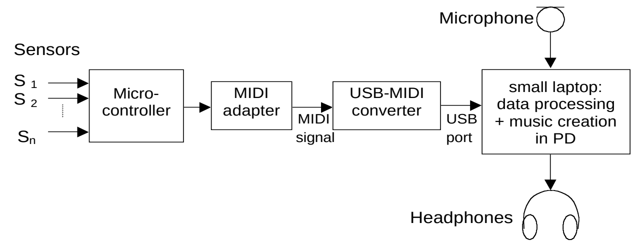



**Figure 8: Dataflow**

The hardware resides within the clothing worn on the user's body. Music is output through semi-open headphones, allowing the user to also hear the real environmental sounds, thus blending together both personal and existing soundscapes. For testing purposes, we have designed a one-size-fits-all adjustable jacket that incorporates all components of the prototype. While this is not a proposal for the final wearable, it allows for easy testing of sensor placement and orientations with multiple users in the city (see Figure 9).

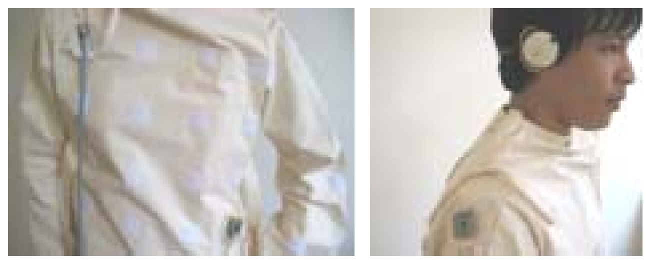


**Figure 9: Testing jacket. Left: multiple sensor placement options; Right: user wearing headphones and sensor**

## 6.2 Sensing and Retrieval

The sensors currently included are: a light-to-frequency converter, a sound gate, a metal detector, an accelerometer, a temperature sensor, and a pollution sensor. Two switch-buttons and a potentiometer knob are also included to support explicit interactions with the system, such as on/off. More sensors, measuring heart rate, ultrasound, electromagnetic activity, and atmospheric pressure, will be added later. Since each sensor affects multiple levels in the mapping, each interaction category currently has between three and six corresponding input possibilities.

The parameters that are identified and mapped to musical parameters in the current prototype are:

- *Contexts:* day vs. night, loud vs. quiet, cold vs. hot weather
- *Actions:* walking straight, left, right, running down stairs, standing still, pace
- *Continuous factors:* temperature, pollution level, lighting conditions
- *Discrete factors:* presence of metal, instantaneous noise level, jumps, steps, stops, starts

Context and action are retrieved inside the PD program, with "if-then" structured recognition algorithms. It is updated for every two steps made by the user.

## 6.3 Sound Processing

The output is shaped in real-time by sound processing objects such as filters, delay loops, envelopes, sampling, playback, mute, and echoes. The audio input can take manifold parallel and serial paths through these objects and can be deviated and redirected to other objects in a flexible way, which allows for a wide variety of output.

Structural composition variables consist of possible paths for the sound input through temporal processing (e.g. delays), the activation of processing channels, and the overall tempo. Spectral variables are determined by filters and feedback loops. Examples of possible triggered events include doubling the tempo, muting the rhythm, or triggering a sample.

## 6.4 Mapping

Input and sound parameters are mapped according to the layered strategy described in the design section. The following aspects have been implemented:

- The light intensity (continuous factor) currently determines the cut-off frequency (spectral variable) of filters that are opened and closed rhythmically. When walking under a shadow for a long time (context), this intensity also begins to influence the Q-value (spectral) of these filters.
- On top of a beat with the same tempo (structural variables) as the user's pace (action), layers of rhythm with tempos that are multiples of that of the first layer are triggered (musical event) when the sound input level is over certain thresholds (discrete factor). The dynamics of noises in urban settings can make the rhythm sound very organic. In a generally loud environment (context), the tempo is twice as fast (structure) as in a quiet context.
- If the user stops walking briefly (discrete), the rhythm is muted just as quickly (event), and if this stop lasts longer than the duration of the preceding two steps (action), everything is muted. The system requires active walking to function.
- Sudden proximity to metal (discrete) triggers filtered echoes (musical event), the delay of which (spectral) depends on the pollution level (continuous).
- At night (context), filtered (spectral) samples recorded randomly are played back and echoed (musical event) when sudden flashes of light (discrete), such as streetlights, are sensed.

We are also in the process of implementing an overall compositional structuring of the music based on metaphors of record spinning practiced by DJs.

# 7. DISCUSSION

## 7.1 The City as Interface

When reflecting on how we thought of the city in the project, what emerges is not only how we see the city as a means to an end – an interface to create music – but also as an end in itself. As an interface, the city is both a setting and the means of music creation. Fixed and fluid conditions in the city (for instance, architectural elements and traffic flows) structure user mobility. Perception of these is accomplished by the system through the range of sensors and formulation of input parameters. Thus, the city is a means for a user to interface with and control music creation – 'playing' the city as a musical instrument. Context parameters and urban sounds are necessary inputs to the system – the city not only enables music creation but music creation cannot happen without the city. In these ways, the system leverages existing urban experience to produce a new one. Our hope is that Sonic City not only encourages personal soundscape creation but also enhances perceptions of the city and possibilities for urban discovery.

## 7.2 Mobility as Interaction

In exploring mobility as a technique for interaction, we designed and implemented a mapping in which user action and context are combined rather than treated in parallel, and low-level city and user input is cross-coupled with sound parameters. Thus, we argue that we have achieved a successful interweaving of user action and context, which is intrinsic to our mobility model. In this respect, Sonic City approaches gestural interaction in a novel way. Mobility as interaction should promote natural behaviours and existing mobile lifestyles as sources of new creativity in everyday life.

## 7.3 Musical Expression

The sound design and mapping reflect our approach to personal musical expression in Sonic City. Besides achieving a close experiential link between the music and the existing soundscape, opting for real-time sound processing ensures an experience of use that will be continually evolving since the sounds around us are always different. Considering that every person has his or her own idiosyncratic behaviours and favourite routes, this means that the musical experience will always be personally expressive and responsive to our situations and choices. Along with carefully crafting the sounds with a sound artist, we hope that these choices will help to achieve ongoing interest in the music itself and a satisfying experience day after day.

# 8. CONCLUSIONS & FUTURE WORK

We have described the *Sonic City* project, which involves the multi-disciplinary design and implementation of a wearable system for music creation that treats the urban environment as an interface and mobility as interaction. Sonic City supports personal expression and transforms everyday behaviours into creative practice through natural and playful means of interacting with the urban environment.

Our next step will be to evaluate different aspects of the project. With the current prototype, we are able to test technical aspects in context and determine short-term user experience (e.g. musical satisfaction, choice of sensors, sense of control). In addition to the one we have had to date, future user workshops will continue to help us evaluate conceptual aspects and address lifestyle and aesthetic questions. We believe that having both a prototype and illustrative scenarios of use is necessary to make the project less abstract for people who are encountering this type of interactive technology for the first time. Based on the feedback we will be collecting in the next 2 months, we will draw conclusions and derive implications for refining the prototype.

However, for testing aspects of everyday, long-term use and misuse, the current prototype is too limited. It has too many wires and hardware components to carry around, a noticeable latency in sound processing and control, and is not modular or robust enough. Therefore, we plan to develop a new prototype, which will also incorporate additional functionality. We are considering switching to the *Smart-Its* platform [20]. Smart-Its are wireless, small-scale computing devices with built-in sensors, computation, and ad-hoc network communication. They are thus modular enough to be re-combinable by the user, and they enable distributed computing of context and action recognition at the device network level, leaving sound processing as the only task to be performed by the computer (preferably hand-held). Once this next prototype implemented, we will use cognitive science methods [12] to evaluate aspects of way-finding behaviours, long-term satisfaction, evolution of skill, emerging behaviours, and how Sonic City could fit into everyday lifestyles of city dwellers.

## 9. ACKNOWLEDGMENTS

We wish to warmly thank Daniel Skoglund, Magnus Johansson, and Margot Jacobs for their valuable contribution to the concept development and design process, as well as Sara Lerén for her input on evaluation methods. This project is funded by the Swedish Foundation for Strategic Research through the *Mobile Services* project, and the *Public Play Spaces* project platform.

## 10. REFERENCES

[1] Andreotti, L. and Costa, X. Theory of the Dérive and Other Situationist Writings on the City. Museu d'Art Contemporary de Barcelona, Spain, 1996.

[2] Borden, I. Skateboarding, Space and the City: Architecture and the Body. Berg Publishers, Oxford, 2001.

[3] Björk, S., Falk, J., Hansson, R. and Ljungstrand, P. Pirates! – Using the Physical World as a Game Board. In Proc. of Interact'01 (Tokyo, Japan, 2001).

[4] Careri, F. Walkscapes: Walking as an Aesthetic Practice. Land&Scape Series, Gustavo Gili, Barcelona, Spain, 2002.

[5] Cascone, K. The Aesthetics of Failure: "Post-Digital" Tendencies in Contemporary Computer Music. In Computer Music Journal, Vol. 24, Issue 4, 2000.

[6] Citywide Performance project. http://www.equator.ac.uk/projects/citywide

[7] Dunne, A. Noiseman, http://www.dunneandraby.co.uk

[8] Feiner, S., MacIntyre, B., Hollerer, T. and Webster, T. A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment. In Personal Technology 1(4), 1997.

[9] Fickas, S., Kortuem, G. and Segall, Z. Software Organization for Dynamic and Adaptable Wearable Systems. In Proc. of ISWC'97 (Cambridge, MA, USA, 1997).

[10] Hunt, A., Wanderley, M. and Paradis, M. The Importance of Parameter Mapping in Electronic Instrument Design. In Proc. of NIME'02 (Dublin, Ireland, 2002).

[11] Interactive Systems and Instrument Design in Music. http://www.computermusic.org/interactivesystems/wg

[12] Lerén, S. How to Evaluate Sonic City. Evaluating Context-Aware Computing for Creative Purposes from a Cognitive Science Point of View. Candidate Thesis, IT-University, Göteborg, Sweden, 2002.

[13] Lynch, K. The Image of the City. MIT Press, Cambridge, MA, USA, 1960.

[14] Maebayashi, A. Sonic Interface. http://www2.gol.com/users/m8/msonic.html

[15] Mazé, R. Nomadic Audio, Computer Related Design project, Royal College of Art, 2000. http://www.viktoria.se/~ramia/portfolio

[16] Mott, I. and Sosnin, J. Sound Mapping, an Assertion of Place. In Proc. of Interface (1997).

[17] Nishimoto, K., Maekawa, T., Tad, Y., Mase, K. and Nakatsu, R. Networked Wearable Musical Instruments Will Bring a New Musical Culture. In Proc. of ISWC'01 (Zurich, Switzerland, 2001).

[18] Parkour. http://perso.wanadoo.fr/parkour

[19] Redström, J. The DishJockey: Integrating Multimedia into Everyday Activities. In Proc. of the 6th ACM International Multimedia Conference (1998).

[20] Schilit, B., Adams, N. and Want, R. Context-Aware Computing Applications. 1st International Workshop on Mobile Computing Systems and Applications (Santa Cruz, CA, USA, 1994).

[21] Smart-Its. http://www.smart-its.org

[22] Tagg, P. Reading Sounds or an Essay on Sounds. Music, Knowledge, Rock & Society, In L'Unità, Italy, 1987.

[23] Wanderley, M. and Battier, M. (eds.). Trends in Gestural Control of Music. IRCAM, Centre Pompidou, Paris, France, 2000.

[24] Wanderley, M. Movements and Gestures. http://www.ircam.fr/equipes/analyse-synthese/wanderle/Gestes/Externe/Sensor_Feedb/node5

[25] Westerkamp, H. Soundscape Composition: Linking Inner and Outer Worlds. In Soundscape Newsletter, Amsterdam, Holland, 1999.

[26] Yonezawa, T., Clarkson, B., Yasumura, M. and Mase, K. Context-Aware Sensor-Doll as a Music Expression Device. In Proc. of CHI'01 Extended Abstracts (Seattle, USA, 2001).

[27] 8Tunnel2. http://217.215.145.193/brax-tone/swebrax/8T2.HTM

# Designing, Playing, and Performing with a Vision-based Mouth Interface

Michael J. Lyons
ATR Media Information Science Labs
2-2-2 Hikaridai, Seika-cho,
Soraku-gun, Kyoto, Japan 619-0288
mlyons@atr.co.jp

Michael Haehnel
Dept of EE and IT
RWTH Aachen University
Aachen, Germany

Nobuji Tetsutani
ATR M.I.S. Labs
2-2-2 Hikaridai, Seika-cho,
Soraku-gun, Kyoto, Japan

## ABSTRACT
The role of the face and mouth in speech production as well as non-verbal communication suggests the use of facial action to control musical sound. Here we document work on the Mouthesizer, a system which uses a headworn miniature camera and computer vision algorithm to extract shape parameters from the mouth opening and output these as MIDI control changes. We report our experience with various gesture-to-sound mappings and musical applications, and describe a live performance which used the Mouthesizer interface.

## Keywords
Video-based interface; mouth controller; alternative input devices.

## 1. INTRODUCTION
Articles on new interfaces for computer music often begin with a call for greater *embodiment* in the way computers are operated. This claim is sometimes backed up by citing developments in cognitive science which stress the importance of physical and physiological context for understanding the mind [25]. Similar considerations may be applied in the domain of machine-mediated human interaction. Current ways of interacting with computers neglect most of the physiology of human-human interaction and are surely unsuitable for most forms of communication, especially expressive forms such as music.

Working at McGill University half a century ago, Wilder Penfield and his colleagues [19] mapped the sensory-motor cortex by electrical stimulation of conscious patients undergoing neurosurgery. Their pictorial summary of the findings, the somatosensory and motor homunculi, are widely known and their importance for human-machine interaction [2] as well as musical interfaces [8] has been recognized. A striking aspect of the motor homunculus (see Figure 1) is the relatively large area devoted to the organs critical for verbal and non-verbal human communication: the lips, mouth, tongue, larynx, and the face.

The importance of the face and especially the mouth in communication inspired us to develop a musical controller which takes input from facial actions. The face, especially the mouth area, is involved both in sound production, in speech, singing, and in non-verbal emotional communication, in facial expression. It therefore seemed interesting to attempt to design a musical interface making use of our expertise for muscular action of the face.

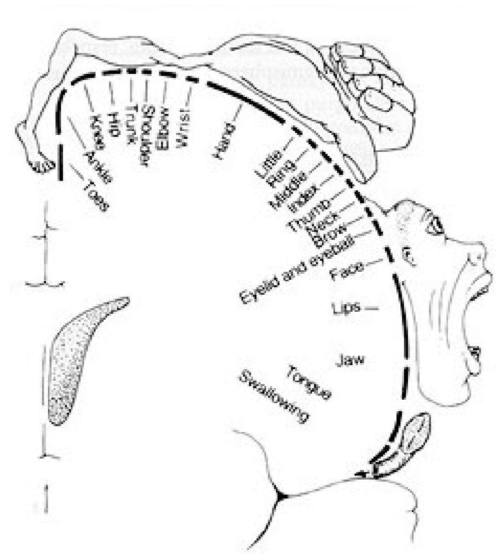



**Figure 1. The motor homunculus or representation of body areas in the motor cortex (adapted from [19]).**

This paper reports work using a video-based approach and focuses on the area of the mouth. Preliminary results of the study were previously published in brief format [13]. The current article is intended as a more complete record of the project in which we: (a) state the context of the work by reviewing related studies (section 2); (b) describe the implementation in detail including more recent developments, discussing design considerations as well as lessons learned (section 3); (c) report our experience with several mappings and musical applications of the controller (section 4); and (e) describe a public performance in which the controller was used (section 5); and (f) conclude with general observations (section 6).

## 2. RELATED WORK

### 2.1 Mouth & Vocal Tract Controllers
The fact that oral cavity shape influences the human voice means there are complex neural circuits relating for muscular control the mapping of shape to sound effect. Use of the oral cavity for modulating sounds other than those produced by the vocal tract is probably as old as instrumental music itself, evidenced by the presence of instruments like the mouthbow in folk cultures around the world.

Functioning by the same principle as acoustic mouth controllers, the TalkBox, which enjoyed popularity in the 1970's, allows a player to directly filter an audio signal with the acoustic transfer of the oral cavity. Holding a small speaker in the mouth, the player modulates the signal by varying the oral cavity shape and position of the tongue. Since the actual acoustic properties of the mouth modify the signal, the TalkBox is intuitive to use. However the range of sound effects is limited by the same acoustic possibilities. It is also requires the player/singer to keep the device in their mouth.

The Vocoder [5] allows effective vocal tract control of synthesized electronic sounds via audio signal processing extraction of voice parameters to modulate synthesized sounds. Using a Vocoder is more akin to speaking or singing than to playing an instrument since it is activated by sounds produced in the vocal tract itself.

By contrast, the interface developed by Orio [16] probes the shape of the oral cavity by stimulation with an external acoustic source. Shape parameters extracted from analysis of the response are output as MIDI controls. Orio found that users could easily learn to control two independent parameters by varying mouth shape, but greater difficulty controlling three parameters.

Vogt *et al.* [26] used ultrasound imaging to measure tongue position and movement in real-time for sound synthesis control. With the Tongue 'n' Groove, an ultrasound device is held below the jaw and an image of the tongue contour reconstructed, or alternatively, optical flow due to tongue motion is calculated. Several mapping metaphors were explored; *e.g.* tongue position was used to play a physical model of the singing voice.

## 2.2 Vision-Based Musical Interfaces

Several previous works have used computer vision techniques for musical interaction.

Multimedia installation artist David Rokeby has experimented for many years with his Very Nervous System [22], or VNS, which is now available for purchase. The VNS web pages do not give an explicit description of what it computes, but VNS appears to be based primarily on pixel calculations responsive to movement, such as temporal differencing in user defined trigger zones.

The BigEye software [23], available commercially from STEIM, allows tracking of coloured objects against a set of definable regions in the video frame, with variables such as relative position, size, and speed output as MIDI parameters.

The EyesWeb software platform [1,6], freely available from the InfoMus group at the University of Genoa, includes several computer vision modules allowing tracking of objects and coloured blobs as well as modules for estimation of affective and expressive qualities of movements, with several output options including MIDI and OSC.

Some vision-based controllers adapt software developed for non-musical purposes. The DanceSpace system [18] added to the MIT Media Lab's Pfinder vision-based person tracker, to allow mapping of movements of tracked limbs, torso, and to changes in musical parameters.

The Augmented Groove system [20] uses the University of Washington HIT Lab's Augmented Reality (AR) Toolkit which supports tracking of high-contrast two dimensional patterns. The AR Toolkit can extract translation, rotation, and tilt of objects labeled with the patterns. The Augmented Groove system maps tracking parameters to MIDI control changes, allowing users to modulate sequencer loops by manipulating physical objects.

**Figure 2. Two miniature CCD cameras used in this work.**

Those with the resources often prefer to develop specific software for a project as this allows greater control over how the solution is implemented.

The Iamascope's vision-to-music subsystem [7] divides the video input frame into detection zones mapped to notes of a chord. A motion detection algorithm allows note triggering with free gestures. The kaleidoscope subsystem mirrors the video input to display intriguing visual feedback of the user's own gestures.

With the Imaginary Piano [24], video input from a camera facing the player is analyzed with a motion detection algorithm which responds to movement of the hands below a vertical threshold to trigger piano notes having pitch determined by the horizontal coordinate.

Ng [15] seems to be the only other work to suggest using a vision-based interface to transform facial gestures to MIDI controls. A video demo of their prototype was shown at NIME-02, but details of their implementation have not yet been published.

# 3. DESIGN EVOLUTION

## 3.1 Face Tracking System

At the outset, our aim was to utilize actions from several areas of the face for musical expression, including movements of the eye regions, eyebrows, mouth, cheeks and movements of the whole head. We started by building upon a vision-based face tracking system developed in our group at ATR. The first prototype was implemented on an SGI O2 computer, using the O2's built-in framegrabber and the IRIS video library. This allowed acquisition of NTSC quarter-frame images (320x240 pixels) at 15 fps. This is the minimum useable frame rate for most musical applications: latencies are noticeable but still tolerable.

We initially considered using a more sophisticated feature shape representation [11,17], but experiments showed that the shadow area of the mouth could be extracted by a very simple colour and intensity thresholding algorithm. First, a region large enough to include the mouth area with certainty is chosen, based on the inter-ocular distance from the face tracking module. Next, pixels in this region satisfying the following equation:

$$I < I_{min} \quad \text{and } R > R_{max}$$

are segmented, where I is pixel intensity and R is its red component and $I_{min}$ and $R_{max}$ are set thresholds. With appropriate values for the thresholds, under a large region of lighting conditions most of the pixels satisfying this condition belong to the shadow area of the mouth. Segmentation by colour thresholding is widely used to track objects in vision systems. For example skin colour is used as a cue in many face detection algorithms, though it is widely known to be affected by the intensity and colour of the illumination. Thresholding of the mouth shadow area seems to be considerably more stable to such illumination changes because we are detecting the *absence of a surface*: the appearance of the cavity is more robust than that of the surrounding skin areas. Use of the automatic gain and colour balance control on the cameras adds to the robustness of the system to lighting changes.

The system has some limitations, for example, dark facial hair near the mouth may also be selected by the thresholding operation. With very dark skin, thresholds may need to be changed or additional lighting used.

The pixels obtained by the thresholding operation were analyzed using a principal components analysis [4] to find the major and minor axis of the segmented area, which approximates an elliptical blob. Use of an algorithm based on colour segmentation and invariant statistics of the pixel coordinates has the advantage that it is robust to translation and in plane rotation of the mouth region. Hence, movements of the camera, unavoidable due to slight vibration of the beam, do not strongly affect the shape parameters extracted from the image. Higher level algorithms based on tracking loops for position estimation, would not share this property.

The two shape parameters were mapped to two MIDI control change values. These were used to control timbre of various physically modelled instruments using the demos Perry Cook's Synthesis ToolKit [3] running on the same SGI O2. Segmented pixels were displayed in red on a video output of the players face. Experiments with this system quickly convinced us that the mouth functions well as a controller of synthesized sound and we next concentrated on developing a mouth controller for use in actual musical performance situations.

## 3.2 Headworn System

When we began this research, the face tracking module was limited to a video processing rate of 13 fps, and tracking was interrupted by large speed or amplitude of head movements. Tracking performance has since been increased to full frame rate, but robustness it is still not adequate for live performance situations. In addition, the apparent facial expression in 2D projection depends on head orientation [12]. Finally, experiments with the tracking system convinced us that a wearable system would allow performers greater mobility and comfort.

These considerations led us to concentrate research on a system based on a head mounted camera pointed directly at the mouth area (see Figure 3). Camera distance and focal length of the lens were chosen so that that the input video frames contained the facial region of the mouth, and excluded other areas that are picked up by thresholding such as the nostrils and, occasionally, a shadow below the lower lip. This eliminates the need for a face tracking system.

**Figure 3. The headset and a view from the camera.**

The headset is a modified Shure SM10A with the microphone and beam assembly removed and replaced with a miniature video camera mounted on a homebuilt aluminum arm. It is important to counterbalance the weight of the camera. Miniature, lightweight video cameras are now widely available (see Figure 2). Most of the work reported here used a Keyence CK-200B miniature colour CCD camera with standard NTSC analog output. We also tried the expensive Elmo QN42H camera pictured in the figure, which gave similar results. The Keyence camera is economical and ideally suitable for use with the Mouthesizer. A web site with more information on obtaining the camera is listed with the references [10].

## 3.3 Machine-Vision Board

To demonstrate the feasibility of an inexpensive, portable, and stable system we next implemented a hardware prototype. We selected the Cognachrome 2000, made by Newton Labs (Renton, WA), a dedicated machine vision board based on the Motorola 68322 processor. The Cognachrome tracks the position, size, aspect ratio, and orientation of several colour blobs at 60 Hz and a spatial resolution of 200x250 pixels, with a proprietary algorithm that uses colour and intensity thresholding and connected region analysis. Video input and output is NTSC format and data communication is via a serial port. Mouth shadow blob dimensions as detected by the board were remapped to two MIDI control changes via a program running on a desktop program. The Cognachrome is programmable allowing onboard implementation of MIDI communications. However, it has several disadvantages, the most important of which were that mouth shadow tracking was more sensitive to illumination changes than with the software algorithm we implemented and that the tracking algorithm could not be easily modified.

## 3.4 Current Implementation

Flexibility considerations led us to return to a software implementation, using Visual C++ and Direct-X running under Windows. The current Mouthesizer operates as a Direct-X filter, allowing use of the system with any input video device for which a driver is available.

Several improvements to the algorithm were made. Connected region analysis is applied after the colour and intensity thresholding operation. This removes thresholded pixels outside of the mouth shadow area. Two types of simple temporal filters were added to remove noise due to rapid fluctuations in lighting or shadows, illustrated in figure 4.

Filter A discards any output that differs from the average of the two most recent output values by more than a set threshold.

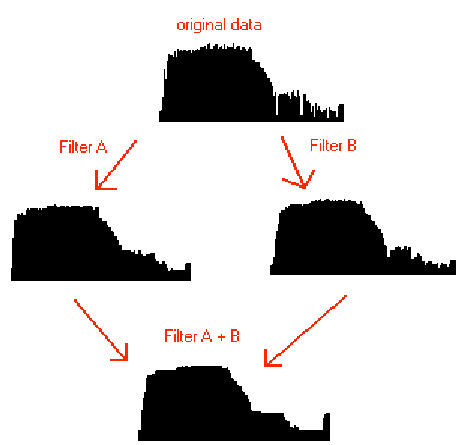



**Figure 4. Filters used with the Mouthesizer to remove noise.**

Filter B temporally smoothes the data by averaging current outputs with previous ones. The filters can be independently turned on or off while the software is running.

A desktop system on a Pentium II with a Winnov Videum capture card ran at 30 fps, while on a Pentium III notebook with I-O Data PCCAP video capture card the system ran at 15 fps, both at a resolution of 320x240 pixels. The system now also works with Firewire and USB cameras at full frame rate. Some recent palmtops should now have sufficient processing power to run the Mouthesizer algorithm at low resolution, which would allow it to be used as a fully wearable device.

# 4. MAPPING & APPLICATIONS

Below we first report gesture-to-sound mappings which were found to work well with the Mouthesizer. Then we describe actual applications of the Mouthesizer to playing music. Most of our experiments with mappings and applications were made with the Nord Modular Virtual Analog Synthesizer (from Clavia, Sweden). With the Nord Modular, synthesis or audio effects patches are edited in software with an intuitive graphical interface, but run on dedicated DSP hardware. Patch variables were easily adjusted using control panel knobs and driven by external MIDI controllers.

## 4.1 Mapping

In all cases the mouth shape parameters were mapped to two MIDI control changes. The mappings discussed below were not intended to be one-to-one mappings of shape to sound. Rather, two principles guided our experimentation: the role of the mouth as a filter in sound production and the action of the facial muscles in the facial expression of emotion. Our goal was to try to create intuitive and compelling mappings from action to sound by making use of existing motor expertise and brain maps for sound production and emotional expression.

Musical interface mapping is a subtle issue [9] and there is room for further exploration of the expressive potential of the Mouthesizer. For example, for some vocal consonants the lips and tongue act as sources of sound. Expression of certain emotions such as surprise or mirth can have relatively rapid onset dynamics, which may not be well modelled as continuously changing controls. Cursory experiments suggested that it should also be interesting to use the Mouthesizer interface to trigger sound events such as samples, but we have so far not pursued this line as a mapping strategy. To encourage further experimentation with mappings, we are planning to make a version of the Mouthesizer available in the near future.

### *4.1.1 Mouth Height*

One of the most is compelling and intuitive mappings we found uses the height of the mouth opening to control the cut-off frequency of a sweeping resonant low-pass filter. With this mapping opening the mouth opens up the filter, letting higher frequency components of the sound pass. This audio effect is popularly known as *wah-wah,* an onomatopoeic term describing the effect of opening and closing the mouth while voicing the sound “ah”. This mapping mimics effects available with mouthbow and jaw harp instruments, as well as the TalkBox. Simple, intuitive effects also result from mapping mouth height to volume control, sustain, or damping.

### *4.1.2 Mouth Width*

We found interesting expressive effects by mapping mouth width to *distortion* level of an amplifier. This was motivated by the action of the mouth in expressions of pain, suffering, or fear. Opening the mouth increases the non-linearity of the response of an audio-amplifier which clips the guitar signal waveform. Stretching the corners of the mouth apart in a grimace increases the level of distortion.

We also tried mapping mouth width to the resonance of a resonant low-pass filter. Stretching the mouth wide gives a high-frequency chirp, expressing arousal without the negative emotional valence of the distortion effect.

### *4.1.3 Mouth Aspect Ratio*

Here the mouth aspect ratio or eccentricity was used to control an audio morph between formant filters for three of the fundamental vowel sounds [a], [i], [o]. [i] has the greatest eccentricity, [o] is the most rounded, having the least eccentricity, [a] has intermediate eccentricity. An existing formant filter module of the Nord was used to control the morph. This gives an intuitively natural mapping of mouth shape to filter audio effect which is similar to acoustic effects playable with controllers like the TalkBox.

## 4.2 Applications

The Mouthesizer was played informally, with three main musical applications, guitar effects, keyboard, and sequenced loops.

### *4.2.1 Guitar Effects*

Ichiro Umata, jazz guitarist and cognitive science researcher, used the Mouthesizer to control guitar effects. Mouth height controlled wah-wah as described above and mouth width adjusted the amount of distortion. This experiment used an early version of the Mouthesizer running at 15 fps on an SGI O2 computer. The guitarist had little prior practice using the Mouthesizer. After a session lasting approximately one hour, he noted that the

**Figure 5. Controlling guitar effects with the Mouthesizer.**

Mouthesizer was easily learned and more natural to play than a pedal controller. He also observed that changes of mouth width and height are correlated for most movements, making the controller more interesting to use than if the two audio effects were independently adjustable. This agrees with the findings of Hunt *et al.* in their study of simple and complex mappings.

### *4.2.2 Keyboard Synthesizer Demo*

We experimented with several keyboard synthesizer patches running on the Nord Modular, controlling patch parameters with the Mouthesizer. We tried these at a live demo during the ATR Open House exhibition. The mappings which easiest for most visitors to understand were simple ones usually associated with keyboard pedals such as volume control, sustain, and damping.

Reactions to the Mouthesizer varied greatly. Some visitors, having conservative musical tastes, found the concept strange or at least humourous. Others, sympathetic to electronic music, found it more appealing.

#### *4.2.3 Sequenced Loops*

Inspired by the Augmented Groove system, we used the Mouthesizer to control techno loops. This allows one to add expression to an automatically played musical sequence. Again, sweeping filters, resonance, distortion, and formant filter morphs work well here.

## 5. LIVE PERFORMANCE

Jordan Wynnychuk gave a 30 minute solo live performance of improvised electroacoustic music using the Mouthesizer at the Kyoto Kyoryukan to audience of about 30 people. Figure 6, a picture of a rehearsal for the performance shows the instrumentation used in the concert. Jordan used a touch-sensitive MIDI control pad and STEIM's LiSa (Live Samples) software, to trigger and manipulate samples with the fingers of both hands. The sounds used in this performance consisted of glitches, squeaks, blips, bangs, buzzes, whirs, other "error" sounds, as well as samples of percussion instruments.

Audio effects running on the Nord Modular were controlled using the Mouthesizer. Aesthetic considerations led us to experiment with audio effect mappings less intuitive than the ones described above, including a mix of extreme distortion, high and low pass filtering, and panning between left and right channels. One of the most interesting effects we discovered panned high and low frequency filtered versions of the audio signal between the two channels by opening and closing the mouth.

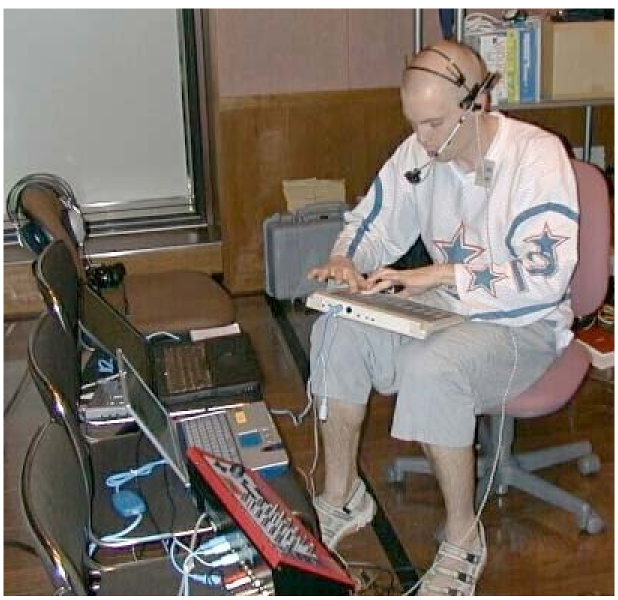


**Figure 6. Jordan Wynnychuk with controllers used for the live performance.**

Figure 7 shows a sequence of pictures from Jordan's performance in which a mouth gesture is being used to adjust sound expressively. Not visible in these images are the highlighted mouth shadow regions, which were projected on a screen beside the stage.

In addition to the interest and originality of Jordan's performance, audience members' attention seemed to be captured by the novelty of the Mouthesizer interface and concept. Many asked afterwards about how it worked or suggested ways it could be used to play music. One or two reported on their check for causality between mouth action and aural effect: they found it sometimes easily visible but quite obscure at other times. This appeared to be mainly a function of the mapping.

## 6. CONCLUSION

Experimentation with a variety of mappings and musical applications, and use of the Mouthesizer in a live public performance confirmed our hypothesis that facial gestures, especially movements of the mouth, are suitable for expressive musical control. Several lessons were learned in the design process.

Abandoning the head-tracking system early in the project later led us to avoid discrete classification of facial expressions for control of musical effects. This was fortunate: a controller which categorized emotion discretely would have limited responsiveness to movement quality and would reduce expressiveness. Rather, we captured a signal responsive to the motion of the mouth with a simple vision algorithm and relied on carefully chosen mappings to enable expressive control of sound production.

A valuable feature of the Mouthesizer is the visual feedback provided by highlighting segmented pixels. This gives a highly visible display mirroring the player's actions. Rizzolatti and colleagues [21] have discovered "mirror neurons" in monkey frontal lobes which respond to specific motor actions both when they are performed and when the same or a similar action is observed. Such circuits are thought to be important for learning motor behaviours. Controllers which mirror a motor action via visual display (and perhaps also auditory display?) should strongly stimulate these circuits. This may lend appeal to musical interfaces that mirror the player's gestures. A previous vision-based interface which shares this property is the Iamascope [7].

Additionally, visual perception of lip movement is known to affect auditory perception of speech [14] (the "McGurk effect"). The Mouthesizer brings such cross-modal sensory mappings into play for both performer and observer.

Such considerations lead us to conclude that video-based and other new gestural controllers offer much more than a visually engaging spectacle for the audience. They enable news ways to use the body and its sensory-motor systems to explore human expression and communication via sound.

Recently the Mouthesizer was integrated with an improved face tracking system to create an interface which allows users to point and click with facial gestures. Hence, work on a musical interface led to a non-musical side product. Musical applications seem to stimulate exploration of a very wide range of interaction paradigms. In this way, the NIME conference may have an important contribution to make to the wider field of human-computer interaction.

**Figure 7. Jordan Wynnychuk using the Mouthesizer in a live performance.**

## 7. ACKNOWLEDGMENTS

MJL thanks Christoph Bartneck, Rodney Berry, Palle Dahlstadt, Sidney Fels, Steven Jones, Ivan Poupyrev, Ichiro Umata, Jordan Wynnychuk, and Tomoko Yonezawa for stimulating and helpful interactions. This work was supported in part by the Telecommunications Advancement Organization (TAO) of Japan.

## 8. REFERENCES

[1] Camurri, A. and Trocca, R., Movement and Gesture in Intelligent Interactive Music Systems, in Wanderley, M., and Baffier, M. (Eds.), Trends in Gestural Control of Music CD-ROM, IRCAM, Paris, 2000, pp. 450-474.

[2] Card, S.K., Mackinlay, and Robertson, G.G., A Morphological Analysis of the Design Space of Input Devices, ACM Transactions on Information Systems 9(2): 99-122, 1991.

[3] Cook, P. and Scavone, G. P., The Synthesis ToolKit (STK), in Proc. of the 1999 International Computer Music Conference, Beijing, International Computer Music Association.

[4] Duda, R.O. and Hart, P.E., Pattern Classification and Scene Analysis. John Wiley & Sons, 1973.

[5] Dudley, H., Remaking Speech. Journal of the Acoustical Society of America, 11: 169-177, 1939.

[6] EyesWeb Project, http://www.eyesweb.org.

[7] Fels, S.S. and Mase, K., Iamascope: A graphical musical instrument. Computers and Graphics, 2(23):277-286, 1999.

[8] Gillespie, B., Haptics in Music, in Cognition, and Computerized Sound, P. Cook (Ed.). MIT Press, Cambridge, MA, 1999.

[9] Hunt, A., Wanderley, M.M., and Paradis, M., The Importance of Parameter Mapping in Electronic Instrument Design, in Proc of the 2002 Conference on New Interfaces for Musical Expression (NIME-02), 149-154.

[10] Keyence Japan CK-200B miniature CCD camera: http://www.keyence.co.jp/hobby/ccd/index.html.

[11] Lyons, M.J., Budynek, J., and Akamatsu, S., Automatic Classification of Single Facial Images, IEEE PAMI 21(12): 1357-1362, 1999.

[12] Lyons, M.J., Campbell, R. *et al.,* The Noh Mask Effect: Vertical Viewpoint Dependence of Facial Expression Perception, Proc. Roy. Soc. Lond. B 267: 2239-2245, 2000.

[13] Lyons, M. J., and Tetsutani, N., Facing the Music: A Facial Action Controlled Musical Interface. in CHI 2001 Extended Abstracts, ACM Press, 309-310.

[14] McGurk, H. and MacDonald, J., Hearing Lips and Seeing Voices, Nature, 264:746-748, 1976.

[15] Ng, K., Interactive Gesture Music Performance Interfaces, in Proc of the 2002 Conference on New Interfaces for Musical Expression (NIME-02), 183-184.

[16] Orio, N., A Gesture Interface Controlled by the Oral Cavity, in Proc. Of the 1997 International Computer Music Conference, San Francisco, International Computer Music Association, 141- 144.

[17] Pantic, M. and Rothkrantz, L.J.M., Automatic Analysis of Facial Expressions: The State of the Art, IEEE PAMI 22: 1424-1445, 2000.

[18] Paradiso, J. and Sparacino, F., Optical Tracking for Music and Dance Performance, in "Optical 3-D Measurement Techniques IV", A. Gruen, H. Kahmen eds., Herbert Wichmann Verlag, Heidelberg Germany, pp. 11-18 (1997).

[19] Penfield, W. & Rasmussen, T., The cerebral cortex of man: a clinical study of localization of function, Macmillan, New York, 1950.

[20] Poupyrev, I., Augmented Groove: Collaborative Jamming in Augmented Reality, in SIGGRAPH 2000 Conference Abstracts and Applications, ACM Press, p. 77.

[21] Rizzolatti, G, Gadiga, L., Gallese, V, and Fogassi, L., Premotor Cortex and the Recognition of Motor Actions, Cognitive Brain Research, 3: 131-141, 1996.

[22] David Rokeby, Very Nervous System: http://www.interlog.com/~drokeby/vns.html

[23] STEIM, BigEye: http://www.steim.nl/bigeye.html

[24] Tarabella, L., Gestural and Visual Approaches to Performance, in Wanderley, M., & Baffier, M. (Eds.), Trends in Gestural Control of Music CD-ROM, IRCAM, Paris, 2000, pp. 604-615.

[25] Varela, F.J., Thompson, E., and Rosch, E., The Embodied Mind, MIT Press, Cambridge MA, 1991.

[26] Vogt, F., McCaig, G., Ali, M.A., Fels, S., Tongue 'n' Groove: An Ultrasound based Music Controller, in Proc of the 2002 Conference on New Interfaces for Musical Expression (NIME-02), 60-6.

# E-mic: Extended Mic-stand Interface Controller

Donna Hewitt
University of Western Sydney
Locked Bag 1797, Penrith South DC
NSW Australia 1797
+61 2 9300 8485
dhewitt@bigpond.net.au

Ian Stevenson
Audile Design
2/4 Peckham Ave, Chatswood
NSW Australia 2067
+61 (0)407135475
audile@bigpond.com

## ABSTRACT
This paper describes work in progress for the development of a gestural controller interface for contemporary vocal performance and electronic processing. The paper includes a preliminary investigation of the gestures and movements of vocalists who use microphones and microphone stands. This repertoire of gestures forms the foundation of a well-practiced 'language' and social code for communication between performers and audiences and serves as a basis for alternate controller design principles. A prototype design, based on a modified microphone stand, is presented along with a discussion of possible controller mapping strategies and identification of directions for future research.

## Keywords
Alternate controller, gesture, microphone technique, vocal performance, performance interface, electronic music.

## 1. INTRODUCTION
Contemporary vocal performance is rarely without amplification and the employment of a microphone. The microphone has become a performance tool of the contemporary vocalist and a means for extending the voice as an instrument.

Given that the microphone is such a prevalent tool and that it is 'played' by the performer, it is possible to extend this idea to include its use as an interface for real time electronic music performance by capturing gestures via the microphone and stand, in order to derive control signals which are sent to a sound engine located in software on a computer. In the process of conceiving an alternate controller system for vocalists, it would seem attractive to address the principal limitation of the microphone/PA system as an instrument. In amplified performance situations, the vocalist has little direct control over the sound of their voice through the sound reinforcement system. Once the sound enters the microphone, any additional signal processing, such as filtering, reverberation, distortion, granulation, delay effects etc., added to the vocal signal, are usually carried out by a sound mixer or third party. Often these effects are of an intrinsically musical nature and are closely allied with other vocal production techniques employed by the performer. Wanderley and Depalle [34] showed how acoustical effects can be achieved through a performer's movements, however, the ability of the performer to shape or interact with the sound can be somewhat inhibited by the fact that some of the more critical aspects of the resulting sound are mediated by the third party.

In any proposal for an alternate controller, one must address the perceived limitations and difficulties of existing electronic music performance practice. This is particularly relevant to the recent trend towards using the laptop computer as a musical instrument without some kind of visible (to the audience) performance/ gestural interface [5], raising issues concerning the performer's relationship to the audience. The most commonly cited 'deficiency' in laptop performance is that, with the performer seated behind the laptop, there is an inherent lack of gestural communication between performer and audience due to the fact that gesture is so small and often hidden from view. As a result, the performance can have a detached, non-communicative quality.

*Musical Performance is also a social act, and, whether real or virtual (in the recording studio), an audience is critical in shaping the performance event.* [10]

Another problem that arises for vocalists using a laptop in performance is that the performer may be physically inhibited by the posture of sitting at the computer when trying to vocalize, see Figures 1 and 2. These issues suggest that there is a need to extend the vocalist's control over sound and to address some of the limitations of recent laptop performance practice by developing an alternate controller which not only captures gesture, but provides a visually engaging performance interface for the vocalist who wishes to work with electronic resources. The photographs in Figure 1 and 2 show both a laptop and desktop computer performance respectively, where the performer is using live vocal input. The performer is seated in front of a laptop and the microphone is placed between the performer and the computer screen.

**Figure 1. Donna Hewitt Impermanent Audio 2002 [18]**

**Figure 2. Donna Hewitt Waveform 2001 [36]**

In determining the design aspects of a proposed alternate controller, there is a need to;

a) study the gestural qualities of vocalists to identify common aspects to the 'language'

b) identify the most effective means of capturing these gestures making use of available sensing technologies and hardware

c) come to an understanding of the most effective means of mapping gesture onto sound in order to produce a flexible and playable instrument.

## 2. GESTURE

### 2.1 General Principles
The broad principles of gestural control have been discussed in the existing literature [14],[28], [29], [33]. This paper aims to focus on matters relating specifically to vocal performance and gestures relating to the development of a vocal interface. The classification of gestures used by vocal performers serves as a starting point for a more comprehensive,

rigorous analysis and categorization of gesture which will inform decisions made in relation to the more subtle aspects of controller design A preliminary categorization has been made by observing vocal performers' behaviour and drawing from personal performance experience. The most logical site for observing the movements and gestures of vocalists is popular music, where the overwhelming majority of microphone gesture practice is located. Although the gestural principles will be derived from a study of popular music, it is envisaged that it will be possible to use the Emic in a wide range of musical styles. The Emic may, in time, allow a whole new set of gestural practices to be developed, as has been seen with developments of other electronically extended musical instruments. The sensor bow described in [2] as an example, required string players to "modify their traditional technique" and showed how certain techniques were effective for the new sensor bow but not useful for playing the traditional string instrument.

Vocal performers employ a wide range of 'gestures' during performance. The term 'gesture' refers to the bodily movements allowing the performer to 'interact with their environment, to modify it and to communicate' [4].

For a vocalist, the body itself is the instrument. Playing the instrument requires control of various body parts involved with the breathing apparatus, vocal articulators and resonating cavities. Vocal performers, as with most traditional instrumentalists, also move in other ways, which may not be directly involved in sound production. "These gestures have been labeled as expressive, accompanist, ancillary or non-obvious" [34]. Studies show a wide range of expressive information is present in, and can be drawn from, the bodily gestures of a performer [7],[8],[9], [10]. Gestures provide interpretive cues for audiences and are the "by-product of psycho-physical, social and cultural practices surrounding performance". [10]

Our observations focused on how the performer approached and/or touched the microphone and microphone stand, i.e. what human movements and physical interactions with the microphone and microphone-stand commonly occurred during performance. A comprehensive understanding of the function of these 'microphone gestures' is a vast area for further study, one that will provide valuable insight into creating an effective mapping strategy.

## 2.2 Microphone Gesture

While every performer possesses a certain number of idiosyncratic movements and gestures, there does appear to be a number of common interactions and gestures associated with microphone and microphone stand use, which are broadly outlined below.

### 2.2.1 Physical Interactions

These gestures include physical interactions where the micro-phone stand is physically touched in some way. The main categories of physical interactions are the following.

#### *2.2.1.1* Grasping Gestures

Grasping the microphone

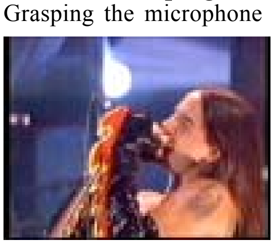


**Figure 3. Red Hot Chilli Peppers, Anthony Keidis [26]**

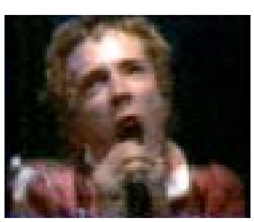


**Figure 4. Sex Pistols, Johnny Rotten [1]**

Grasping the stand

**Figure 5. The Doors Jim Morrison [11]**

Stroking the stand (sliding hands up and down the stand)

**Figure 6. Mariah Carey [23]**

#### 2.2.1.2 Stand Moving Gestures

Tilting the stand

**Figure 7. Red Hot Chilli Peppers [26]**

**Figure 8. Midnight Oil Peter Garret [31]**

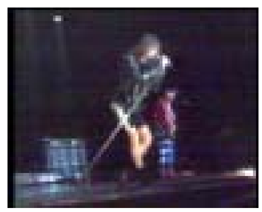


**Figure 9. INXS Michael Hutchence [19]**

**Figure 10. The Doors Jim Morrison [11]**

Moving and swinging the stand

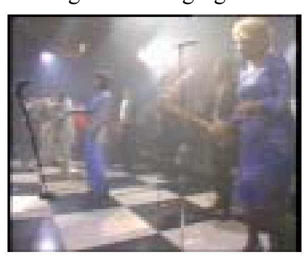


**Figure 11. James Brown [30]**

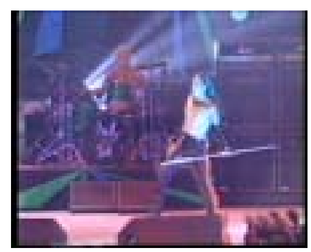


**Figure 12. Red Hot Chilli Peppers [26]**

(James Brown (left) throws the stand and reins it back in with the microphone lead)

Straddling the stand between the legs

**Figure 13. Jim Morrison The Doors [11]**

##### *2.2.1.3 Tapping*

Foot tapping the base

Hand tapping the microphone and stand

##### *2.2.1.4 Other*

Altering the stand height

Moving the microphone in and out of its clip/holder

**Figure 14. Red Hot Chilli Peppers [26]**

### 2.2.2 Free Arm/Non Contact Gestures

Vocalists make a lot of free arm gestures, where they do not touch the stand but move their arms, hands and bodies around it. In these instances, the microphone and stand provide a focal point around which the performer works or interacts, acting as a point of spatial reference for the non-contact gestures. These gestures include open hand gestures (palms facing toward the stand) and caressing type gestures (where the hands do not make contact with the stand but move around it).

**Figure 15. Stevie Nicks [38]**

**Figure 16. Mariah Carey [23]**

## 2.3 Functional and Contextual Aspects

Preliminary observations of vocal performers show a number of relationships between intent and physical gesture, for example, the increased grip strength of the microphone and stand most often correlate with an increase in tension in the musical intent. Performers tend to grasp the stand in a more aggressive manner when conveying more violent or angry passages, delicate stroking seems to occur more often during gentler passages. These are immediately observable one to one relationships, however gesture in performance is a complex system which is mediated by contextual factors. Gestures do not always have the same meaning or function and the mapping of each gesture needs to be considered in its unique musical context.

The congruity between gesture and intent are particularly important for a vocal performer due to the close ties between the body, psychological state and the sound produced, since "the body cues the mental representations of the music"[8]. Singers often employ learned or mimetic gestures in performance and many vocalists carefully choreograph their movements to achieve various effects such as to cue other musicians (i.e. conducting), or perhaps to create a deliberate expressive effect or to elicit a response from the audience. Singing teachers often teach singers what to do with their hands in order to furnish a performance with expressive intention. Gellrich [15] has suggested that these learned gestures can have both a positive or negative effect on a performance. It has been shown that [15] learned, mimetic or choreographed gestures can be problematic for communication with an audience when the gestures are incongruous with the intent. It may, therefore, be desirable for a performer's gestures to appear as natural and organic as possible, however the Emic does not dictate such relationships.

While vocal performance lends itself to study, valuable insights may be gained by looking at gesture in the context of verbal communication in general. Davidson [10] has shown that singer's gestures correspond with conversation related gestures. In gesture associated with speech, it seems that the listener relies more heavily on gesture for interpretation when the speech is ambiguous [32] or as background noise increases [27]. In relating these observations to a musical context and by noting the importance of gesture in situations with higher background noise, it is possible that in musical performance, the presence of other musical elements around the vocalist increases the importance of gesture where the intelligibility and the meaning and intent of the lyric text are of prime importance. Evidence for this can be observed in popular music performance practice, where an increase in gestural activity can be observed in situations where the vocalist is wishing to articulate the lyric clearly to the audience. Gesture is thus an important functional and expressive device in making the text intelligible for the audience and as an important means for expressive communication between the performer and audience.

# 3. INSTRUMENT DESIGN - Emic

Interfacing the voice and electronic processing requires a) sound capture b) a gesturing device for control input, i.e. something that the performer touches or moves to create the required control information and c) a signal processing engine which takes the captured audio and processes it in real time. The instrument is thus a gesturally responsive device bearing a relationship to an acoustic instrument in that the performer touches or moves something in order to produce and transform sound. For a vocalist, the logical signal source for an electroacoustic instrument is a microphone output. The microphone is generally placed on a microphone stand, making the microphone stand itself a logical choice as a gesturing device. The proposed design, therefore, is to build an instrument resembling a microphone stand with the parts made active as gestural controllers. The microphone stand will serve the function of a controller device, resembling a large multi-axis joystick with various buttons, sliders and sensors. The aim being to capture the common gestures that have been identified in section 2.1.

Emic is a logical adaptation of a device that many vocal performers are comfortable handling. The design aims to minimize 'physical retraining' by "retaining a physical interface that is functionally very similar to the practiced instrument"[14]. By choosing common gestures, the intent is to make the system more intuitive and accessible to a larger number of performers. The mic-stand is an extremely popular, widely used device that has historically endured due to its ergonomic suitability. The other advantage of the mic-stand is its familiarity to audiences. This allows the relative social codes and cultural connections associated with the interface and performance to be maintained. By re-integrating the body into the social context of music performance the device is attempting to address the perceived "lack of somatic/corporeal presence" in the performance of electro-acoustic music, an issue raised in [2].

# 4. MAPPING

Two fundamental approaches to the question of control mapping are those which see mapping as an integral part of the experimental process of composition and those, on the other hand, which identify the requirement for fixed and repeatable mapping of gestural input to system control outcome. [17]

A number of points relating to mapping strategies have been highlighted in key papers and studies [17], [28], [33], [37], [39]. It seems that one of the more important aspects of mapping is to maintain the congruity between the character of the music, the expressive intent and the corresponding physical gesture. This is important from an audience perspective, since in many contexts, the audience relies on physical gesture for much of the information concerning expression and musical intent [7],[8], [9]. An important mapping consideration is to strive for a compatibility and a logical relationship between the physical gesture and the sonic outcome of that gesture and to avoid cognitive dissonances. For example, live vocalists often use the act of tilting the microphone stand when producing more intense sounds, so it would seem logical that this parameter was mapped to a parameter which intensified texture or sound intensity in some way. Stroking the microphone (ribbon sensors) would be mapped to a more intimate, subtle sound transformation. As stated in Wessell and Wright

*... there should be a correspondence between the size of a control gesture and the acoustic result. Although any gesture can be mapped to any sound, instruments are most satisfying both to the performer and the audience when subtle control gestures result in subtle changes to the computers sound and larger, more forceful gestures result in more dramatic changes to the computer's sound.* [37]

The two primary goals of the mapping process are firstly to have a satisfying communicative relationship from an audience perspective and secondly to create a workable relationship from a performers' perspective which meets the requirements for satisfactory control of the sound source and allows high level performance skills to be developed.

The software packages most likely to be employed in the mapping stage are Miller Puckettes' PD [25] and Ross Bencina's Audiomulch [3]. The prototype design inherently provides support for fixed approaches to control mapping while at the same time allowing new mapping strategies to be developed to support a new and emerging electro-acoustic performance practice associated with the mic-stand controller.

## 5. APPLICATIONS

The mic-stand interface device will find applications in a range of contemporary performance situations. Popular commercial music increasingly employs specialized vocal processing systems. This interface allows these systems to come under direct control of the performer providing scope for new avenues of musical expression.

In the field of experimental electro-acoustic music and performance art, advanced control systems have a long history. This device fits neatly into this field, where the performer is often inhibited by clumsy general purpose computing interfaces.

It may also be possible to utilize the interface in conjunction with other existing gesture capture devices such as the Yamaha MIBURI system (body suit) [40] or the Mouth Synthesizer [22] (captures facial gestures). This would enable additional gestural information to be collected.

## 6. PROTOTYPE FEATURES

### 6.1 Overview

The mic-stand interface device must provide a range of simple mechanisms to capture the characteristic gestures listed above. The control systems must be simple and intuitive but must not restrict the virtuosic performer. There is no fixed relation between control signal and sound processing.

**Figure 17. Emic Prototype**

### 6.2 Transducers

The transducers employed in the prototype are detailed below:

#### 6.2.1 Mic Holder Joystick

The standard microphone holder allows the microphone to pivot front to back enabling the capture of microphone tilt movements. This arrangement is augmented with a dual axis pivot arrangement with a simple linear relationship between microphone angle and control signal across two orthogonal axes. The microphone holder joystick is fitted with a return spring stiff enough to support the microphone even when the stand is tilted.

**Figure 18. Mic Holder Joystick**

#### 6.2.2 Slide Sensors

Microphone stand grasping and stroking gestures are captured with two 300mm linear resistive pressure/position sensors fitted either side of the stand. These sensors may be used as continuous controllers or as multi-position discrete switch inputs to be decoded in software.

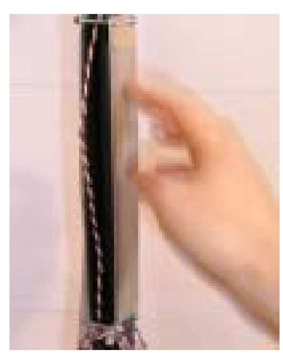


**Figure 19. Right Slide Sensor**

#### 6.2.3 Distance Sensors

Free arm gestures can be captured with a distance sensor that can be played in a *Theremin* type manner. Two optical sensors with a range of 400mm are fitted just below the microphone holder on either side of the stand.

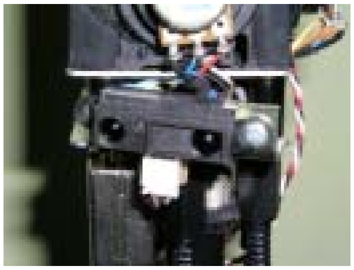


**Figure 20. Distance Sensors**

#### 6.2.4 Tilt Sensors

Microphone stand tilting, swinging and moving gestures can be captured with a dual axis tilt sensor. This custom made device captures the fixed gravitational acceleration across two orthogonal axes, providing tilt sensing in X and Y planes. In addition to capturing the tilt of the stand this device will also capture rapid acceleration due to impacts on the stand from hitting, kicking or dropping.

**Figure 21. Tilt Sensor**

#### 6.2.5 Mic Holder Pressure Sensors

Microphone grasping gestures can be captured with two small pressure sensors attached to the microphone holder.

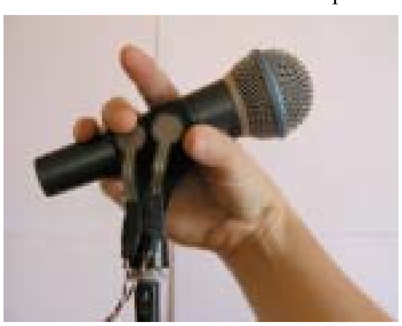


**Figure 22. Mic Pressure Sensors**

#### 6.2.6 Foot Pressure Sensors

Foot pressure on the base of the stand is captured using a simple pressure sensor.

**Figure 23. Foot Pressure Sensor**

### 6.3 Control Systems

Many commercially available and experimental real-time signal-processing devices are fitted with the Musical Instrument Digital Interface (MIDI). In addition to this there are a range of commercially available analogue to MIDI interfaces. The availability and wide use of the MIDI interface is its main benefit. The most significant disadvantage of MIDI is its limited resolution. MIDI may easily be substituted with a floating-point control system such as Open Sound Control or other system specific messaging system such as Max/PD [25]. Part of the composition process will be concerned with finding mappings from available physical controls to signal processing parameters. The system must be flexible in respect of providing unlimited mapping arrangements.

### 6.4 Interfacing

The prototype system employs a simple interfacing strategy based on ready availability of components, simplicity, tourability and reliability in performance environments. The components can be easily removed to protect them while travelling.

#### 6.4.1 Multi-core Cable

This simple interfacing method provides reliability and low cost construction. FM radio data transmission devices may replace this method.

#### 6.4.2 CV to MIDI Converter

A low cost control voltage to MIDI converter made by Angelo Fraietta [13] is used. This device provides sixteen inputs for analog to MIDI conversion.

### 6.5 Control Feedback

The tactile controls employed in the system provide inherent positional feedback. The choice of non-mechanical sensing technology in the tilt sensor provides the user with the familiar ballistic response associated with conventional microphone stands. The non-tactile distance sensor provides no positional feedback. These sensors require either advanced technique or reduction of sensitivity or resolution in the control mapping stage. Slide position sensors are fitted with tactile detents for positional orientation or multi-position switch use.

## 7. CONCLUSION

Having identified the key categories of gesture and the means by which those gestures may be captured, an Emic prototype has been developed. The next stage in the process is to develop workable mapping strategies and to implement the compositional process.

## 8. REFERENCES


[1] Anarchy in the U.K.. The Filth and The Fury – A Sex Pistols Film. Video. Directed by Julien Temple. USA, Warner Home Video (2000).

[2] Bahn, C., Hahn,T. & Trueman, D. Physicality and Feedback: A Focus on the Body in the Performance of Electronic Music. Multimedia Technology and Applications Conference. University of California, Irvine. (2001)

[3] Bencina, R. Audiomulch: Interactive Music Studio (software) http://www.audiomulch.com

[4] Cadoz, C., Luciani, A., & Florens, J. L. Responsive input devices and sound synthesis by simulation of instrumental mechanisms: The CORDIS system. Computer Music Journal, 8(3) (1984), 60-73.

[5] Cascone, K., Laptop Music: Counterfeiting Aura in the Age of Reproduction. Parachute No. 107 (2002), 56.

[6] Cook, P., Principles for Designing Computer Music Controllers. Conference Proceedings NIME (2001) http://www.csl.sony.co.jp/person/poup/research/chi2000wshp/papers/cook.pdf

[7] Davidson, J.W. Visual perception of performance manner in the movements of solo musicians. Psychology of Music, 21 (1993), 103-13.

[8] Davidson, J.W. Which area of a pianist's body convey information about expressive intention to an audience? Journal of Human Movement Studies, 26 (1994), 279-301.

[9] Davidson, J.W. What does the visual information contained in music performances offer the observer? Some preliminary thoughts. R. Steinberg (ed.), The Music Machine: Psychophysiology and Psychopathology of the Sense of Music Heidelberg, Germany: Springer (1995), 103-15.

[10] Davidson, J. W. The role of the body in the production and perception of solo vocal performance: A case study of Annie Lennox. Musicae Scientiae, Vol V, No. 2, (Fall 2001), 235-256.

[11] The Doors, Live at the Hollywood Bowl. Video. Directed by Ray Manzarek USA. Universal Edition (1987).

[12] Ekman, P. & Friesen, W. The repertoire of nonverbal behavioral categories – origins, usage, and coding. Semiotica, 1 (1969), 49-98.

[13] Fraietta, A. CV-to Midi–Midi to CV Midi Controller http://www.users.bigpond.com/angelo_f/midicontroller/MidiController.html

[14] Garnett, G., & Goudeseune, C. Performance Factors in Control of High-Dimensional Spaces. Proceedings of the 1999 International Computer Music Conference. San Francisco. (1999), 268–271 http://zx81.isl.uiuc.edu/camilleg/icmc99.html

[15] Gellrich, M. Concentration and Tension. British Journal of Music Education, 8 (1991), 167-79.

[16] Hunt, A., & Kirk, R. Mapping Strategies for Musical Performance. M. Wanderley and M. Battier (eds). Trends in Gestural Control of Music. Ircam, 2000.

[17] Hunt, A., Wanderley, M. & Kirk, R. Towards a Model for Instrumental mapping in Expert Musical Interaction. In the Proceedings of the International Computer Music Conference. San Fransisco. International Computer Music Association (2000), 209-212.

[18] Impermanent Audio, Live Performance. Frequency Lab. Sydney, Australia. (August 2002). Photo : Mr Snow. http://laudible.net/impaud/archive/

[19] INXS : In Search of Excellence (1989) Video Directed by Hamish Cameron. Polygram Hamrod Productions ISBN 080 850 *

[20] Jorda, S. FMOL: Toward User-Friendly, Sophisticated New Musical Instruments. Computer Music Journal 25:3 (Fall 2002), 23-29.

[21] Lazzetta, F. Meaning in Musical Gesture. In Trends in Gestural Control of Music. M. Wanderley and M. Battier (eds) Ircam - Centre Pompidou, 2000.

[22] Lyons, M. & Tetstani, N. Mouthesisor A Facial Gesture Musical Interface http://www.mis.atr.co.jp/~mlyons/pub_pdf/79.pdf

[23] Mariah Carey performing Hero, Live at Proctors Theater New York City avail on NBC Thanksgiving Special Home Video Video. Directed by Larry Jordan USA 1994 Warner Brothers.

[24] Mulder, A., Fels, S. & Mase, K. Mapping Virtual Object Manipulation to Sound Variation. T. Rai and R. Basset (eds). IPSJ SIG notes 97(122), (1997). 63-68.

[25] Puckette, M. Pure Data (software) http://www.crca.ucsd.edu/~msp/software.html

[26] Red Hot Chilli Peppers 'Warped' (live performance) MTV Video Music Awards (1995) http://www.mtv.com/bands/az/red_hot_chili_peppers/audvid.jhtml

[27] Rodgers, W.T. The contribution of kinesic illustrators toward the comprehension of verbal behavior within utterances. Human Communication Research, 5 (1978), 54-62

[28] Rovan, J., Wanderley, M., Dubnov, S., & Depalle, P. Instrumental Gestural Mapping Strategies as Expressivity Determinants in Computer Music Performance. Kansei, The Technology of Emotion. Proceedings of the AIMI International Workshop, A. Camurri, (ed.) Genoa: Associazione di Informatica Musicale Italiana, October 3-4, (1997), 68-73. http://www.ircam.fr/equipes/analyse-synthese/wanderle/Gestes/Externe/Mapp/kansei_final.html

[29] Sapir, S. Gestural Control of Digital Audio Environments. Journal of New Music Research, Vol. 31, No. 2 (2002), 119-129.

[30] A Soul Session, James Brown and Friends. Video. Directed by David Grossman, USA, Cinemax, 1987.

[31] So You Want To Be a Rock Star. Music Documentary Video. Directed by Ray Argall, Sydney Australia. Produced by Christina Pozzan, Daniel Scharf, Ray Argall & Australian Film Commission, (1987).

[32] Thompson, L.A. & Massaro, D.W. Evaluation and integration of speech and pointing gestures during referential understanding. Journal of Experimental Child Psychology, 42 (1986), 144-168.

[33] Wanderley, M. Gestural Control of Music International Workshop Human Supervision and Control in Engineering and Music, Kassel, Germany (September 2001).

[34] Wanderley, M & Depalle, P. Gesturally-Controlled Digital Audio Effects. In Proceedings of the Cost G-6 Conference on digital Audio Effects (DAFX-01), Limerick, Ireland, (Dec-2001).

[35] Wanderley, M. Quantitative Analysis of Non-obvious Performer Gestures. Gesture and Sign Language in Human-Computer Interaction. Wachsmuth, I & Sowa, T. (eds.). Springer Verlag, 2002.

[36] 'Waveform' Concert. Australian Computer Music Conference. University of Western Sydney, Australia (July, 2001). http://www.acma.asn.au/

[37] Wessel, D., & Wright, M. Problems and Prospects for Intimate Musical Control of Computers. Computer Music Journal, 26:3 (Fall 2002), 11-22. http://www.csl.sony.co.jp/person/poup/research/chi2000wshp/papers/wessel.pdf

[38] White Wing Dove. Video. Directed by Marty Callner CBS/FOX, U.K. (1983).

[39] Winkler, T. Making Motion Musical: Gestural Mapping Strategies for Interactive Computer Music. Proceedings of the 1995 International Computer Music Conference. San Francisco, International Computer Music Association (1995), 261-264.

[40] Yamaha, (1996) Miburi R3 Manual. Tokyo, Japan, Yamaha Corporation (In Japanese, no author listed)

# Contexts of Collaborative Musical Experiences

Tina Blaine
Entertainment Technology Center
Carnegie Mellon University
Pittsburgh, PA 15123
bean@cs.cmu.edu

Sidney Fels
University of British Columbia
Dept. of Electrical & Computer Engineering
Vancouver, BC, V6T 1Z4
ssfels@ece.ubc.ca

## ABSTRACT

We explore a variety of design criteria applicable to the creation of collaborative interfaces for musical experience. The main factor common to the design of most collaborative interfaces for novices is that musical control is highly restricted, which makes it possible to easily learn and participate in the collective experience. Balancing this trade-off is a key concern for designers, as this happens at the expense of providing an upward path to virtuosity with the interface. We attempt to identify design considerations exemplified by a sampling of recent collaborative devices primarily oriented toward novice interplay. It is our intention to provide a non-technical overview of design issues inherent in configuring multiplayer experiences, particularly for entry-level players.

## Keywords

Design, collaborative interface, musical experience, multiplayer, novice, musical control.

## 1. INTRODUCTION

The emergence of electronic instruments, and most notably the computer, has led to the creation of new interfaces and sounds never before possible. In addition, the computer can be used to create arbitrary mappings between gesture and sound, thereby providing the possibility of computer-supported sound and directed musical interaction. Consequently, a wave of new types of collaborative interfaces and group experiences has emerged for collective music making with the potential to include people with little or no musical training. Therefore, understanding the role of music in relation to people's experiences playing collaborative instruments requires a shift in perspective. By attributing less relevance to the importance of traditional music metrics based on melody, more emphasis can be placed on metrics that involve the players' experience. The psychological state of "flow" is achieved by engaging in deeply satisfying experiences that alter one's state of consciousness [1]. Making collaborative interfaces relatively simple and easy to learn facilitates flow for novices. This approach can also support the development of intimacy with the interface, which has an "aesthetic of control" [2]. When designing collaborative musical experiences for first-time players in public places, the amount of time necessary to learn an interface must be minimized, coupled with achieving a balance between virtuosity and simplicity [3]. Providing an upward path of increasing complexity necessary for maintaining flow, while at the same time providing an entry level low enough for novices, is very challenging and continues to necessitate further inquiry by experience designers.

### 1.1 Accessible Music

The underlying premise of most collaborative interface design is that with various design constraints, playing music can be made accessible to non-musicians. Participation in making music gives players a sense of belonging and access to a new community at the expense of limiting the musical range and possible gestures associated with sound in a collective space. We suggest that analyzing the musical experience of collaborative interfaces should be examined in this context. Essentially, low-level accessibility is necessary for people to participate and communicate with the instruments and each other. Furthermore, many collaborative interfaces are intended for public exhibition, where people casually "walk-up and play". This restricts the amount of time that a designer can expect someone to spend learning an interface, and necessitates highly constrained interfaces that are conducive to easily accessible musical experiences.

Therefore, we suggest that providing novices with easily accessible music making experiences is more important than having a complex interface with built-in, upward capability for virtuosic expression. The counter-argument to this assumption is that a low entry fee should have no ceiling on virtuosity [4]. Wessel and Wright posit that "…many of the simple-to-use computer interfaces proposed for musical control seem, after even a brief period of use, to have a toy-like character and do not invite continued musical evolution" [4]. While this is fundamentally true for expert musicians, the main opposition to this viewpoint regarding novice interplay is that the demographic for most multiplayer instruments are non-musicians and accordingly, the same principles do not necessarily apply. Although expert musicians are concerned with expressive capabilities and mastery of their instruments, it is unlikely that first time players have the expectation of becoming expert players on any musical instrument.

### 1.2 Balancing Complexity and Expressivity

The trade-off in determining the appropriate balance of complexity and expressivity of an interface is not easily resolved. Historically, the field of musical controllers has advanced primarily through the creation of highly complex single player instruments developed for experts, as opposed to multiplayer interfaces/environments designed for novices [5] [6]. Developing musical interfaces using familiar objects that ordinarily serve another purpose, or inventing entirely new instruments, can change the level of musical expectation by redefining "expert" and "novice" interplay as the basis for engagement. . "Playful" interfaces can also avoid the look and feel of traditional instruments [7]. Designers of collaborative devices that are easy to control but have limited expressive capabilities are challenged not only to conceive of opportunities for musical exploration, but must also cultivate meaningful social interactions and experiences for the players. In a collaborative musical environment, it becomes even more imperative that the technology serves primarily as a catalyst for social interaction, rather than as the focus of the experience [8]. Conversely, interfaces that have extended expressive capabilities tend to be more difficult to control and cater more to the expert player. For designers of most musical interfaces, the overriding challenge is to strike a balance of multimodal interaction using discrete and continuous controls [9], [10], and generally, limit rather than increase the number of features and opportunities for creativity [7].

### 1.3 Mapping and Control Issues

Natural mapping behaviors evolve from the creation of a direct relationship between gesture and musical intent. Players' perception of control in collaborative musical environments can be increased by creating predetermined musical events, subject to players manipulating complex parameters of sound through gestures, such as stretching or squeezing [11]. Enhancing the illusion of control can also be achieved with supplemental effects such as lighting, visual imagery and more, to create a highly responsive system based on player input. While the use of pre-composed musical events or sequences severely limits certain aspects of an individual's creative control, it has the benefit of creating more cohesive sound spaces in multiplayer environments. With these mappings, players are not responsible for playing specific notes, scales or harmonies, which helps to minimize chaotic musical interaction.

## 2. CONTEXTS OF COLLABORATIVE INTERFACES

Collaborative musical interfaces may be roughly classified by a number of different attributes unique to the context of communal experience. Table 1 provides a sample listing of multiplayer systems organized by the following elements of design: **Focus**, **Location**, **Media**, **Scalability**, **Player Interaction**, **Musical Range**, **Physical Interface**, **Directed Interaction**, **Pathway to Expert Performance** and **Level of Physicality**.

Design issues regarding the input interface, input-to-output mapping and the output interface are of the utmost relevance as well as the topic of much research.[1] Thus, the type of collaborative interface depends on a number of factors including range, sensor(s), directed interaction, and pathway to expert performance. Good design practice for these instruments, whether cooperative or not, overlaps with issues regarding human-computer interaction [12]. Such issues include usability, ease of learning, and functionality, specifically in relation to their effects on the success of the *collaborative* experience. Finding the balance between virtuosity and simplicity provides fertile ground for new collaborative interfaces. Due to space constraints, the authors were unable to include a more comprehensive list, or technical discussion regarding the systems referenced herein.

### 2.1 Focus

The focus of the experience is determined by establishing whether the communication is primarily between players or between players and an audience. Collaborative instruments are usually designed to enhance the communicative experience between players rather than exploit virtuosic play for the benefit of an audience. This may or may not be very interesting for an audience to listen to, since they are not privy to the subtleties of interaction that occurs between players. Most computer-based instruments do not provide direct means for audiences to see how players' gestures affect the music and instead must rely upon indirect means, such as explanation of the interaction or visualization.

### 2.2 Location

Many collaborative interfaces for musical expression are created as installations for public exhibition. In these instances, people are often expected to converge at a specific location and/or gather around an instrument to play together. Because they are co-located, players can see each other's gestures and more readily understand the relationship between each player's actions and the sounds produced. However, if the sounds are not easily attributable to specific actions or devices, then players must find other ways to communicate. *Beatbugs* [13], *Musical Trinkets* [14], and *SoundMapping* [15], all work around this issue in a variety of ways. With the growth of the Internet, a new genre of collaborative interfaces allows players to communicate over a network from non-specific locations, from virtually anywhere in the world [16]. Systems such as the *Hub* [17], *Brain Opera* [18][19], *Faust Music OnLine* (FMOL) [20], and *Rocket Network* [21], are notable examples of efforts in this direction that integrate(d) more professional levels of musicianship.

### 2.3 Media

Many collaborative interfaces combine audiovisual elements as a way of enhancing communication and creating more meaningful experiences. The use of visual imagery can facilitate the collaborative experience by reinforcing the responsiveness of the system to players' actions. However, visual imagery can also distract players from seeing other players' actions, or from attending to aural elements, or both. Some of the systems that include visual imagery as the primary medium include *Jamoworld* [22], *Jamodrum* [23], *Iamascope* [24], and *Currents of Creativity* [3]. One particular challenge with visually oriented systems, is that the identification of players with imagery can be so strong that the act of making music becomes a secondary part of the experience.

### 2.4 Scalability

By their very nature, collaborative interfaces are designed for a minimum of two or more players. However, the number of players greatly influences the types of interfaces and music that is appropriate. An interface built for two people is generally quite different from one built for tens, hundreds or thousands of players. When considering scale, factors such as turn-taking protocols and gesture-sound correspondences shift as the number of players increase. For example, it does not make sense to expect turn-taking protocols to emerge in an interface with three hundred drum pad inputs distributed through a large area, as embedded in the *RhythmTree* structure [18]. Directly refuting this notion is the *MidiBall* [25] interface, where only a few people are physically able to hit the ball at one time, even if hundreds or thousands of people are present.

### 2.5 Player Interaction

Generally, collaborative instruments provide each player with a method for individual control within a shared sonic environment. Although the control devices may be identical or different for each player, the underlying method of interaction is quite often the same. For example, in *Musical Trinkets* [14] and Musical Navigatrics [26], each player has their own unique set of figures used to control sound. While each trinket has a specific sound or algorithmic effect associated with it, all players interact in the same way, by moving the objects over a shared tabletop surface in order to activate those sounds. In a communal space without too many people and/or distractions, this approach has the advantage that players are able to observe each other to determine what distinguishes each player's visual and aural impact. However, if the mapping between the interface or device and its affect on the sonic output is unclear, then it becomes more difficult to use the interface for musical collaboration.

---

[1] Organized Sound special issue on mappings and the New Interfaces for Musical Expression (NIME) proceedings all address these design issues.

| System | Focus | Location | Media | Scale | Player Inter-action | Musical Range/ Notes | Physical Interface/ Sensor | Directed Inter-action | Learning Curve | Pathway to Expert Perform-ance | Level of Physical-ity | Musical Genre |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Audio Grove** (Moeller, 1997) | Players | Local | Sound, Light, Device | 1-30 | Same | Players control DSP | Touch, Capacitive sensing | Low | Fast | No | High | Ambient |
| **Augmented Groove** (Pouprev et al., 2001) | Players | Local | Sound, Image, Device | 1-3 | Same | Players control DSP | Camera, HMD, Glyph Disks | Med-High facilitator | Med-Fast | No | High | Techno, House |
| **Beatbugs** (Weinberg et al., 2002) | Players + Aud-ience | Local | Sound, Device | 1-8 | Same | Players control DSP + rhythmic input | InfraRed, Bend sensors, Piezos | High workshops + dist'd leadership | Slow | Possibly | High | Electronic Poly-rhythmic |
| **Brain Opera** (Machover, 1996) | Players + Aud-ience | Local and Net | Sound, Image, Device | 1-100's | Differ-ent | Limited & Unlimited | Varied Custom Devices | Conductor, facilitators + freeplay | Slow - Fast | Possibly | Med-High | Varied |
| **Bullroarer** (Robson, 2001) | Players | Local | Sound, Device | 1-3 | Same | Players control DSP | Sliders, potentio-meters | Low | Fast | No | High | Ambient Drones, Electronic |
| **Composition on the Table** (Iwai, 1998) | Players | Local | Image, Sound, Light, Device | 1-6 | Same | Players control rhythm + midi loops | Buttons, Switches, Faders | Low | Fast | No | Med | Minimalist |
| **Currents of Creativity** (D'Arcangelo, 2001) | Players | Local | Image, Sound, Device | 1-6 | Same | Limited: pre-composed loops | Computer Kiosk | High | Fast | No | Med | World |
| **FMOL** (Jorda, 1999) | Players | Net | Sound, Image, Software | 2 | Same | Unlimited | Mouse, Kybd | No | Medium | Yes | Low | Electronic |
| **Hub** (Gresham-Lancaster, 1998) | Aud-ience | Local and Net | Sound, Soft-ware | 1-6 | Differ-ent | Unlimited | Mouse, Keyboard, Joysticks Trackball + MIDI Devices | No | Slow | Yes | Low | Electronic |
| **Iamascope** (Fels and Mase, 1998) | Players | Local | Image, Sound | 1-3 | Same | Limited | Camera | Low | Fast | No | High | Simple Melody |
| **Jamodrum /Jamoworld** (Blaine & Perkis, 2000) (Blaine & Forlines 2002) | Players | Local | Image, Sound | 1-12, 1-4 | Same | Limited, Midi + Pre-composed loops | Drumpads + turntable disks | Med - High: virtual facilitator, Dist'd leadership | Fast | No | High | World, SFX, percussion samples |
| **MidiBall** (Jacobson, Blaine, and Pacheco, 1993) | Players are the Aud-ience | Local | Sound, Image, Device | 1-1000s | Same | Limited | Custom Device +RF | Low | Fast | No | High | Vox Samples, variable |
| **Musical Trinkets /Navigatrics** (Paradiso et al., 2001), (Pardue and Paradiso, 2002) | Players | Local | Sound, Device | 1-5 | Same | Players control DSP | Passive RF Tags | Med-High facilitator | Fast | No | High | Beat mix |
| **Rhythm Tree** (Paradiso, et al., 2001) | Players | Local | Sound, Lights, Device | 1 – 50 | Same | Limited | Drum Pads | Low | Fast | No | High | Percussion & Vox Samples |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Sound Mapping** (Mott, Sosnin, 1997 | Players | Local | Sound, Device | 1-4 | Same | Players control timbre, pitch + rhythm | GPS, tilt, Accelero-meters | Med-High | Fast | No | High | Ambient |
| **Speaking Orbs** (Ask, 2001) | Players | Local | Sound, Device | 1-8 | Same | Limited | Photo-resistors | Low | Fast | No | High | Ambient |
| **Squeezables** (Weinberg and Gan, 2001) | Players + Aud-ience | Local | Sound, Device | 1-3 | Same | Players control DSP | FSR's, Potentio-meters, Variable resistors | Med-High | Fast | No | High | Ambient World, Drum & Bass |
| **Tooka** (Fels and Vogt, 2002) | Players + Aud-ience | Local | Sound | 2 | Same | Limited | Breath | No | Slow | TBD | High | Open |

**Table 1: Contexts of Collaborative Interface Design**

## 2.6 Musical Range/Notes

The most common technique used to provide an easily learned interface is to limit the range of notes or sounds that any action creates. Group dynamics and social interaction are consistently achieved by limiting the players' opportunities for extended musical exploration, and in many cases, directing the players' interaction. For example, providing players with short musical phrases, percussion loops, or melodies that are constrained by key, tempo or rhythm are proven methods of designing a limited range of elements that can still be satisfying and fun to play. A number of the experiences such as *Augmented Groove* [27], *Composition on the Table* [28], *Audio Grove* [29], MusiKalscope [30], *Bullroarers* [8], *Musical Trinkets* [14], and *Squeezables* [11], approach limiting the potential for chaotic musical interaction between players by adding control over effect algorithms of pre-composed or algorithmically generated music. A few commonly used effect-algorithm-control-parameters include volume, modulation, pitchbend, tremolo, delay, and echo, in addition to numerous other digital signal processing effects and filters that affect the timbral qualities of predetermined sound elements.

## 2.7 Physical Interface/Sensor

Designers of collaborative instruments can choose from an extensive selection of sensors, software and signal processing options. Joysticks, ultrasound, infrared, accelerometers, potentiometers, force-sensitive resistors, piezos, magnetic tags, and many more sensor technologies are available to those interested in converting voltage data into MIDI or routing signals through other sound synthesis systems such as Max/MSP™[2], SuperCollider[3] or Open Sound World[4]. Measuring changes in motion, light, gravity, pressure, velocity, skin conductivity or muscle tension are just a few of the ways that a player's gestural input can be turned into musical output. The ways in which a physical interface and sensors are integrated are of primary importance as they provide the affordances [31] that make the interaction obvious to the novice. For example, when someone encounters the spongy objects known as *Squeezables* [11], the immediate response is to manipulate and squeeze these soft toy-like sculptures, thus affecting the musical outcome of these instruments. Conversely, the Iamascope does not have a tangible interface, but invites the player with a visual display, as a camera tracks their motions. As another example, players simply wave their hands between the opening of the *Speaking Orbs* [32] and a reflective light to trigger an array of windchime sounds via photo-resistors that send MIDI "note on" and "note off" messages.

## 2.8 Directed Interaction

Group dynamics and social interplay for novices is often achieved by directing the players' interaction. *Augmented Groove* [27] , *Beatbugs* [13], *Musical Trinkets* [14], and *SoundMapping* [15] are experiences that initially provide a knowledgeable person to assist the players. Another effective method for constraining the musical space is accomplished through distributed leadership [33] and turn-taking behaviors. *Beatbugs* [13], integrates different play modes with session leaders who "pass" rhythmic motifs amongst the group to enable real-time manipulation and response to sonic events. The *Jamodrum* [23] software elicits a "call and response" behavior as a means of orchestrating the players' experience and allowing opportunities for individuals to take turns in order to hear their contributions to the overall mix. The *Tooka* [34], was specifically designed for two players with the idea of suspending the need for turn-taking protocols entirely. In other experiences such as *Currents of Creativity* [3], software limits the player's interactions.

## 2.9 Pathway to Expert Performance

Ideally, a collaborative musical instrument would be initially easy to learn. On the other hand, musical expression is something that requires mastery of an instrument before subtlety can be achieved. Over time and with practice, a player can continue to refine their range of musical expression and become an expert. Traditional acoustic musical instruments have different entry levels for players to become musically adept. However, they all share the capacity to provide subtle forms of musical expression as players develop their skills. Supporting a pathway to expert performance is difficult because the ease of learning is often realized by restricting the range of musical possibilities available to the player through computer-mediation. Nevertheless, it is exactly this broader range of musical possibilities that is necessary for expressive expert performance. The evaluation of any collaborative instrument necessitates balancing this trade-off between speed of learning and musical capability.

## 2.10 Level of Physicality between Players (and Interface)

The availability of new sensors and computer interfaces for building novel musical controllers allows the creation of instruments that can involve virtually every part of the human

---

[2] Max/MSP is a trademark of Cycling '74, 379A Clementina Street, San Francisco, CA 94103.

[3] Available at: http://www.audiosynth.com

[4] Available at: http://www.cnmat.Berkeley_EDU/OSW

body including brain waves, muscle activations [9] and tongue movements [35]. Many collaborative instruments encourage various levels of movement, gesture, touch, and physical interactions such as dancing with strangers in highly customized environments. These design strategies lay the foundation for developing intimate personal connections with other players and their instruments over relatively short periods of time, and also help foster a sense of community. Frequently, it is the group ambience and development of synergistic relationships between players, rather than the interface itself, that leads to positive communal experiences.

## 3. CONCLUSION

**"Interactive instruments embody all of the nuance, power, and potential of deterministic instruments, but the way they function allows for anyone, from the most skilled and musically talented performers to the most unskilled members of the large public, to participate in a musical process." (Chadabe, 2002)** [36]

In conclusion, there are many challenging issues only beginning to be understood as they relate to the experience of collaborative instruments and computer-mediated experiences. Crafting interaction to create a satisfying and aesthetic musical encounter relies on the fulfillment of the basic qualities of social desire and human experience. Finding a balance between ease-of-learning, type of control (i.e. discrete versus continuous control), level of cross-modal interaction and support of virtuosity varies for every instrument and interface, depending on the functionality designers address. Issues of complexity and simplicity must be balanced as well. Building in enough depth to sustain interest while providing easy entry for first-time players is challenging in any environment. Multimodal inputs can assist with easy access for novices and still provide greater depth of expression for musicians. The reality of designing for public spaces is that an installation's flow-through capacity may translate into people having as little as three to five minutes to experience the act of playing music together.

Particularly when designing for novice players, it seems clear that the overriding similarity between systems is that the overall *experience* takes precedence over the generation of music itself. Music and sound are still significant aspects of the experience, but the ability to control individual notes, harmonies, melodies, and so forth, is not the most important factor to a non-musical person in determining whether or not an interface is engaging. The opportunities for social interaction, communication, and connection with other participants is of paramount importance to the players' comfort with the interface. Ultimately, this will lead to a sense of community, even with strangers, in a public setting. While the affordances of the sensors and interface should be transparent to the players, understanding their individual impact on the system is critical. This can be achieved through the use of music, lights, images, sound effects, or a broad range of other possibilities; anything that supports the intentions of the players will serve to reinforce the perception of a highly responsive system.

## 4. REFERENCES


[1] Csikszentmihalyi, M. (1990), Flow: The Psychology of Optimal Experience, Harper Perennial, 1990.

[2] Fels, S. (2000), Intimacy and Embodiment: Implications for Art and Technology, *Proceedings of the ACM Conference on Multimedia*, 2000, pp. 13-16.

[3] D'Arcangelo, G. (2001), Creating Contexts of Creativity: Musical Composition with Modular Components, *In Proceedings of the 1st Workshop on New Interfaces for Musical Expression (NIME01)*, ACM SIGCHI, 2001, electronic proceedings.

[4] Wessel, D., and Wright, M. (2001), Problems and Prospects for Intimate Musical Control of Computers, *In Proceedings of the 1st Workshop on New Interfaces for Musical Expression (NIME01)*, ACM SIGCHI, 2001, electronic proceedings.

[5] Paradiso, J. (1997), Electronic music interfaces: new ways to play, IEEE Spectrum Magazine, Vol. 34, No. 12, 1997, pp. 18-30.

[6] Cutler, M., Tobari, G., and Bean (2000), "OuterLimits," *Electronic Musician Magazine*, August 2000, pp. 49-72.

[7] Cook, P. (2001), Principles for Designing Computer Music Controllers, *In Proceedings of the 1st Workshop on New Interfaces for Musical Expression (NIME01)*, ACM SIGCHI, electronic proceedings.

[8] Robson, D. (2001), PLAY!:Sound Toys For the Non Musical, *In Proceedings of the 1st Workshop on New Interfaces for Musical Expression (NIME01)*, ACM SIGCHI, electronic proceedings.

[9] Tanaka, A. and Knapp, R.B. (2002), Multimodal Interaction in Music Using the Electromyogram and Relative Position Sensing, *In Proceedings of the 2nd International Conference on New Interfaces for Musical Expression(NIME02)*, 2002, pp. 43-48.

[10] Verplank, B. (2001), A Course on Controllers, *In Proceedings of the 1st Workshop on New Interfaces for Musical Expression (NIME01)*, ACM SIGCHI, 2001, electronic proceedings.

[11] Weinberg, G. and Gan, S. (2001), The Squeezables: Toward an Expressive and Interdependent Multiplayer Musical Instrument, Computer Music Journal, 25:2, 2001, pp. 37–45.

[12] Orio, N, Schnell, N, Wanderley, M. (2001), Input Devices for Musical Expression: Borrowing Tools from HCI, *In Proceedings of the 1st Workshop on New Interfaces for Musical Expression (NIME01)*, ACM SIGCHI, 2001, electronic proceedings.

[13] Weinberg, G., Aimi, R., and Jennings, K. (2002), The Beatbug Network: A Rhythmic System for Interdependent Group Collaboration, *In Proceedings of the 2nd International Conference on New Interfaces for Musical Expression(NIME02)*, 2002, pp. 107-111.

[14] Paradiso, J., Hsiao, K., Benbasat, A. (2001), Tangible Music Interfaces using Passive Magnetic Tags, *In Proceedings of the 1st Workshop on New Interfaces for Musical Expression (NIME01)*, ACM SIGCHI, electronic proceedings.

[15] Mott, I. and Sosnin, J. (1997), Sound Mapping: An Assertion of Place, Interface'97: http://www.reverberant.com/SM/paper.htm accessed on Oct 18, 2002.

[16] Weinberg, G. (2002), The Aesthetics, History, and Future Prospects of Interdependent Music Networks, Computer Music Journal, Göteborg, Sweden: International Computer Music Association, 2002, pp. 349-356.

[17] Gresham-Lancaster, S. (1998). "The Aesthetics and History of the Hub: The Effects of Changing Technology on Network Computer Music." Leonardo Music Journal vol. 8, 1998, pp. 39-44.

[18] Paradiso, J. (1999), The Brain Opera Technology: New Instruments and Gestural Sensors for Musical Interaction and Performance, *Journal of New Music Research*, 28(2), 1999, pp. 130-149.

[19] Machover, T. (1996), Brain Opera, In Memesis: The Future of Evolution. Ars Electronica Editions, Linz, Austria, 1996.

[20] Jordà, S. (1999), Faust Music On Line: An Approach to Real-Time Collective Composition on the Internet, *Leonardo Music Journal,* Vol. 9, 1999, pp. 5-12.

[21] Hall, G. (2002), Bands Without Borders, Electronic Musician Magazine, October 2002, pp. 72-86.

[22] Blaine, T. and Forlines, C. (2002), JAM-O-WORLD: Evolution of the Jam-O-Drum into the Jam-O-Whirl Gaming Interface, *In Proceedings of the 2nd International Conference on New Interfaces for Musical Expression(NIME02)*, 2002, pp. 17-22.

[23] Blaine, T., and Perkis, T. (2000), The Jam-O-Drum Interactive Music System: A Study in Interaction Design. *DIS2000 Conference Proceedings*, 2000, pp. 165-173.

[24] Fels, S. and Mase., K. (1999), Iamascope: A Graphical Musical Instrument, *Computers and Graphics*, Vol. 2, No. 23, 1999, pp. 277-286.

[25] Jacobson, L., Blaine, T., and Pacheco, C. (1993). Time for Technojuju, *New Media Magazine*, January, 1993, pp.18.

[26] Pardue, L. and Paradiso, J. (2002) Musical Navigatrics: New Musical Interactions with Passive Magnetic Tags. *In Proceedings of the 2nd International Conference on New Interfaces for Musical Expression (NIME02)*, 2002, 168-176.

[27] Poupyrev, I., Berry, R., Billinghurst, M., Kato, H., Nakao, K., Baldwin, L., Kurumisawa, J. (2001), Augmented Reality Interface for Electronic Music Performance. In proceedings of the 9th International Conference on Human-Computer Interaction (HCI International 2001), 2001, pp. 805-808.

[28] Iwai, T. (1998), Composition on the Table, exhibition at Millennium Dome 2000, London, UK, 1998.

[29] Möller, C. (1997), Audio Grove, *Exhibition: Spiral Art Center*, Tokyo, May, 1997, http://users.design.ucla.edu/projects/arc/cm/cm/staticE/page8.html accessed on Nov. 7, 2002.

[30] Fels, S., Nishimoto, K., and Mase, K. (1997), MusiKalscope: A Graphical Musical Instrument, *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, 1997, pp. 55-62.

[31] Norman, D. (1990), The Design of Everyday Things, Currency/Doubleday, 1990.

[32] Ask, E. (2001), Speaking Orbs - Interactive Multi-Participant Sound Sculpture, *Demonstration at the 1st Workshop on New Interfaces for Musical Expression (NIME01)*, ACM SIGCHI, 2001, electronic proceedings.

[33] Cirigliano, G. and Villaverde, A. Dinámica de grupos y educación: fundamentos y técnicas. Buenos Aires, Argentina. Editorial Hvmanitas, 1966.

[34] Fels, S. and Vogt, F. (2002), Tooka: Exploration of Two Person Instruments, *In Proceedings of the 2nd International Conference on New Interfaces for Musical Expression(NIME02)*, 2002, pp. 116-121.

[35] Vogt, F. and McCaig, G. and Ali, A. and Fels, S. (2002), Tongue 'n' Groove. *In Proceedings of the 2nd International Conference on New Interfaces for Musical Expression(NIME02)*, 2002, pp. 60-64.

[36] Chadabe, J. (2002), The Limitations of Mapping as a Structural Descriptive in Electronic Instruments. *In Proceedings of the 2nd International Conference on New Interfaces for Musical Expression (NIME02)*, 2002, pp. Keynote-2-i-v.

# MidiGrid: Past, Present and Future

Andy Hunt
Media Eng. Group
Dept. Electronics
University of York
Heslington, York
YO10 5DD, U.K.
+441904 432375
adh@ohm.york.ac.uk

Ross Kirk
Media Eng. Group
Dept. Electronics
University of York
Heslington, York
YO10 5DD, U.K.
+441904 432375
ross@ohm.york.ac.uk

## ABSTRACT

MidiGrid is a computer-based musical instrument, primarily controlled with the computer mouse, which allows live performance of MIDI-based musical material by mapping 2-dimensional position onto musical events. Since its invention in 1987, it has gained a small, but enthusiastic, band of users, and has become the primary instrument for several people with physical disabilities. This paper reviews its development, uses and user interface issues, and highlights the work currently in progress for its transformation into MediaGrid.

## Keywords

Live performance, Computer-based musical instruments, Human Computer Interaction for Music.

## 1. INTRODUCTION

The MidiGrid project was an experiment to investigate the design of a new computer-based interface to electronic tone generators. It was started at the University of York UK in 1987, and has been developed in stages since then [1][2][3]. MidiGrid has been used by a wide range of people (composers, schoolchildren, and special needs teachers and their clients) and its use has raised several important issues relating to the design of interactive musical systems. A summary of its key features is now given, followed by a discussion of the issues that arise from its use.

## 2. THE MIDIGRID CONCEPT

Many computer programs with mouse control exist to allow musical information to be stored and edited on a computer, rather like a musical word processor. These programs are often called (sequencers, editors) as they enable people to build up sequences of music, track by track, usually in non-real time.

In contrast MidiGrid allows users to trigger musical material freely in real-time using the computer's mouse. As shown below, the screen shows a grid of boxes, each of which contains a nugget of music that is triggered as the mouse cursor moves over it.

Hand gestures are thus converted, via the mouse, into notes, chords and musical sequences.

The range of movement can be customised so that more or less of the user's physical action can move the mouse cursor around the grid. The grid can be set up in advance to consist of any number of boxes containing any musical material (including that played in from a keyboard).

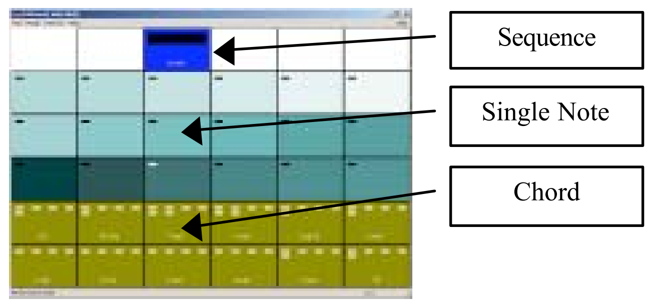



**Figure 1. Main MidiGrid performance screen (PC version) showing sequences, single notes, and chords.**

The grid can be shaded to separate different areas of notes, for example (in figure 1, above) some areas consists of melodic notes whilst others contain chords (denoted by more than one dot in a grid box.

In fact, the grid can be arranged to allow an assortment of instrumental sounds to be present on the screen. Thus the user can freely explore several timbres by moving the mouse to different areas of the screen. Anything that is played on the grid can be recorded and placed into a box of its own as a sequence. Further recordings can be made which involve sequences, and thus complex layers of musical material can be rapidly constructed.

### 2.1 Customizing the grid for performance

The grid is of a user-definable size and shows the musical contents of each box by simple graphical representations of notes and sequences. Thus the layout of the cells can be customized for each player, performance or musical use. The grid pattern shown in Figure 2 contains many shaded areas, each containing sound elements on different timbres.

The player sweeps a cursor (using the computer mouse) around the screen and notes are triggered when the mouse buttons are pressed. Consequently identical gestures produce identical musical results, but these gestures have to be learnt and rehearsed by the player.

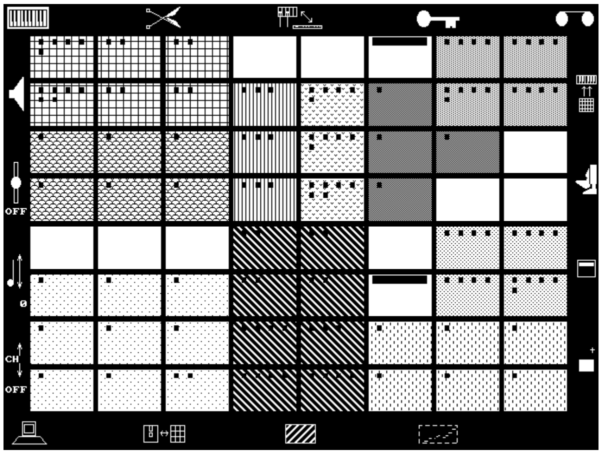



**Figure 2. Grid pattern showing a variety of shaded areas (Atari version), each containing a different timbre.**

The system uses the MIDI protocol for controlling notes on an external synthesiser or sampler. Versions of the software since 1990 have allowed a variety of MIDI continuous controllers to be sent in response to mouse movement. The effect of this is to permit much more subtle forms of musical control using gestures to bend the pitch, sweep the panning position and swell the volume, using the same gestural input to choose and control the notes.

## 2.2 External Control

Other MIDI instruments (e.g. electronic keyboards, drums or wind controllers) can be used to trigger the musical material in the boxes, so - for instance - notes on a keyboard can be used to activate several pre-recorded sequences of music. This feature allows MidiGrid to be 'remote-controlled' by any other MIDI-compatible device. This has yielded some of its most interesting uses over the years, as it moves the player away from the computer terminal and allows them to concentrate on a physical performance interface. One of the more popular configurations has become known as 'Carpet-Grid'[5].

MidiCreator [6][7] is a device which converts the various signals from electronic sensors into MIDI. Based on a music technology student's project, it was subsequently developed by the York Electronics Centre, the commercial arm of York University's Electronics Department. Assorted sensors are available which sense Pressure, Distance, Proximity, Direction etc. These are plugged into the front of the unit, which can be programmed to send out MIDI messages corresponding to notes or chords. Thus movement is converted to music.

When a grid of pressure sensors is placed on the floor a 'carpet-grid' is formed. Each pad can trigger a note on a specified instrumental sound. When these notes are routed through the abovementioned MidiGrid software, entire musical sequences can be triggered from different areas of the floor. This forms a fascinating 'floor-based' instrument which people of moderate movement can explore. In some cases people have driven their electric wheelchairs over it to achieve the same effect.

## 2.3 MidiGrid version history

MidiGrid's original and most fully developed platform was the Atari ST. During the 1990s a series of ports were made to the PC platform and the most stable of these is available for a trial download from `www.midigrid.com`. This version does not have the comprehensive real-time controller implementation or piano-roll editing of the Atari version, but embodies most of the main grid-cell features for live performance of MIDI material.

## 2.4 Context: Computers in Live Performance

Since the introduction of digital computing technology to the art of music in the late 1950s, there have been several strands of research which use the computer as a performance instrument. The most common thread is the addition of a specifically designed control interface. Axel Mulder [16] and Joe Paradiso [17] each describe the various forms that these devices tend to take – mostly in emulation of existing acoustic gestural models, such as wind interfaces, keyboards, guitars or conducting devices. They also note the less familiar concept of the using the computer's native interfaces (specifically the mouse and the keyboard) as live performance tools.

A number of systems have been developed over the last fifteen years for using the computer mouse as a means of triggering and controlling real-time sonic material. MidiGrid is one such system, so is Music Mouse[4] (see section 2.4 below). Other systems which use similar paradigms include Fleximusic [18] (allows keys to trigger sound and MIDI files), and MousMuso [19] (mouse is used to strum virtual melodies and harmonies). More recent systems use the concept of pre-packaging musical material into a form ready to be triggered in live performance, but using a specially designed physical interface. A particular example to note is the BlockJam project [20] which uses a graspable block-structured interface to allow several users to trigger audio samples and algorithms in real-time, with graphical and tactile feedback.

## 2.5 Music Mouse

Comparisons have occasionally been made between MidiGrid and Laurie Spiegel's Music Mouse [4]. This is hardly surprising since both pieces of software were developed (in different continents!) at about the same time, and both allow the user to move the mouse to make real-time musical improvisations.

However, the programs and the concepts are quite different, and I have discussed this with Laurie some years ago and more recently during the production of this paper. Music Mouse uses a level of 'intelligence' to provide an interactive environment within which users can improvise using different mouse gestures.

In contrast MidiGrid provides no interpretive intelligence; the boxes (and the musical material contained within) are simply triggered when the cursor passes over them or the mouse is clicked on them. So, MusicMouse 'joins in' your improvisation, whereas MidiGrid simply reproduces the stored musical material on demand. You probably have to work harder with MidiGrid to create a coherent sounding polyphony.

Despite these differences, Music Mouse and MIDI Grid do share something that has turned out to be far less common among music programs than might have been expected by either of their authors when these programs were first created. A core value of both is the satisfying immediacy of sound responding directly to human movement and touch that has been central to most successful human interfaces to musical sound for millennia. Both programs place the computer in the role traditionally given to musical instruments, rather than seeing the computer as a tool for storage and editing of materials to be played subsequently via some other instrument[26].

# 3. USES OF MIDIGRID

MidiGrid has been used by a variety of people from many walks of life. Whilst it has been used for triggering sequences and notes in live electroacoustic performances, it seems to have found its niche with people who benefit from the pre-storage of musical material – yet under live performance control. Many of these users have been people with physical disability as the mouse movement and grid layout can be customized to suit individual gestural capacity.

## 3.1 Music Therapy

Music Therapy is an increasingly popular form of clinical practice which engages client and therapist in dynamic interaction without necessary recourse to words, by involving the power and universality of music. Gary Ansdell, in the book "Music for Life"[22], describes the main function of Music Therapy as being for the therapist to hear, respond and answer, while the client experiences being heard, being responded to and being answered. *"The aim of Creative Music Therapy is to benefit people by giving them access to a creative music relationship within a sustained and dependable therapeutic context"*[22].

Many of the branches of Music Therapy make use of improvisation sessions involving both clients and therapist. It is therefore important that the client has access to a device which enables real-time musical interaction. Traditional acoustic musical instruments are customarily used, but cause problems when clients have restricted movement or weak muscles. This is where the use of electronic-powered music technology devices becomes important. Surprisingly little work has been done to provide technology in such an improvisational context for therapy, but Phil Ellis has been working with the direct use sound for such purposes for a number of years[23][24]. Fitzwilliam describes some of the potential of electronic technology, along with several practical reasons why technology has not been as popular with music therapists[25]. Most of these relate to the overt technicality (wires, menus, programming, setting up) that seem to be required by many electronic instruments, in contrast to the simplicity and directness of an acoustic instrument.

In Music Therapy, MidiGrid has been used [7][8][9][10] to allow free improvisation on a palette of sounds. The sounds have been pre-chosen by the therapist to constrain the musical material to a particular genre, tonality or timbre set. This has allowed access for people with limited movement to trigger self-consistent performance material. One of the outcomes is that where several acoustic instruments are used in the same piece aswell, MidiGrid can be effectively 'pre-tuned' so that, for example, it is in the same tonality as the chime bars. This enables groups of musicians to play together.

MidiGrid was built into the mobile Music Therapy van, devised by Mary & Raymond Abbotson [11] and used as part of the North Yorkshire Music Therapy centre's service in the UK over a number of years.

Outside of clinical Music Therapy, MidiGrid has been used on a number of occasions to allow people with limited movement to access musical material in an immediate way.

Figure 3 shows a grid pattern that consists of a web of individual notes which form scales (when played up and down) and arpeggios (when played across).

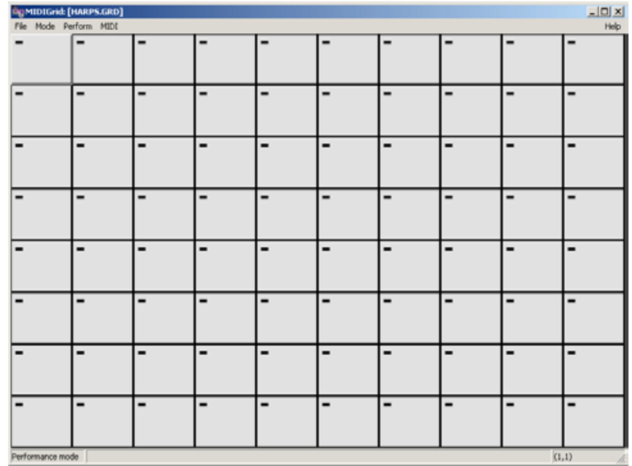



**Figure 3. Grid pattern containing harp notes arranged in arpeggios and scales.**

A harp-like sound is used, and flurries of notes can be easily generated by gentle mouse movements. In a series of tests in the children's centre at York District hospital, a 2 year old blind girl with severe learning impairment moved the mouse rapidly and even began talking to it. Therapists noted that this was her longest recorded concentration span without one of her regular seizures.

In a concert by the Drake Music Project [21] several people with movement difficulties each played a MidiGrid as part of a live performance with conventional (professional) musicians at London's Millennium Dome. For some people, MidiGrid has become their primary instrument, the tool that has enabled them to contribute to the musical world.

# 4. INTERACTION ISSUES RAISED

Several issues have emerged from personal experimentation with MidiGrid and from watching others performing with it. These issues have implications for the design of computer instruments and of new forms of interactive human computer interfaces in general [12].

## 4.1 Learning an instrument

Players of musical instruments have always required considerable dedication and commitment to hard work and rehearsal in order to learn how to play well. During the difficult times, particularly at the start of this process, it is the inspiration of watching an accomplished musician perform on the instrument that provides the motivation for continuing to practice for long periods of time.

Therefore we should assume that if a computer interface demands more than a surface level of operation, users should be expected to spend long periods of time learning the dynamics of how to 'drive' it. Many computing interfaces are, however, based on the assumption that users do not need to learn it, since they navigate the menu system and interpret accordingly each time they want to access a certain function.

## 4.2 Configurable Instruments

MidiGrid can be customised by the user to produce individual grid patterns of different size and complexity, containing whatever layout of performance material is required. This flexibility of configuration has been responsible for MidiGrid's successful use in schools and for various Music Therapy situations. Teachers and therapists can devise, restrict or expand the musical material that is available to the end-user. In doing so they form customised musical environments where the tonality, instrumentation and physical layout of the notes (and thus the type of hand gestures used to play them) are defined for a particular music/client combination.

However, there is a danger that players will never learn to control the instrument beyond a surface level of exploration because the 'goalposts are constantly being moved'. Players of traditional acoustic instruments undergo a good deal of configuration themselves in the process of learning to control their instrument! Generally, if we allow system interfaces to be continually reconfigured, we are perhaps in danger of removing any reason for human operators to work hard at learning to control the system interactively. We should perhaps set up an instrument for a particular situation and then always use that configuration with that particular situation. After all a cymbal or a drum does not change its character from one session to the next.

## 4.3 Necessity of graphics for musical instrument control

One aspect of a developing control intimacy shown by a traditional instrumentalist is a decreasing reliance on visual cues.

*"Novice users of MidiGrid frequently request that material be annotated so that they may remember the location of material within the grid. This is analogous to the labeling of a piano keyboard with the letter names of the notes on the stave. Observation of competent pianists will quickly reveal that they do not even look at their fingers, let alone any annotation which may be associated with the keys".* [3]

As users develop their musical performance ability on a particular instrument, they rely increasingly on tactile, audio and kinaesthetic feedback, and less on graphical information. Therapists cannot be expected to constantly stare at a computer screen in order to operate the program without breaking the concentration and eye contact that is so vital for effective musical communication. However, there is also a general lesson here for the designers of human-computer interfaces in high-performance systems; graphics are a useful way of presenting information (especially to beginners) but are not necessarily the primary channel which humans use when they are fully accustomed to the system.

## 4.4 'Performance Mode'

When users are performing with MidiGrid there is no 'dialogue' between user and computer, instead the computer responds instantly to the user's hand movements. The computer does not set the agenda or dictate the conversation or insist the users select from a set of predefined options, but instead provides an environment for creative exploration. This is very close to the concept of 'flow', coined by Mihaly Csikszentmihalyi [27], where users experience a continuous stream of enjoyable and creative activity for its own sake.

This mode of operation is very different to the conventional means of communication with a computer. Traditionally the software is there to gather data, and often does so by dominating the interaction. Even in those situations where the user is fully in charge of the interaction, it usually takes place at a certain level of language ability (for example, the need to read, interpret and take action on hierarchically arranged menus).

# 5. THE FUTURE OF MIDIGRID

It is a strange feeling to have produced an experimental instrument, moved on to other things, then to regularly hear about the 'new life' that the instrument has in other people's hands. So recently, once again, we have returned to the design table, and are reconsidering how MidiGrid should evolve in the 21st century. Two major new projects are planned, and are currently in their early stages.

## 5.1 MidiGrid on a PDA

Work is underway to produce a portable version of MidiGrid that will run on a Portable Digital Assistant (PDA). The ready availability of a touch-sensitive screen in a portable device would enable a miniature version of MidiGrid to be used on the move, and very easily in concert situations. We are currently experimenting with various devices and operating systems, and at the time of writing have a simple grid which responds to the touch-screen. An artist's impression of the final product can be seen below in Figure 4.

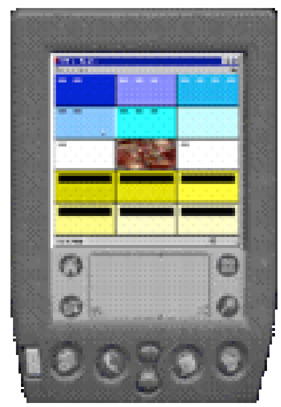


**Figure 4. Artist's impression of MidiGrid on a PDA**

## 5.2 MediaGrid

There is a natural progression for MidiGrid to be developed into a device capable of allowing live performance of pre-stored multiple media material [13]. The speed of today's PCs makes this perfectly feasible. What is interesting is to speculate on its uses. Imagine a touch-screen (or a mouse-control) on which there is a grid. As the grid is touched images appear on a screen, or soundfiles play, or movie snippets begin. Other boxes control the evolution or transformation of the material.

Though this project is in its infancy, we are excited by the possibilities of having a simple two-dimensional mapping of finger/hand position to animation clips, graphics, sounds, and the original MIDI-based notes, controllers and sequences.

MidiGrid continues to be used and developed, and has prompted a good deal of discussion and research into the role of mapping for live performance control [14][15].

# 6. REFERENCES


[1] "MIDIGRID - A new musical performance and composition system", A Hunt and PR Kirk Proceedings of the Institute of Acoustics 1988

[2] "MIDIGRID - An innovative computer-based performance and composition system", A Hunt & PR Kirk Proc. International Computer Music Conference 1990 pp127

[3] "MidiGrid - A computer-based Musical Instrument", Andy Hunt & Ross Kirk, Journal of the Institute of Musical Instrument Technology, Vol. 1 June 1994 pp 3-13

[4] Laurie Spiegel's "Music Mouse" software: http://retiary.org/ls/programs.html

[5] "The Role of Gesture in Environmental Control", Ross Kirk , Andy Hunt, Mark Hildred and Adrian Verity, Proc. EuroMicro Conference 2000, vol II, pp 377-381, Maastricht, September 2000.

[6] MidiCreator website: www.midicreator.com

[7] "Computer Music in the service of Music Therapy: The MIDIGRID and MIDICREATOR systems", M & R. Abbotson, PR Kirk, AD Hunt, A Cleaton, Medical Engineering Physics, Vol. 16, May 1994, pp. 253.

[8] "Enabling Musical Performance in Therapy: The MIDIGRID and MIDICREATOR systems", M & R. Abbotson, PR Kirk, AD Hunt, A Cleaton Proc. 11th International Congress World Federation of Occupational Therapists, April 1994.

[9] "Technology in the service of Music Therapy", M & R. Abbotson, PR Kirk & AD Hunt, World Congress of Music Therapy. July 1993

[10] "Music Therapy and Electronic Technology", Andy Hunt, Ross Kirk, Mary Abbotson and Raymond Abbotson, Proc. EuroMicro Conference 2000, vol II, pp 362-367, Maastricht, September 2000.

[11] "MIDIGRID - in the Mobile Music Therapy Unit", PR Kirk, AD Hunt, M & R Abbotson World Congress of Music Therapy, April 1991

[12] "Radical user interfaces for real-time control", Andy Hunt & Ross Kirk, Proc. EuroMicro Conference 1999, vol II, pp 6-12, Milan, September 1999.

[13] "Graphical Performance Interfaces for Computer Music Systems", Ross Kirk, Andy Hunt. EuroGraphics HCI conference April 1993 pp. 103

[14] Radical User Interfaces for real-time musical control, Thesis by Andy Hunt, University of York, 2000. Available on-line: http://www-users.york.ac.uk/~elec18/download/adh_thesis/

[15] "The importance of parameter mapping in electronic instrument design", Hunt, A.D., Paradis, M. & Wanderley, M., Proc. Conf. New Instruments for Musical Expression, Dublin, May 2002.

[16] "Virtual Musical Instruments: Accessing the Sound Synthesis Universe as a Performer", Axel Mulder. Available on-line at: www.cs.sfu.ca/~amulder/personal/vmi/BSCM1.rev.html

[17] "Electronic music interfaces: new ways to play", Joe Paradiso, IEEE Spectrum Magazine, Vol. 34, No. 12, Dec. 1997, pp. 18-30. Available on-line at: http://www.spectrum.ieee.org/select/1297/muse.html

[18] Fleximusic: http://www.fleximusic.com

[19] MousMuso: http://www.busker.net/mousmuso/mm1.html

[20] BlockJam: http://www.csl.sony.co.jp/IL/projects/blockjam/index.html

[21] Drake Music Project: www.drakemusicproject.com/

[22] Ansdell, G. "Music for Life – Aspects of Creative Music Therapy with adult clients", London 1995, ISBN 1-853-02299-3.

[23] Ellis, P. & Dowsett, R. (1987) MicroElectronics in Special Education British Journal of Music Education 4 (1), 17-23.

[24] Ellis, P. (1997) 'The Music of Sound: a new approach for children with severe and profound and multiple learning difficulties' (British Journal of Music Education, vol.14, no.2) 4.

[25] Fitzwilliam, A. (1988) An Assessment of the Benefits of Micro Technology in Music Therapy Journal of British Music Therapy 2 (1), 24-3.

[26] Laurie Spiegel, personal correspondence with the author.

[27] Csikszentmihalyi, M., "Beyond Boredom and Anxiety: Experiencing Flow in Work and Play (1975; reprint, Jossey Bass Wiley; 2000 ISBN: 0787951404).

# Bimanuality in Alternate Musical Instruments

Loïc Kessous
CNRS-LMA
31 Ch Joseph Aiguier
13402 Marseille
0033 491 16 42 10
kessous@lma.cnrs-mrs.fr

Daniel Arfib
CNRS-LMA
31 Ch Joseph Aiguier
13402 Marseille
0033 491 16 42 10
arfib@lma.cnrs-mrs.fr

## ABSTRACT
This paper presents a study of bimanual control applied to sound synthesis. This study deals with coordination, cooperation, and abilities of our hands in musical context. We describe examples of instruments made using subtractive synthesis, scanned synthesis in Max/MSP and commercial stand-alone software synthesizers via MIDI communication protocol. These instruments have been designed according to a multi-layer-mapping model, which provides modular design. They have been used in concerts and performance considerations are discussed too.

## Keywords
Gesture control, mapping, alternate controllers, musical instruments.

## 1. INTRODUCTION
Since the beginning of human evolution, the hands of man were always privileged tools of expression. Be it for survival, communication, or artistic creation, few other elements can claim to play such a determining part in the relations and the interactions between the man and its social or material environment. Naturally, other elements such as voice, posture, glance, and facial expression play an important role in communication and creative activity. Musicians and researchers have explored some of these aspects [11] [18]. Bimanuality has always played an important part in conventional acoustic instruments be it in a coordinated or co-operative way. All these constituents must deserve all our attention when it is a question of defining new interfaces in the artistic domain, made of nuances and sensibilities. The possibilities brought by sound synthesis and digital effects have already changed our way of conceiving musical composition. Today, the composer in many different music currents is accustomed to not only write melodies, rhythms and harmonies, but also timbres and spectral evolutions. Today technology allows to play, to interpret, and possibly to improvise the sound itself. New musical gestures require new instrumental gestures or at least the reorganization of them to allow a real interpretation. In this article we will first introduce previous studies on bimanual skills from Human Computer Interaction. Then we will consider the concept of musical instrument and the mapping strategy that we use to design our instruments using alternate controllers. Three examples will be detailed. Finally we shall consider the performance situation.

## 2. HUMAN SKILLED BIMANUAL ACTION
Bimanual human behavior has been the subject of some interesting studies. Some efforts have been done to borrow tools for design and evaluation from Human Computer Interaction [1] [15]. Let us point out some interesting studies concerning this subject.

### 2.1 Bimanual coordination
In [7] Y.Guiard describes what he calls the kinematic chain model, a general model of skilled bimanual action (i.e a serial linkage of abstract motors). The KC model hypothesizes that the left and right hands make up a functional kinematic chain. This leads to three general rules:

- *Preferred-to-non-preferred reference:* The preferred hand performs its motion relative to the frame of reference set by the non-prefered hand.
- *Asymetric scales:* The preferred and non-preferred hands are involved in asymetric temporal-spatial scales of motion. The movements of the non-preferred are low frequency compared to the detailed work done by the right hand. The preferred hand acts efficiently at microscopic scales and the non-preferred hand at macroscopic scales.
- *Preferred hand precedence*: The non-preferred hand precedes the preferred hand; for example left hand positions the paper, then the right hand begins to write (for a right-hander).

Interaction between the two hands is a fact. It could be considered as a constraint in designing instruments, a benefit (if one knows it) for making them more efficient (when hands act on different tasks of a musical process), or the starting point for the design of a musical interface.

### 2.2 Cooperative bimanual action
In cooperative bimanual action, the two hands combine their action to achieve a common goal. Cooperative bimanual action is particularly effective to manipulate objects. These can be real [8] or virtual objects [2]. This kind of gesture is very interesting for performing electroacoustic music pieces. In this prospect, the analogy between visual or material objects and sound objects (as described in Schaeffer's theory [16] ) is obvious. Using a multi-point tactile screen can be an interesting way to use cooperative movements.

## 3. MUSICAL INSTRUMENT
### 3.1 Acoustic musical instruments
The acoustic instruments have mechanical properties, which spontaneously offer to the performer force feedback, visual reference marks and multiple choices of selections. The force feedback and the visual and tactile reference marks allow an immersion in the playing and the regulation of the actions. Sense of touch, proprioception and visualisation provide different modalities of sensation. Multi-modality multiplies and diversifies sources of information, which allow a closer relation with the instrument.

### 3.2 Electronic musical instruments
#### *3.2.1 Generalities*
Generally, in electronic musical instruments, only the action of gestures is crucial to produce sound. Mechanical and

vibratory properties exist only if they are wished. Adding force feedback, visual feedback, visual and tactile reference marks must restore the epistemic function (the action of getting knowledge from the environment). Force feedback is not a standard programming feature on Macintosh, so we have concentred on visual and tactile reference marks which are easy to experiment (at very low cost). Some of our experiments seem to significantly improve the efficiency. On another side, electronic technology contributes to improve possibilities concerning the semiotic function (conveying information to the environment). Making the gestures symbolism understandable to computer is a good example of newly brought possibilities. In electronic musical instruments, the mapping plays a determining part because it makes possible to define the personality and the expressivity of the instrument. Measurement precision and interpretation of the ergotic function (the action of modifying and transforming the environment) are determininative points to allow the design of an efficient instrument.

### 3.2.2 Choice of alternate controllers

Existing controller can be classified as Instrument-like Controllers, Hybrid Controllers, or Alternate Controllers (see [19] [6] [12] [14] for examples and references). Our study is deliberately directed towards alternate controllers. What interests us is the possibility of defining an expressivity according to our wishes. Instrument-like controllers would impose in a large part the expressivity of their models. Hybrid controllers (acoustic instruments augmented by the addition of extra sensors) also do and require an expert control of the acoustic instruments. To start from more elementary data (position, pressure, flexion et cetera) makes it possible to conceive the expressivity on the level of the mapping without having to manage existing layers of mapping constrained by the physical configuration of the instrument. This approach seems to us most adequate to conceive the gestures corresponding to a sound and his evolution. A subdivision of Instrument-like Controllers is Instrument-inspired Controllers. This type of instruments is interesting in our musical context but one other reason to use Alternate Controllers relates to our interest for modular instruments. The postulate is to offer to the electronic performer instruments made up of couples of manual instruments. So the performer can choose the elements according to his needs.

## 3.3 Instrumental gesture typology

Various studies have defined different types of gesture. We use a typology close to the one defined in [3] and [20]:

- Excitation Gestures conveys the energy that will be found in the sonic result. It can be continuous, instantaneous or sustained. Static position that gives energy will be considered also as excitiation gesture.
- Modification Gestures modify the properties of the instrument but their energy do not participate directly in the sonic result. Such gestures can be divided into two groups: Parametric modification gesture - which continually changes a parameter (also called modulation gesture), and structural modification gesture - which modifies the structure of the object (instrument). Modulation gestures are more related to our study. Modifying a spectrum or applying vibrato will be considered as modulation gesture.
- Selection Gestures perform a choice among different but equivalent structures to be used during a performance. We will consider choice of musical note as a selection gesture.

# 4. MULTI-LAYER MAPPING CHAIN

Mapping can be used to define the personality and the expressivity of the instrument. It also allows defining an operating mode and a behavior in agreement with our sensori-motor system. We use three main steps between a gesture and a sound created by the synthesis model. The first mapping deals with the interpretation: it transforms gesture data into related-to-gesture-perception parameters, by means of a gesture extraction algorithm. This means that we transform measurements, quantitative data, into more qualitative information, closer to perception. By related-to-perception parameters, we mean parameters that make sense to our perception. This approach makes the implementation highly modular. This modularity is powerful to design instruments with a specific desired expressivity.

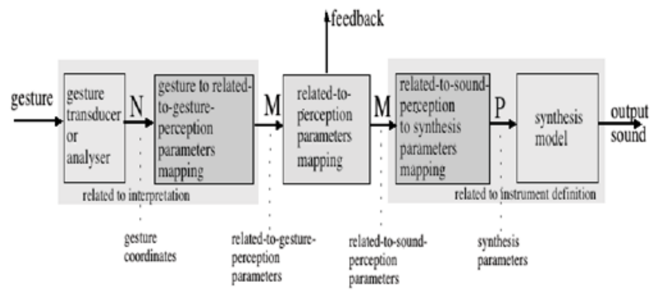



**Figure 1. Three-Layer-Mapping chain**

- In the first layer, gesture data are then transformed into related-to-gesture-perception data. Let's give an example take in literature. In [23], the parameters of three fingers are interpreted as a triangle (i.e. more significant parameters).
- The second layer transforms these data into related-to-sound-perception data. In [21], MIDI data from a wind controller is mapped to abstract parameters.
- Finally, a third mapping transforms related-to-sound-perception data into synthesis model data. In [22] pitch, loudness, and brightness are mapped to additive synthesis.

# 5. EXAMPLE 1 : THE VOICER

## 5.1 Description

The Voicer is a digital musical instrument producing vowel-like sounds with an expressivity depending upon the skills of the performer. It uses a bimanual control of subtractive synthesis with standard controllers. We have adapted a Wacom graphic tablet and a joystick to use them as an expressive instrument. Low-cost considerations, bimanual possibilities, and focus on making an efficient mapping have influenced the choice of two off-the-shelf controllers. Facilitating transmission to other (Mac addicted) musicians also encouraged us in this direction. This makes easier to evaluate the Voicer at expert level. It has been technically described in [10], so we shall shortly describe it here before going into bimanual considerations. The synthesis model consists of a sawtooth signal filtered by three second-order all-pole filters in cascade. This model can simulate a vowel singing voice. Here, we experiment with a mapping strategy that simultaneously allows melodic expressive control and vowel articulation.

## 5.2 Bimanuality in the Voicer

### *5.2.1 Preferred hand*

To permit control within one octave and from one to the other, we divide the tablet's active space into 12 angular sectors where each sector corresponds to a semitone of the chromatic scale, but glides between successive notes are also permitted. Turning clockwise changes pitch from low to high. We can go from a note to its lower or higher octave by clicking the stylus lateral button up or down or incrementally by making a whole turn. Sound doesn't exist if there is no pressure (or MIDI aftertouch-like damping). The preferred hand carries out circular gestures when there is a continuous excitation (pressure), and gestures in quasi-straight line if not. In a first step we thought about controlling the voiced/non-voiced balance by the radius on the tablet, but we didn't do it because of too many influences from the other hand on this gestural parameters. We do not give up the prospect using it in another way, but this use was not adapted. In fact the influence of a trajectory of one hand to the trajectory of the other is an interesting point.

### *5.2.2 Non-preferred hand*

The joystick controls the vowel articulation. X and Y data are mapped with perceptual attributes of sound described by Slawson [17]. In the case of vowel sounds, these attributes are closely related to tongue hump position and degree of constriction [5]. The non-preferred hand carries out gestures in straight line, but also, in an intuitive manner, circular arc. The non-preferred hand first selects the starting point of the articulation, then the other hand starts playing a note, then the non-preferred hand navigates in the vowel sound color space.

## 5.3 Mapping in the Voicer

### *5.3.1 First mapping layer*

X and Y coordinates on the tablet are first converted into polar coordinates. Then the 360° angle is divided into 12 angular sectors. Variation within each sector is extracted separately and a transfer function is applied to allow glissando and vibrato. This gives rise to a modulation gesture [10]. Choosing a sector is a selection gesture, moving inside the sector is a modulation gesture. Clicking down or up on the stylus button is a pre-selection gesture. Time interval between 2 consecutive defined pressure data gives us an instantaneous gesture as in a percussive instrument (like a MIDI note velocity). Maintaining or varying the pressure allows to maintain or vary the energy: it is considered as a continuous exciter gesture, so the pressure data is scaled and we apply another transfer function to it in order to get the right perception. X and Y data from the joystick are just clipped and scaled.

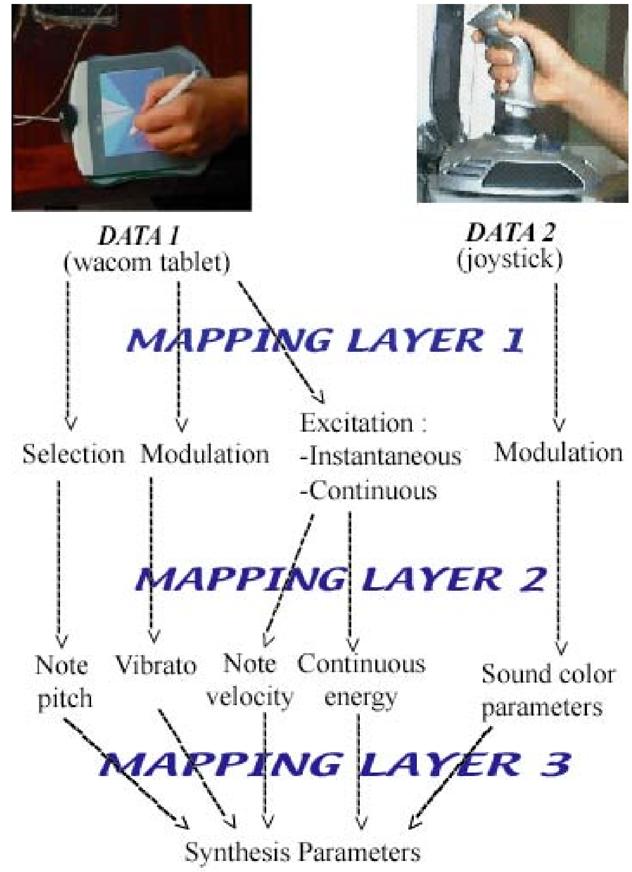



**Figure 2. Three-Layer-Mapping chain for the Voicer**

### *5.3.2 Second mapping layer*

Joystick modulation gestures (or navigation) are assigned to a position into the plane of a 2D interpolator. Moving into this plane can be considered as 2 combined modulation gestures or as a navigation gesture. Selection gesture (selecting an angular sector) is assigned to a defined pitch. Here we define the register of the instrument, the way in which it is tuned (for a guitar, open string pitches are E, A, D, G, B, E), and possibly the tonal scale (1/2 tone, 1/4 tone, just scale, et cetera). Tablet modulation gesture (relative movement into a sector) is mapped to vibrato (like a MIDI pitch bend). Obviously, the tablet instantaneous excitation gesture is mapped to note velocity. Continuous excitation is mapped to continuous energy (in MIDI standard it could be volume, breath controller or others). By defining the interpolator parameters, navigation is mapped to sound color.

### *5.3.3 Third mapping layer*

By using the interpolator, we interpolate the center frequencies of the formants (this mapping is clearly more conceptual), and sound colors are mapped to filter parameters (after conversion from frequency and radius). Note velocity and continuous energy are mapped to level and possibly (as we did in some experimentation) to parameters acting on source sound timbre (for example, when source sound is produced by Non Linear Distortion or FM). Note pitch and vibrato are mapped to fundamental frequency.

# 6. EXAMPLE 2 : SCANGLOVE

## 6.1 Description

The Scanglove is a bimanual instrument consisting of two different gloves equipped with sensors, controlling the parameters of a scanned synthesis [2]. In scanned synthesis, the shape of a simulated mechanic system is scanned at audio frequencies to produce sound. The scansynth~ object [4] generates a circular string (boundary conditions at the end of the string are transferred at the beginning) modeled with finite differences. We act on the initial shape and on the forces we apply. The synthesis meta-parameters that we have used are global damping, force gain and force extra-parameters (waveform "comb-like effect" suggested by Max Mathews and implemented by Jean-Michel Couturier). The non-preferred hand uses a 5dt Data glove [25]. The preferred hand uses a "home made" glove with FSR sensors (pressure and flexion) by way of an IcubeX (analog-to-midi converter).


**Figure 3. Using The Scanglove on stage during "Glovy dub"**

## 6.2 Bimanuality in the Scanglove

### *6.2.1 Preferred hand (home made glove)*

This hand uses an explicit mapping velocity with pressure sensors which detect instantaneous excitation to trigger a note. Pressure is also mapped to continuous excitation. Two pressure sensors are positioned on first and second phalanx of the index and the thumb act on it. The upper sensor is used to trig notes defined by the other hands in a defined octave, the lower acts on the octave below. Two others flex sensors act on continuous parameters. The first one is placed on the middle finger and the second one on the little finger. Movement of the 4th finger is too dependent of the others fingers to be used. Middle finger and little finger are relatively more independent.

### *6.2.2 Non-Preferred hand (5dt Data glove)*

For this hand we use the gesture's symbolism to do selection gesture. Using symbolism could be very helpful to drive complex harmonic structures while playing with the spectrum. A Multi Layer Perceptron (a Max external object available at [13], which can be trained in a real context) is used to map flex data from optic fibers of the 5dt glove to symbolic signs. The MLP external recognizes patterns of "Mimophony". The mimophony is a gestural code of a empty-hand symbolic sign representing a pitch note. It was recently used by a Contemporary Orchestra named "Allegro Barbaro" for conducting improvisation. From long time ago polyphonic singers from Corsica (a Mediterranean island near south of France) used it to communicate with each other while improvising.

## 6.3 Mapping in the Scanglove

### *6.3.1 First mapping layer*

The 5 flexions from the 5dt data glove are mapped to symbolic signs. The pressures on the different sensors from the "home made" glove are mapped to continuous and instantaneous excitation gestures and selection gesture (simultaneous selection and excitation gestures could be interpreted as a decision gesture). The 2 flexions from the "home made" glove are mapped to modulation gestures.

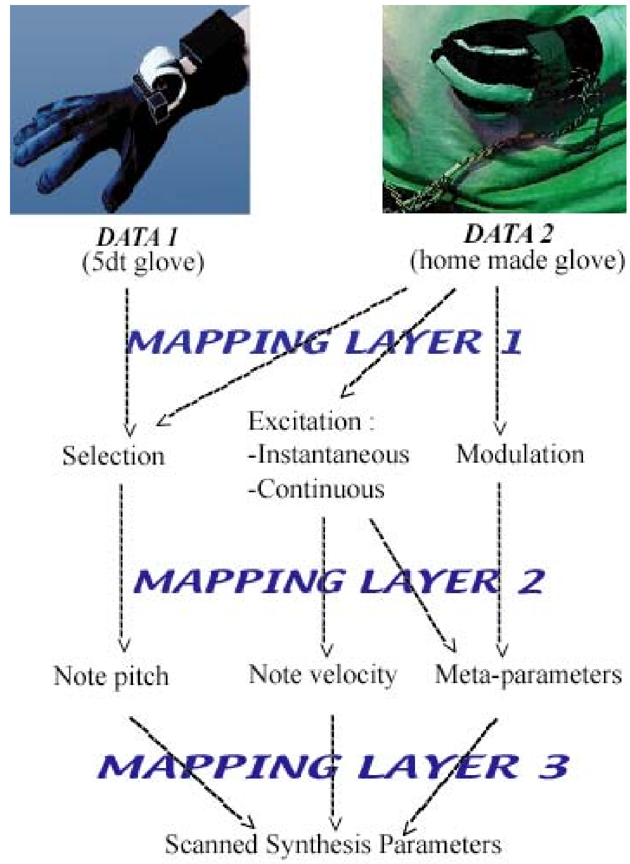



**Figure 4. Three-Layer-Mapping chain for the Scanglove**

### *6.3.2 Second mapping layer*

Selection of sign is interpreted as selection among the 12 semitones of the chromatic scale, and selection of upper or lower pressure sensors is interpreted as choosing the octave: this is mapped to note pitch. Instantaneous excitation gesture is mapped to note velocity and continuous excitation gesture is mapped to force gain meta-parameters. The first modulation gesture is mapped to global damping, and the second modulation gesture is mapped to force extra-parameter (waveform "comb-like effect").

### *6.3.3 Third mapping layer*

Note pitch is mapped to frequency, note velocity to initial shape properties, force gain and extra meta-parameters are mapped to low-level parameters.

# 7. EXAMPLE 3 : Voicer-like Controller

In this example we use the first two mapping layers of the Voicer with commercial stand-alone software synthesizers via the MIDI communication protocol (using IAC bus, or virtual MIDI port). Modularity provided by the three-layer-mapping chain makes it easy to use the expressivity (phrasing and articulation) of the Voicer to drive others synthesis models. In this case the last mapping layer can be done inside the software stand-alone synthesizer (most synthesizer are modular and can use GEN-like function). Some of them do not have strong-enough MIDI implementation to be driven. However, modulating pitch continuously (without triggering a new note) in a range of more than 24 semitones seems to be a real problem because the implementation is not available with enough precision. We have done this with the Voicer, going down to a sub audio frequency for which the glottal pulse is audible.

## 8. ON STAGE

We have used in concert bimanual instruments the design of which is based on the previously defined methodology. The band was composed of guitar, bass, drums, and a saxophone. Other electronic instruments (in particular, a wind controller mapped to a non-linear distortion synthesis model) were also used. The first challenge was to integrate a band and to adapt the instruments to the musical repertory. One does not have to permanently be in front of the computer to be able to communicate at any moment with the other musicians. Making gestures understandable to the audience is another challenge. Exaggerating gestures make them more comprehensible. A wireless technology could help in these two challenges, and probably more mechanical systems (which can also provide force feedback) could help for visual impact.


**Figure 5. On stage with the band**

## 9. CONCLUSION

We have shown in this article how to conceive, make and use digital musical instruments using a bimanual control. The principles exposed can help to design other instruments, and they give guidelines for their musical expressivity. The use in performance within a musical context illustrates the saying: “the proof of the cake is in the eating, not in the cooking”.

## 10. ACKNOWLEDGMENTS

Our thanks to Jean-Claude Risset for comments and suggestions about this paper. Our thanks to Jean-Michel Couturier who has implemented the scansynth~ Max/MSP external object.

## 11. REFERENCES

[1] Beaudouin-Lafon M., “Moins d’interfaces pour plus d’interaction”. In *Interfaces Homme-Machine et Creation Musicale*, Hermes Science Publishing, 1999.

[2] Buxton W. and Myers B., “A study in two-handed input”, In *Proceedings of the 1986 Conference on Human Factors in Computing Systems* ACM CHI’86, 321-326, 1986. http://www.billbuxton.com/2hands.html

[3] Cadoz C. and Wanderley M., “Gesture-Music”, in *Trends in Gestural Control of Music*, Wanderley M. and Battier M. eds, Ircam - Centre Pompidou, 2000.

[4] Couturier J.M., “A Scanned Synthesis virtual instrument”. *In proceedings of the 2002 Conference on New Instruments for Musical Expression* (NIME-02), pp. 176-178, Dublin, Ireland 2002.

http://www.mle.ie/nime/Proceedings/paper/couturier.pdf

[5] Flanagan, J.L., “Speech Analysis Synthesis and Perception”. Springer-Verlag, New York, 1965.

[6] Fels, S. and Hinton, G., “Glove-TalkII: A neural network interface which maps gestures to parallel formant speech synthesizer controls*”, IEEE Transactions on Neural Networks*, pp. 205-212, Vol 9, No. 1, 1998.

http://www.ece.ubc.ca/~ssfels/fpub.html

[7] Guiard, Y., “Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model”. *Journal of Motor Behavior* 19 (1987), 486-517, 1987.

[8] Hinckley, K., Pausch, R., Proffitt, D., Patten, J., Kassell, N., “Cooperative Bimanual Action”, In *Proceedings of the 1997 Conference on Human Factors in Computing Systems* ACM CHI’97, pp. 27-34, 1997.

http://research.microsoft.com/users/kenh/

[9] Hunt A., Wanderley M. and Kirk R., “Towards a Model for Instrumental Mapping in Expert Musical Interaction” *In proceedings of the 2000 International Computer Music Conference*, Berlin, Germany, 2000.

http://www.ircam.fr/equipes/analyse-synthese/wanderle/Gestes/Externe/Hunt_Towards.pdf

[10] Kessous L., “Bimanual mapping experimentation, with angular frequency control and sound color navigation”. *In proceedings of the 2002 Conference on New Instruments for Musical Expression* (NIME-02), pp. 174-175, Dublin, Ireland, 2002. http://www.mle.ie/nime/Proceedings/paper/kessous.pdf

[11] Lyons M. and Tetsutani N., “Facing the Music: A Facial Action Controlled Musical Interface”, In *Proceedings of the 2001 Conference on Human Factors in Computing Systems*, CHI’01 pp.309-310, Seattle, March 31 - April 5 2001.

http://www.mis.atr.co.jp/~mlyons/pub_pdf/Lyons_chi2001.pdf

[12] Merlier B., “A la conquête de l’espace”, *Journées d’Informatique Musicale*, La Londe Les Maures, France, 1998.

http://tc2.free.fr/Espace/

[13] MLP, (short for "multi-layer perceptron"): Max Artificial Neural Network Object that implements simple backpropagation learning and forward-pass.

Available at:

http://cnmat.cnmat.berkeley.edu/Max/neural-net.html

[14] Modler, P., “Neural Networks for Mapping Gestures to Sound Synthesis”. *In Trends in Gestural Control of Music*, M. Wanderley and M. Battier (eds), Ircam - Centre Pompidou, 2000.

[15] Orio N., Schnell N. and Wanderley M., "Input Devices for Musical Expression: Borrowing Tools from HCI" *Presented at the New Interfaces for Musical Expression Workshop* (NIME-01) during ACM CHI'01 - Seattle, USA, April 2001.

http://www.csl.sony.co.jp/person/poup/research/chi2000wshp/papers/orio.pdf

[16] Schaeffer P., "Traité des objets musicaux" Paris Seuil (eds), 1966.

[17] Slawson W., "Sound Color", Berkeley, University of California Press, 1985.

[18] Wanderley M., "Non-obvious Performer Gestures in Instrumental Music", in A. Braffort et al. (eds): *Gesture-Based Communication in Human-Computer Interaction*, Springer Verlag, 1999.

http://www.ircam.fr/equipes/analyse-synthese/wanderle/Gestes/Externe/GW99F.pdf

[19] Wanderley, M., "Performer-Instrument Interaction: Applications to Gestural Control of Music". PhD Thesis. Université Pierre et Marie Curie Paris VI, France, 2001.

http://www.ircam.fr/wanderle/Thesis/Thesis_comp.pdf

[20] Wanderley M., Viollet J-P, Isart F. and Rodet X., "On the Choice of Transducer Technologies for Specific Musical Functions". In *proceedings of the 2000 International Computer Music Conference*, Berlin, Germany, 2000.

http://www.ircam.fr/equipes/analyse-synthese/wanderle/Gestes/Externe/Wanderley_OntheChoiceFinal.pdf

[21] Wanderley M., Schnell N. & Rovan J., "Escher - Modeling and Performing Composed Instruments in Real-time", *IEEE Symposium on Systems, Man and Cybernetics* - SMC98 - San Diego, CA - USA, October, 14-17, 1998.

http://www.ircam.fr/equipes/analyse-synthese/wanderle/Gestes/Externe/SMC_FINN.pdf

[22] Wessel D., Drame C., and al. "Removing the Time Axis from Spectral Model Analysis-Based Additive Synthesis: Neural Networks versus Memory-Based Machine Learning", In *proceedings of the 1998 International Computer Music Conference*, Ann Arbor, Michigan, 1998.

http://cnmat.cnmat.berkeley.edu/ICMC98/papers-pdf/removing-time.pdf

[23] Wright M., Lee A., Momeni A., Freed A. "Managing Complexity with Explicit Mapping of Gestures to Sound Control with OSC", In *proceedings of the 2001 International Computer Music Conference*, Havana, Cuba, 2001.

http://cnmat.cnmat.berkeley.edu/ICMC2001/pdf/OSC-GestureMap.2.pdf

[24] Web site URL: 5dt glove. http://www.5dt.com/products/pdataglove5.html

# An Experimental Set of Hand Gestures for Expressive Control of Musical Parameters in Realtime

Paul Modler
Hochschule fuer Gestaltung
Lorenz Str. 15
76135 Karlsruhe
pmodler@hfg-karlsruhe.de

Tony Myatt
Music Department
University of York
Heslington, York
tone@cage.york.ac.uk

Michael Saup
Hochschule fuer Gestaltung
Lorenz Str. 15
76135 Karlsruhe
michael@particles.de

## ABSTRACT

This paper describes the implementation of Time Delay Neural Networks (TDNN) to recognize gestures from video images. Video sources are used because they are non-invasive and do not inhibit performer's physical movement or require specialist devices to be attached to the performer which experience has shown to be a significant problem that impacts musicians performance and can focus musical rehearsals and performances upon technical rather than musical concerns (Myatt 2003).
We describe a set of hand gestures learned by an artificial neural network to control musical parameters expressively in real time. The set is made up of different types of gestures in order to investigate:

- aspects of the recognition process
- expressive musical control
- schemes of parameter mapping
- generalization issues for an extended set for musical control

The learning procedure of the Neural Network is described which is based on variations by affine transformations of image sequences of the hand gestures.
The whole application including the gesture capturing is implemented in jMax to achieve real time conditions and easy integration into a musical environment to realize different mappings and routings of the control stream.
The system represents a practice-based research using actual music models like compositions and processes of composition which will follow the work described in the paper.

## Keywords

Gesture Recognition, Artificial Neural Network, Expressive Control, Real-time Interaction

## 1. INTRODUCTION

Discussions relating to gestural data processing for musical applications have emerged in recent years. This discussion developed from the background of interactive computer music systems and their use in performance, but also from the experience of novel interfaces to control the generation of sound and musical processes. Several data processing paradigms have been established inspired by the availability of a larger range of sensor systems and the increasing processing power These all use gestural data to control musical parameters (or light etc.) within artistic environments.

Issues of mapping control data to musical parameters are related to these paradigms and the detection of higher level expressive information of music parameters are of great interest (RIMM, MEGA, Wanderley/Battier).

A range of sensors and processing algorithms are available for specific applications, each with advantages and drawbacks according to the context of their use. This is especially true in dance and installation environments where video systems are often used to track movements or objects of interest (*Fingerprint, Palindrom, SoftVNS, BigEye*).

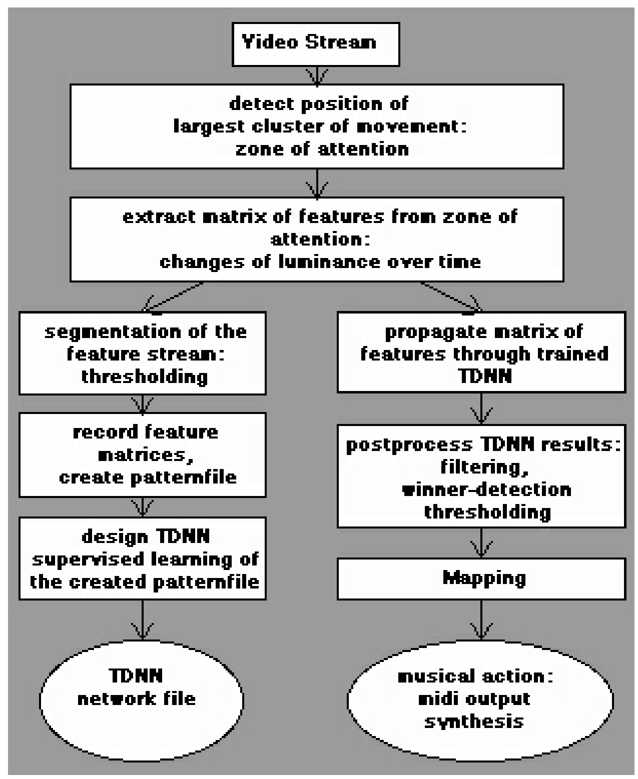



**Figure 1: Dataflow**

In this paper we describe a set of 17 gestures taught to a video based system developed for the recognition of gestural data in real-time

## 2. OVERVIEW OF THE GESTURE RECOGNITION PROCESS

The video stream from a standard dv-camera is analyzed and information relating to luminance magnitudes of consecutive video frames is extracted and presented to an artificial neural network.

The output of the neural network is evaluated using a post processing function that results in a binary output signifying the recognition of a trained gesture.

Video digitization and visualization, recording and editing of data as well as real-time gesture recognition is realized on a Pentium-4 Linux system running at 2.8 GHz with 25 Frames per seconds video resolution and running *jMax* 2.5.1. A standard consumer video camera as well as a low cost web-cam, had been used successfully.

A mapping patch combines the output of the neural network with continuous parameters derived from the gesture. Different setups of the mapping can be switched through the recognition process itself.

## 3. EXPERIMENTAL SET OF HAND GESTURES

Based on previous work demonstrating successfull recognition of a small set of four hand gestures we choose a larger set of hand gestures to be taught and recognised. The set is experimental, in such a way, that it is incomplete and does not represent an entire hirarchy or typography of gestures. But it was composed out of gestures representing different types like full handed or single finger oriented or functional like pointing or abstract like moving the hand in a certain way.

The gesture set is recorded two times on a consumer dv camera. One set is the trainings set and one set the test set. For each set every gesture is recorded 2 times at 3 different speeds: slow, medium, fast.

The training-set is then digitized using a *jMax* patch described below. For each class of gestures a *pattern-file* is created in the highest resolution. This gives the possibility to compose in a later stage various trainings pattern-sets in different resolutions or integrating later new recorded gestures.

The patterns are stored in the pattern-file format of the Stuttgarter Neuronale Netz Simulator.

**Table 1: Experimental Gesture Set**

| | **Gestures** | **Abbr.** |
|---|---|---|
| 1 | Flat figure 8 | fig8 |
| 2 | index moving with hand right | wa-ri |
| 3 | index moving with hand left | wa-le |
| 4 | pointing down | in-do |
| 5 | pointing up | in-up |
| 6 | pointing left | in-le |
| 7 | pointing right | in-re |
| 8 | Index & middle moving “walking” (fingers down) | walk |
| 9 | index rotating in a circle | ro-ci |
| 10 | waving hand , finger down | wa-do |
| 11 | musical stop (grasping, circle) | mu-st |
| 12 | hand opening, hand horizontal, fingers to front | hh-op |
| 13 | hand closing, hand horizontal, fingers to front | hh-cl |
| 14 | hand opening, fingers up | open |
| 15 | hand closing, fingers up | close |
| 16 | waving, fingers up | waving |
| 17 | flatter, all fingers move randomly except thumb | flatter |

## 4. PREPROCESSING AND FEATURE EXTRACTION

According to the layout of the algorithm in figure 1 the incoming video stream is searched to find the largest cluster of luminance variations in consecutive video frames.

For this a clustering process is used to detect the overall location of the gesture. The aim of this is to achieve a position independent sub-frame and also to increase the resolution of the relevant video data only for the relevant parts of the visual stream. In other words: the whole picture is inspected to indicate a small zone for high resolution data capture (zone of attention).

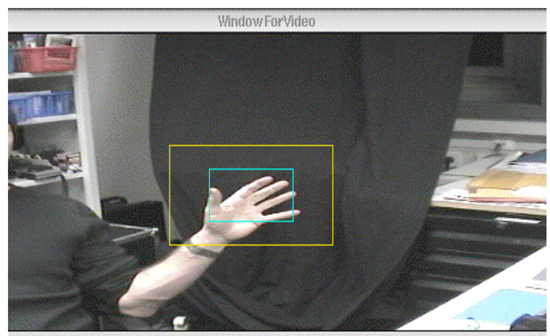



**Figure 2: Video Input and Zone of Attention**

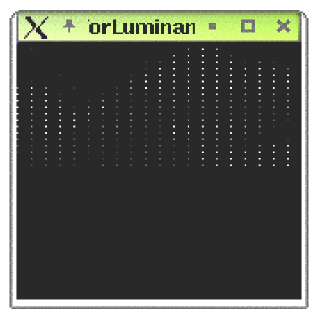



**Figure 3: Feature Extraction**

In our case the whole video frame is searched to find the location where the gesture of the hand is produced. A certain area (zone of attention) is then cut out and used to extract the feature-map for the neural network. The coordinates of the zone of attention are also extracted and can be used as subgestural information to control additional parameters.

## 5. VARIATIONS OF PATTERNS BY AFFINE TRANSFORMATIONS

Recording and editing of gestures is time consuming. We applied linear affine transformations on each recorded gesture in the following order to multiply the recorded training sets for increased stability and quality of the recognition:

- *stretching* in the x, y direction
- *rotation* in the x, y plane around the center of gravity of the gesture

- *shifting* in the x, y plane

Each of this transformation operations are realized as an appropriate method in *jMax* to compute the transformation.

## 6. DESIGN OF THE ARTIFICIAL NEURAL NETWORK: TDNN

For the recognition of the gestures we use a Time Delay Neural Network architecture. The Time Delay structure was developed for phoneme recognition (Waibel 1989, Berthold 1994) but has been successfully applied in gestural processing (Modler, Zannos 1997) as well as in musical audio applications (Marolt 1999). A certain form of Time Delay Networks was succesfully applied for recognition of image sequences for gestural control (Vassilakis, Howell, 2001)

This neural network architecture provides recognition of timed patterns at low processing power requirements independent from the pattern speed reference.

The network for the gesture set we designed to have one input layer with 900x6 input units, one hidden layer with 50x4 units and one output layer with 17 units. For each frame of the video stream the network is presented a new set of pixels plus the previous 5 sets. This can be seen as a windowing function over the whole data stream. (Zell, 1994)

The TDNN is design and trained using the Stuttagrter Neuronale Netz Simulator (SNNS) with the pattern-sets created as described above.

The time needed to teach the TDNN varies depending on the size of the network, the size of each pattern and the number of instances for each gesture type, and on the number of cycles a pattern set has to be taught before the pattern set is learnt sufficiently.

One Cycle of a whole pattern- set for the set of 17 gestures needs about 20 hours to be taught on a Pentium 4, 2.2GHz. A pattern-set we are using for training the network contains about 17 times approximately 1300 patterns giving a rough total sum of 22100 patterns.

## 7. POST-PROCESSING

The output of the neural network is processed with threshold and filter functions.

Together with the overall level of the energy of the hand gesture the onset and offset of a gesture as well as the type of the gesture is estimated.

The output of the recognition process can be displayed on the screen as well as be sent to external devices via Midi or it can be internally used to control routing, mapping or sound parameters.

## 8. RECOGNITION RATES

Various parameters of the hand gestures, like distance, location, rotation and size of the hand in the video frame have influence on the recognition results.

Overall light conditions, gesture speed and the number of taught gestures etc, also have an impact on recognition rates.

Figure 4 shows the jMAx patch displaying the resulting values of the 17 output neurons and the energy of the gesture and the index of the winner neuron as time diagrams. On the right side a column with the abbreviations of the gestures names can be seen. On the left side the actual values are displayed. In the shown diagrams three instances of the gesture "waving" are presented and successfully recognized. In the middle part of the patch the energy of the gesture is displayed and below the index of the winner.

Smaller excitation peaks of non valid neurons, like "flatter", "close" or "mus-st" are not strong enough to disturb the correct recognition.

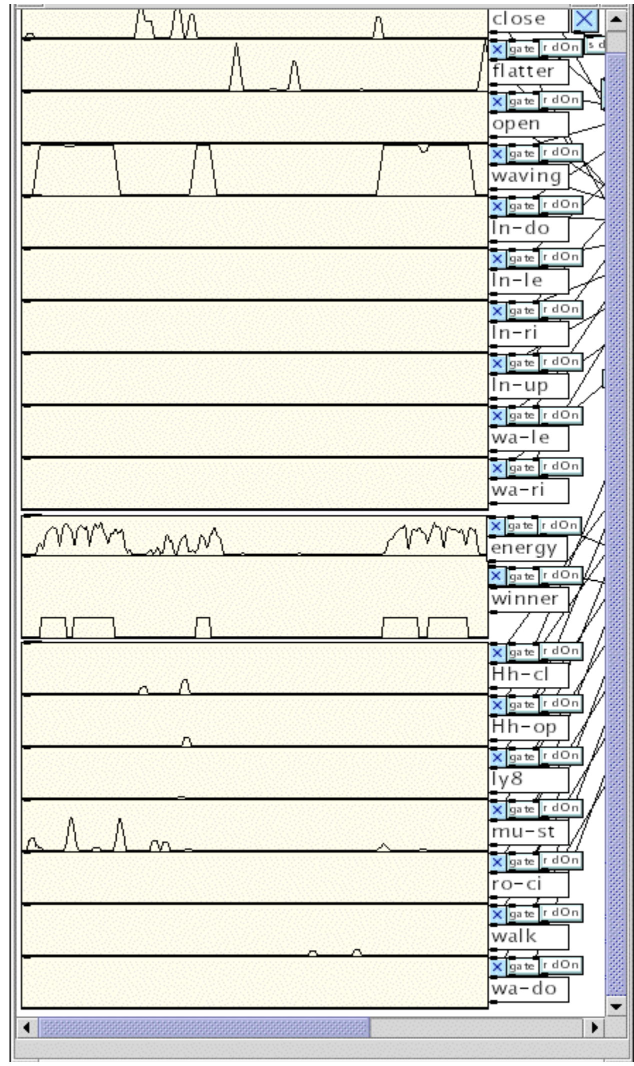



**Figure 4: Results of the Output Neurons, Energy and Index of Winner over Time for 17 Gestures**

An output value from the neuron close to 1.0 (upper line) indicates the network assuming the output unit as recognized.

For the estimation of recognition rates we use different means. First we process the original gestures played back from the dv camera and directly processed through the system. This gives a recognition rate of about 100%.

Then the recorded test set is processed. It contains also 17 gestures recorded 2 times at 3 different speeds. The test-set is recognized at a rates of about 93 %.

As a third test we feed directly the gestures from a live performer. The performer chooses a set of 30 gestures at random and presents them to the system through the dv camera. The outcome varies depending on how the gestures are presented to the network, and what gestures are presented. At this stage 80% recognition rates have been demonstrated.

We envisage improvements through future developments of the algorithm and data gathering and feature extraction process. This order of recognition rate may be sufficient for free structured musical works like improvised and semi-improvised music which will be determined through future practice-based research. The communication of meaningful performance data to an interactive computer system is of such significance that musical benefits from this techniques are likely without increased recognition rates.

Testing a selected subset of the gestures for example a set of open handed gestures like "waving,, flatter, stop, flat-figure-8, waving-down" rises the recognition rate up to about 90%.


**Figure 5: Motion-Energy, Winner-Index, Output of Index-Up and Index-Down Neurons over Time**

## 9. ROBUSTNESS

### *Camera distance*

Adjusting the video frame through the cameras zoom function the distance of the camera to the performer can be increased for the recognition phase.

For a network trained with gestural data recorded at a distance of 1.6m we achieved a loss in recognition of about 0%-10% increasing the cameras distance to up to 8m.

### *Light Intensity and Light Direction*

The system is robust against light intensities different from light intensities of the training data. We had no significant loss in recognition rates when reducing the light-intensity from 840 Lux (training data) to about 180 Lux (recognition data) which is equivalent to a focal reduction of 2.5 steps.

The system is sensible against different light directions. since they produce different overall shapes of the extracted features of the hand.

## 10. RECOGNITION DELAY

A crucial question is the processing time of a recognition process especially in a musical real-time environment. Due to the structure of the Time Delay Neural Network gestural data is presented continuously to the network. This can be seen as the recognition process can start at any point in the data-stream. It is possible to recognize the gesture in before the gesture is finished or before it is "felt" to be finished.

In figure 5 the output of the motion energy of the attention rectangle, the index of the estimated gesture (winner), motion parameters of the attention rectangle (arVX, arVY, energyAR) and the output of the Neurons for the gestures "Index-Up" and "Index Down" are shown over time.

The diagram shows that the gestures are recognized before the gestures are fully completed. The maximal motion energy of the gestures are corresponding with the recognition results.

## 11. INTEGRATION INTO A PERFORMANCE ENVIRONMENT

The whole system is integrated into the jMax environment For that we realized following objects as external objects in jMax: (Modler, 2002):

**Table 4: External jMax Objects**

| Object | **Purpose** |
|---|---|
| grabber | video input |
| window | video ouput and data visualisation: |
| recorder | recording editing and saving of multidimensional data |
| feature | feature extraction |
| nn | TDNN and patternfiles, gesture recognition |

The object realizing the Time Delay Neural Network is based on the kernel of the Stuttagarter Neuronale Netz Simulator (SNNS, 1994). It is integrated into the jMax environment.

The results of the recognition process as well as values of the extracted features can be sent to external devices through the standard *jMax* midi-port.

In a test setup we trigger different audio samples in jMax from the output of the gesture recognition process . Additional parameters, like the position of the Zone of Attention or the volume of the luminance inside the Zone can be sent via midi or ethernet as sub-gestural control information to remote units. Depending on the load of the processor audio synthesis can be triggered and controlled on the same machine.

## 12. MAPPING STRATEGIES FOR EXPRESSIVE CONTROL OF PARAMETERS

To control musical performances we took 3 schemes for connecting the output of the neuronal network to musical parameters into account.

### *Triggering - Symbolic*

For each trained gesture an audio file is associated. If the recognition algorithm results a valid winner, the appropriate soundfile is played. For this a larger set is desirable, since a restricted set of gestures reduces the possibility of variations.

### *Triggering & Continuous Control - Symbolic & Parametric*

A combination of symbolic commands derived from the recognition process and additional continuous parameters are used. A selection situation is a paradigm for such a mapping: For example the x coordinate of the hand location in the video image is mapped to the index of the sound selection.

The gesture "pointing down" selects the sound and "open" plays it back with a volume proportional to the energy of the "open" gesture. The latter mapping is used for expressive control of the onset of the sound. Additional sensor data gathered from sensor devices like accelerometers, can be integrated into the recognition and mapping process.

### *Direct mapping - Parametric*

Since the results of the output neurons is a continuous stream of floating point values we directly can map this stream to a set of continuous parameters. For this we connect them to the volume controls of a set of sine wave generators. Each output neuron then controls the loudness of a sine wave generator.

## 13. CONCLUSIONS

Our aim was to investigate the use of a small experimental set of hand gestures in combination with a video and neural network based gesture recognition system.

Gestures for the set are chooen according to the scheme to combine gestures of different types.

To increase the robustness of the recognition results and to reduce the necessary number of gesture recordings we successfully used affine transformations on the training-set for creating variations to extend the set of recorded gesture patterns. Through a set of external objects for jMax we provide all facilities to record, edit and generate the desired patternfiles.

For the recognition we designed a Time Delay Neural Network architecture and trained it with the generated patterns successfully. The trained Network is loaded into the jMax environment and used successfully for the gesture recognition process in real-time. The recognition rates differ from about 90% for the laboratory training-set to about 80% for the real performing situation with live camera input.

We showed that the dictionary of hand gestures can be used in different types of mappings to control musical parameters. The combination of symbolic triggering of musical events with the parametric use of continuous sub-gestural data like the location of the hand in the x,y image or the energy of the hand gesture offer expressive musical control.

Further work in this area will provide details about the maximum number of different gestures which can be learned sufficiently from a chosen network design and about the use of different gesture types, such as hand gestures or full body gestures. Also more detailed setups for the musical mapping have to be investigated.

The system provides a promising environment for experimental setups using gesture recognition for expressive control of musical parameters.

## 14. REFERENCES

[1] Berthold, R. Michael.: A Time Delay radial basis function network for phoneme recognition. In Proceedings of IEEE International Conference on Neural Networks, volume 7, pages 4470-- 4473, Orlando, FL, 1994. IEEE Computer Society Press.

[2] de Cecco, M., Dechelle, F., jMax/FTS Documentation, http://www.ircam.fr, 1999

[3] Harling, P.A., Edwards, A.D.N., (Eds), Proceedings of Gesture Workshop'96, Springer-Verlag (Pub.), 1997.

[4] Hofmann, F.G, Hommel, G.: Analyzing Human Gestural Motions Using Acceleration Sensors., Proc. of the Gesture Workshop '96 (GW'96), University of York, UK

[5] Marolt, Matia, A Comparison of feed forward neural network architectures for piano music transcription, Proceedings of the ICMC 1999, ICMA 1999

[6] MEGA, Multisensory Expressive Gesture Applications, V Framework Programme IST Project No.1999-20410, 2002, http://www.megaproject.org/

[7] Modler, Paul, Zannos, Ioannis, Emotional Aspects of Gesture Recognition by Neural Networks, using dedicated Input Devices, in Antonio Camurri (ed.) Proc. of KANSEI The Technology of Emotion, AIMI International Workshop, Universita Genova, Genova 1997

[8] Modler, Paul, A General Purpose Open Source Artificial Neural Network Simulator for jMax, IRCAM-Forum, Nov. 2002, Paris

[9] Myatt, A: Strategies for interaction in *construction 3*, Organised Sound, Volume 7 Number 3, CUP, Cambridge UK 2002, pp157-169

[10] Palindrome, http://www.palindrome.de/

[11] The RIMM Project, Real-time Interactive Multiple Media Content Generation Using High Performance Computing and Multi-Parametric Human-Computer Interfaces, European Commission 5th Framework programme Information, Societies, Technology 2002, http://www.york.ac.uk/res/rimm/

[12] Rokeby, David, SoftVNS Motion Tracking system, http://www.interlog.com/~drokeby/softVNS.html

[13] SNNS, Stuttgarter Neural Network Simulator, User Manual 4.1, Stuttgart, University of Stuttgart, 1995.

[14] Vassilakis H.,Howell, J. A., Buxton, H. I, Comparison of Feedforward (TDRBF) and Generative (TDRGBN) Network for Gesture Based Control, Proceedings of the Int. Gesture Workshop 2001,

[15] Wachsmuth, I., Froehlich, M. (Eds), Proceedings of the Int. Gesture Workshop'97, Lecture Notes in Artificial Intelligence, Springer-Verlag (Pub.), 1998.

[16] Waibel, A., T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang, Phoneme recognition using time-delay neural networks, IEEE Transactions On Acoustics, Speech, and Signal Processing, Vol 37, No. 3, pp. 328-339, March 1989.

[17] Wanderley, Marcelo, Battier Marc, Trends in Gestural Control of Music, CD-Rom, 2000, IRCAM, Paris,

[18] Zell, Andres, Simulation Neuronaler Netze, Bonn, Paris: Addison Wesley, 1994.

# *Immersion Music*: a Progress Report

Teresa Marrin Nakra
Artistic Director, Immersion Music, Inc.
P.O. Box 230248
Boston, MA 02123
1-617-686-4898
teresa@immersionmusic.org

## ABSTRACT

This paper describes the artistic projects undertaken at Immersion Music, Inc. (www.immersionmusic.org) during its three-year existence. We detail work in interactive performance systems, computer-based training systems, and concert production.

## Keywords

Interactive computer music systems, gestural interaction, Conductor's Jacket, Digital Baton

## 1. INTRODUCTION

Since January of 2000, I have been actively developing a new line of work with a nonprofit startup called Immersion Music. Our mission is to bring the high-tech interaction paradigm to classical and traditional musical performances. This report describes the projects that we have been working on for the past three years and the general themes and directions that have emerged during this time.

## 2. INITIAL DEVELOPMENT

While a graduate student at the MIT Media Lab, June 1994-October 1999, I developed a quantitative method of research into the phenomena of orchestral conducting. First with a device called the "Digital Baton"[11][12][13], and then with device called the "Conductor's Jacket"[5][7][8][9][10], I developed ways to capture the gestures and indications of professional musical conductors. Professors Tod Machover and Rosalind Picard guided my work and significantly influenced my thought processes during this time.

The technical basis of this work has been to analyze and leverage the gestures that musical performers make while practicing their craft and technique. While at the MIT Media Lab I also developed many performance applications for gestural interfaces. Ongoing research informed the performances, and a groundbreaking event was presented at a concert of the Boston Pops Orchestra in June of 1998, where my team presented a demonstration of interactive technology with conductor Keith Lockhart.

My experiences at the Media Lab inspired me to set out to create a structure within which I could continue both the research and artistic applications of my work. Immersion Music was created as a not-for-profit organization in 2000 to realize that purpose.

## 3. PROJECTS 2000-2003

The types of work that we have done at Immersion Music include interactive performance systems, computer-based training systems, and concert production. We have developed six different performance systems during the past three years, including solo, duo, and concerto formats. We have also assisted with the development of three different visual accompaniment systems for performers, including an interactive video system, an interactive lighting system, and a system for interactive animation.

### 3.1 Music Systems for Gesture Performers

Among the solo music systems that we have developed is a software environment for the Conductor's Jacket interface that runs Manfred Clynes' "Superconductor" software in real-time. We frequently present this system in performance in a "solo conductor" format with the Bach Brandenburg Concerto. Our first duet for gesture performers was developed by Lia Davis at Harvard University, where it was premiered during an Immersion Music residency in February 2002. Our largest performance format to-date has been a "Concerto for Conductor"[2], which we initiated and co-commissioned. The piece was written by composer John Oswald and presented by both the Boston Modern Orchestra Project at the Boston Cyberarts Festival (May 2001) and the American Composers Orchestra at the Orchestra Tech Festival in New York (October 2001), with generous funding from the LEF Foundation and the Canada Council for the Arts.

### 3.2 Gesturally-controlled visuals

We have also developed several performance pieces that are accompanied by responsive visuals. The first was a gesture performance system with interactive video processing and control. This was developed with Walter Wright, the inventor of the music video format and an active real-time video artist. Our piece, "SP/RING," developed in May 2000, uses the "solo conductor" performance format. The performer uses conducting-style gestures to simultaneously control both interactive music and video systems.

In addition, we have worked with the abstract elements of theatrical lighting instruments to develop an interactive lighting system. In May 2001 we premiered a new system in collaboration with lighting designer Herrick Goldman at the Boston Cyberarts Festival. The performer was violinist Joanna Kurkowicz, playing a sonata by Alfred Schnittke. The piece had no electronic music aside from some minor amplification, but since Joanna plays on a 1699 Guarnerius violin, the sensing of her gestures could not be done by means of instrumenting the violin. Therefore, using the Conductor's Jacket sensors was a good solution to allow her performance gestures to influence and control the aspects of a complex lighting design. We further advanced the lighting system in February 2002 with lighting designer Sarah Sidman during the Immersion Music residency at Harvard University. Using solo violin works by Bach and Paganini as the basis for the performance, we built several different responsive visual environments for the performing violinist.

In November 2002, we extended our visual repertoire to computer animation. Working with visual artists Dong-Keun Jang and Jan Kubasiewicz, we helped to develop a visual response to “Fratres” by Arvo Paart. The work was presented as part of the Massaging Media conference.

## 3.3 Computer-based Music Training Systems

In collaboration with Professor Gary Hill and the Music Department at Arizona State University, we have designed and developed a Digital Conducting Laboratory. The system has enhanced the curriculum for more than fifty undergraduate music students during the past three years, and has also served as a platform for a few graduate research projects. This work has been described elsewhere in [6] and also at www.immersionmusic.org. Future work in this area is of increasing interest for the organization, and we see a great deal of scope for interesting new projects in computer-based interactive musical training.

## 3.4 Concert Production

It should also be mentioned that in addition to our technical and artistic activities, Immersion Music has been deeply involved in the production of several events in the Boston area during the past few years. For some time we ran a series called the “Immersion Music Salon” at a local art gallery, which brought together many local improvising artists using technology in interesting ways. We have also presented events at the Boston Cyberarts Festival and the Boston First Night Festival.

Our largest undertaking to-date has been the joint presentation of “Orchestral Music at the Technological Frontier,” an event at Boston’s Symphony Hall in May 2001. We provided nearly all the technical support for a 3-hour concert of technologically-enhanced orchestral music, co-presented with the Boston Modern Orchestra Project. This concert attracted nearly 2000 people, which is quite a large number for a concert of new music. Ellen Pfeifer of the Boston Globe praised our efforts, writing: “the free concert drew a large audience that appreciated innovation. Young and cool, middle-aged and hip, plugged-in, free-spirited, garbed in the many varieties of geek chic and la mode Bohemien, the demographic was the sort many conventional music organizations would kill for.”

Our forays into concert production have been an integral part of our mission, and will continue to be important in our future work. Interaction with the public gives us much-needed feedback and helps us understand where the work can be improved.

## 4. ACKNOWLEDGEMENTS

I would like to acknowledge the incredible work of the members of the Board, Advisory Board, and volunteers at Immersion Music, all of whom have given generously of their time and talents to help realize a special vision together. We also are deeply indebted to the fantastic artists we’ve had the pleasure to collaborate with, especially the Boston Modern Orchestra Project and its director Gil Rose. Finally, none of this work would have been possible without the careful and caring guidance of my graduate advisors Tod Machover and Roz Picard of the MIT Media Lab, as well as my doctoral committee members John Harbison and David Wessel.

## 5. REFERENCES

[1] Marrin Nakra, T. (2002). Musical Performance Applications of Affective Computing. Invited paper, Computer Human Interaction, April 2002.

[2] Marrin Nakra, T. (2001). The “Concerto for Conductor”: a new musical form. Invited paper, Acoustical Society of America, December 2001.

[3] Marrin Nakra, T. (2001). Translating Conductors’ Gestures to Sound. Invited paper, Human Supervision and Control in Engineering and Music, September 2001.

[4] Marrin Nakra, T. (2001). Synthesizing Expressive Music through the Language of Conducting. Journal of New Music Research, June 2001.

[5] Marrin Nakra, T. (2000). Inside the Conductor's Jacket: Analysis, Interpretation and Musical Synthesis of Expressive Gesture. Ph.D. Thesis, Media Laboratory. Cambridge, MA, Massachusetts Institute of Technology. http://www.media.mit.edu/~marrin/HTMLThesis/Dissertation.htm

[6] Marrin Nakra, T and Gary Hill. (2000). Quantitative Practice for Students at the Digital Conducting Laboratory. Invited paper, Acoustical Society of America, December 2000.

[7] Marrin Nakra, T. (1999). Searching for Meaning in Gestural Data: Interpretive Feature Extraction and Signal Processing for Affective and Expressive Content. Trends in Gestural Control of Music. Wanderley, M. and Rovan, B., eds., Paris, IRCAM.

[8] Marrin Nakra, T. (1999). Incorporating gestures into the musical control stream with the Conductor's Jacket. Journal of the Acoustical Society of America, vol. 106, no. 4, part 2. Awarded Best Student Paper Award in Musical Acoustics at the meeting of the ASA in Columbus, Ohio, November 1-5 1999.

[9] Marrin, T. and R. Picard. (1998). The Conductor's Jacket: a Device for Recording Expressive Musical Gestures. International Computer Music Conference, Ann Arbor, MI, pages 215-219. http://www.media.mit.edu/~marrin/ICMC98.htm

[10] Marrin, T. and R. Picard. (1998). Analysis of Affective Musical Expression with the Conductor's Jacket. XII Colloquium for Musical Informatics, Gorizia, Italy, pages 61-64. http://www.media.mit.edu:80/~marrin/CIM.htm

[11] Marrin, T. and J. Paradiso. (1997). The Digital Baton: a Versatile Performance Instrument. International Computer Music Conference, Thessaloniki, Greece, pages 313-316. http://www.media.mit.edu/~marrin/ICMC97.htm

[12] Marrin, T. (1997). Possibilities for the Digital Baton as a General-Purpose Gestural Interface. CHI '97 Conference on Human Factors in Computing Systems, pages 311-312. http://www1.acm.org:82/sigs/sigchi/chi97/proceedings/short-talk/tm.htm

[13] Marrin, T. (1996). Toward an Understanding of Musical Gesture: Mapping Expressive Intention with the Digital Baton. M.S. Thesis, Media Laboratory. Cambridge, MA, Massachusetts Institute of Technology. http://www.media.mit.edu/~marrin/Thesis.htm.

# OpenSound Control: State of the Art 2003

Matthew Wright
Center for New Music and Audio Technology (CNMAT)
Univ. California, Berkeley
1750 Arch St., Berkeley, CA 94709
(1) 510 643-9990
matt@cnmat.berkeley.edu

Adrian Freed
CNMAT
Univ. California, Berkeley
1750 Arch St., Berkeley, CA 94709
(1) 510 643-9990
adrian@cnmat.berkeley.edu

Ali Momeni
CNMAT
Univ. California, Berkeley
1750 Arch St., Berkeley, CA 94709
(1) 510 643-9990
ali@cnmat.berkeley.edu

## ABSTRACT
OpenSound Control ("OSC") is a protocol for communication among computers, sound synthesizers, and other multimedia devices that is optimized for modern networking technology. OSC has achieved wide use in the field of computer-based new interfaces for musical expression for wide-area and local-area networked distributed music systems, inter-process communication, and even within a single application.

## Keywords
OpenSound Control, Networking, client/server communication

## 1. TUTORIAL OVERVIEW of OSC
This is a user-level overview of OSC. For more technical details such as exact semantics and the binary format of OSC messages, please see the OSC specification [33].

### 1.1 Client/Server
OSC is designed to support a client/server architecture. OSC data is transmitted in data units called *packets*. Anything that sends OSC packets (e.g., an application, physical device, subprogram, etc.) is a *client* and anything that receives OSC packets is a *server*.

OSC is a transport-independent, high-level application protocol; in other words, OSC does not specify what low-level networking mechanism will be used to move OSC packets from the client to the server.

### 1.2 Messages
The basic unit of OSC data is a *message*, consisting of an *address pattern,* a *type tag string,* and *arguments*. The address pattern is a string that specifies the entity or entities within the OSC server to which the message is directed (within the "Addressing Scheme" described below) as well as what kind of message it is. The type tag string gives the data type of each argument. The arguments are the data contained in the message.

For example, a message's address pattern might be /voice/3/freq, its type string might indicate that there is a single floating-point argument, and the argument might be 261.62558.

### 1.3 Argument Data Types
Each message contains a sequence of zero or more arguments. The official OSC data types are ASCII strings, 32-bit floating point and integer numbers, and "blobs," chunks of arbitrary binary data. OSC's type mechanism allows for many other types, including 64-bit numbers, RGBA color, "True," and "False." Only a few implementations support these other types, but they all represent them in a standard way.

### 1.4 Addressing Scheme
All of the points of control of an OSC server are organized into a tree-structured hierarchy called the server's *address space*. Each node of the address space has a symbolic name and is a potential destination for OSC messages.

Each OSC server defines its own address space according to the features it provides and the implementor's idea of how these features should be organized. This is in contrast to protocols such as ZIPI [22] and MIDI that attempt to define in advance what the architecture of a synthesizer should be and what kinds of messages can be sent to it.

An OSC *address* is simply the full path from the root of the address space tree to a particular node, with a slash-delimited format like a URL or file system pathname. For example, the address "/voices/3/freq" refers to a node named "freq" that is a child of a node named "3" that is a child of a top-level node named "voices."

An OSC server's address space may change dynamically, therefore OSC's query system (described below) includes a mechanism for discovering the current address space.

### 1.5 Address Pattern Matching
Remember that each OSC message contains an OSC address *pattern*, not an OSC address. An OSC address pattern is exactly like an OSC address, except that it may contain special characters for regular expression [10] pattern matching. When an message's address pattern matches more than one of the addresses in the server's address space, the effect is the same as if there were individual messages (all with the same arguments) sent to each of the matched addresses.

The special characters are '?', '*', a string of characters inside '[brackets]', and a comma-delimited list of strings inside '{curly,braces}'. They work like Unix shell filename globbing.

### 1.6 Bundles and Temporal Atomicity
A *bundle* is a sequence of messages and/or bundles. This recursive definition allows for arbitrary nesting of sub-bundles. All of the messages in the same bundle must be processed by the OSC server atomically; in other words it should be as if all of the messages in the bundle are processed in a single instant. An OSC packet may be a bundle or a (single) message.

### 1.7 Time Tags
Each bundle has a *time tag* that specifies the desired absolute time at which the messages in the bundle should take effect. The format is that used by the Internet Network Time Protocol [23] and provides sub-nanosecond accuracy over a range of over 100 years. OSC currently relies on an outside mechanism to synchronize clocks on different machines to the same absolute time, for example, NTP [23] or SNTP [24].

## 1.8 Queries

*Queries* are OSC messages that request the server to send information back to the client. Example queries include "what is the current list of nodes under this given node?", "what argument types are expected for messages sent to this given node?", "what is the value of the parameter that can be set by sending messages to this node?", and "please send me some documentation for the object or feature specified by this address."

# 2. IMPLEMENTATIONS OF OSC

CNMAT created OSC and maintains a web site, downloadable code, and developers' email list. As the public face of OSC we often hear about other people's use of OSC; this section lists the implementations and uses of OSC of which we are aware. Since the standard is open and our code is freely downloadable, we assume that there are other implementations of which we are not aware; we look forward to hearing about them. No doubt some of the details described in this section will be obsolete by the time this paper goes to press, especially, we hope, some of the limitations of certain systems.

All of the implementations mentioned in this section are linked from the main OSC home page at CNMAT (http://www.cnmat.berkeley.edu/OSC).

## 2.1 CNMAT's Open-Source OSC Software

All of the software mentioned in this section is available for download from CNMAT (http://www.cnmat.berkeley.edu/OSC).

When we introduced OSC in 1997, we released *OSC-client.c*, a C library for constructing OSC packets through a procedure call interface. There is nothing more to implementing an OSC client than constructing proper OSC packets and sending them over the network.

We also released a pair of text-based Unix command-line utilities: *sendOSC* allows the user to type in message addresses and arguments via a no-frills text interface, and formats and sends these messages via UDP to the desired IP address and port number; *dumpOSC* listens for OSC messages on a given UDP port and prints them out in a simple ASCII format.

As part of CNMAT's early efforts to promote OSC, we released the *OSC Kit* [32] in source code form in 1998. Our reasoning was that although the community as a whole was in favor of OSC and its features (as they had been of ZIPI), people would be reluctant to implement OSC (as they had been of ZIPI) unless we did a lot of the work for them (which we did not do for ZIPI). Thus, the OSC Kit implements most of the features needed for an OSC server: (dynamic) construction of an address space, parsing OSC packets, pattern-matching address patterns within the address space, associating a user-defined callback procedure with each node of the address space and invoking that procedure in response to messages sent to that node, and even a scheduler for implementing time tags. The OSC Kit is completely neutral to architecture and operating system and was designed not to degrade reactive real-time performance.

## 2.2 Music Programming Environments

All of the current OSC implementations known to the authors send and receive OSC packets only as UDP packets.

### *2.2.1 Max/MSP*

The first programming environment to implement OSC was *Max/MSP* [28, 36], in the form of Max "externals" written by Matt Wright. The *OpenSoundControl* external translates in both directions between native Max data lists and OSC-formatted binary data. The *otudp* external (as well as the now-obsolete *udp* external) sends and receives arbitrary UDP packets and can be used in conjunction with the *OpenSoundControl* object. These are implemented as separate objects to allow for transmission of OSC packets other than by UDP packets and to allow for transmission of UDP packets other than OSC packets. Finally, the *OSC-route* external enables the parsing of OSC address patterns by Max programmers and implements OSC's pattern matching features. All of these objects have been ported to the OSX version of Max/MSP.

The Max/MSP implementation has full support for sending and receiving messages and bundles, but there is currently no integration between OSC time tags and Max's scheduler and no support for queries. There is backwards-compatible support for both sending and receiving non-type-tagged messages. Temporal atomicity of bundles is handled by the fact that *OpenSoundControl* outputs a "bang" after outputting all of the messages in a bundle; it is the responsibility of the Max/MSP programmer to ensure that all of the messages take effect atomically when the "bang" is output.

### *2.2.2 SuperCollider*

James McCartney added OSC support to the object-oriented *SuperCollider* ("SC") language and environment [21] in 1998. The *OSCNode* object represents a node of the OSC address space and contains a symbolic name, a list of the children of the node, and a function to be called when the node receives a message. The *OSCOutPort* and *OSCInPort* objects represent UDP ports that can send or receive (respectively) OSC packets. Every *OSCInPort* has an *OSCNode* that is the root of the address space associated with that port.

There is a large sub-tree of OSC messages that can be sent to the SC environment itself, including "run the main patch," "stop synthesis," "play this sound file from the local disk," and even "compile and execute the code in the string argument to this message." There is also an OSC representation for all of the important MIDI messages (note on/off, continuous controllers, pitch bend, program change, channel and key pressure, and all-notes-off); when SC receives one of these OSC messages it's exactly as if SC had received the corresponding MIDI message via MIDI input.

Version 3 of SC, only for OSX, has a completely new architecture and is called *SuperCollider Server*. In this version, the synthesis engine is a separate application from the SC language and programming environment; the two communicate exclusively with OSC messages via UDP or TCP. This allows the SC synthesis engine to be controlled by applications other than the SC language.

### *2.2.3 Open Sound World*

*Open Sound World* (OSW) [5] is a scalable, extensible object-oriented language that allows sound designers and musicians to process sound in response to expressive real-time control. OSW has the same graphical dataflow model and nested subpatch structure as the Max family of languages; one important difference is that every OSW object has a symbolic name. Thus, the objects in an OSW patch automatically form an OSC-style hierarchical address space and can thus easily be addressed with OSC messages; the OSW kernel handles this automatically. OSW also provides an object called *OSCListen* that can be used to process incoming OSC messages manually; this allows OSW programmers to construct an OSC address space that does not necessarily reflect the tree structure of the OSW program that is the OSC server.

OSW has the best support of OSC queries of any implementation known to the authors, thanks in large part to recent work by Andy Schmeder at CNMAT. The dynamic address space of an OSW program can be discovered by a querying client. Any message that can be understood by any of the objects in an OSW patch can be sent to that object via OSC. An OSC client can get the current value of any variable of any OSW object.

OSW fully supports type tags. There is currently no connection between OSC time tags and OSW's notion of the current time. A careful programmer can use OSW's "synchronous outlets" mechanism to ensure that all elements within a bundle will be processed atomically.

### 2.2.4 Pd

OSC support in the *Pd* programming language and environment [29] is in the form of third-party objects. The *sendOSC* and *dumpOSC* objects are for sending and receiving OSC packets and are derived from CNMAT's text-based Unix command-line utilities of the same name.

The *sendOSC* object must be set to write to a given IP address and UDP port. Then it translates Pd lists to properly-formatted OSC messages and sends them. There is also support for creating bundles, but not for specifying bundles' time tags.

The *dumpOSC* object creates a UDP socket, parses incoming OSC packets on that port, converts each OSC message to a Pd list, and outputs the lists sequentially. Time tags are ignored. There is no mechanism to assist with temporal atomicity of bundles; in fact, no representation of the bundle structure of incoming OSC packets is available to the Pd programmer—consecutive lists output by *dumpOSC* might be from the same bundle or from different OSC packets entirely.

The *routeOSC* object is derived from and practically identical to the Max/MSP *OSC-route* object; it supports the parsing of address patterns with pattern matching.

Pd does not currently support queries.

### 2.2.5 Virtual Sound Server

NCSA's *Virtual Sound Server* ("VSS") [2] is an environment for real-time interactive sound synthesis; it is designed to be used in conjunction with graphics rendering software and includes mechanisms for synchronization of its audio with other applications' video. VSS can be controlled with a limited form of OSC utilizing a flat address space. Type tag strings, pattern matching, bundles, time tags and queries are not supported.

### 2.2.6 Csound

Csound support for OSC currently exists only as part of the "unofficial" release (http://web.tiscali.it/mupuxeddu/csound). This implementation is based on the OSC Kit and allows users to define Csound orchestras that can be controlled by OSC. The Csound programmer can name elements of the OSC address space, but the overall tree structure of the OSC address space is constrained by the fact that all Csound instruments are at the same level in a flat namespace. A procedure called at the K-rate checks for and processes newly-received OSC messages.

## 2.3 Software Synthesizers

*Grainwave* [3] is a software granular synthesizer with very limited OSC support. It accepts MIDI messages formatted as OSC messages; the use of OSC is solely as a mechanism to transmit MIDI-style data over the Internet.

Native Instruments' *Reaktor* (www.native-instruments.com) is a general-purpose environment for building software synthesizers. Reaktor's OSC support in version 3 is similar to Grainwave's, essentially just MIDI over the Internet, but version 4, currently still in beta, is said to have much more integrated OSC support.

## 2.4 General Purpose Programming Languages

All of the implementations described in this section are available in source code form via CNMAT's OSC home page.

Chandrasekhar Ramakrishnan has implemented Java classes that can create OSC packets via a procedural interface and send them in UDP packets [30]. It supports type tags but not time tags. Future plans include the ability to receive OSC.

C. Ramakrishnan has also built an Objective-C wrapper around OSC-Client.c, designed primarily to allow Cocoa applications to send OSC messages.

There is an implementation of OSC in Perl (http://barely.a.live.fm/pd/OSC/perl). The sending half is implemented by a Perl wrapper around the *sendOSC* program that was created automatically with the *SWIG* interface compiler (http://www.swig.org). The receiving half is a port of the *dumpOSC* program to Perl; it provides a function called *ParseOSC* that takes in a binary OSC packet (such as data received via Perl's built-in UDP implementation) and returns the address and arguments of an OSC message.

There are two OSC implementations for Python; unfortunately both are Python source files with the name "OSC.py". Daniel Holth's and Clinton McChesney's *pyOSC*, part of the *ProctoLogic* project (http://galatea.stetson.edu/~ProctoLogic), translates bidirectionally between the binary OSC format and Python data types. Bundles are read correctly but cannot be constructed. It also includes a *CallbackManager* that allows a Python programmer to associate Python callbacks with OSC addresses and then dispatches incoming OSC messages. Unfortunately pattern matching is not yet implemented. ProctoLogic is covered by the LGPL.

Stefan Kersten's *OSC.py* is a Python module for OSC clients (http://user.cs.tu-berlin.de/~kerstens/pub/OSC.py). It can construct arbitrary OSC packets and send them in UDP packets, and can even produce OSC time tags based on Python's built-in time procedures.

Smalltalk also has two implementations of OSC. *VWOSC* (http://www.mat.ucsb.edu/~c.ramakr/illposed/vwosc.html) was written by C. Ramakrishnan and Stephen Pope and currently only send OSC. The *Siren* Music and Sound Package for Squeak Smalltalk (http://www.create.ucsb.edu/Siren) includes an experimental OSC implementation.

## 2.5 Web Graphics Systems

Macromedia's *Flash* displays web application front-ends, interactive web site user interfaces, and short-form to long-form animation. It contains a scripting language called *ActionScript* that has good support for manipulating XML documents as well as a mechanism for sending and receiving streamed XML documents via a TCP/IP socket. Ben Chun has defined an XML document type to represent OSC packets in XML and created a bidirectional gateway between Flash and OSC with a program called *flosc* [6] that translates between OSC packets over UDP and XML documents over TCP.

As a multimedia authoring tool designed to create rich interactive content for both fixed media and the Internet, Macromedia's *Director* can incorporate photo-quality images, full-screen or long-form digital video, sounds, animation, 3D models, text, hypertext, bitmaps, and Macromedia Flash content. Garry Kling at UCSB has written an extension ("xtra") to Director called OSCar [18] that can send OSC packets from the *Lingo* scripting language. Future plans include the ability for Lingo to receive OSC.

### 2.6 Gesture-to-OSC Hardware

The *Kroonde* [19] is a system for receiving data from wireless sensors, for example, pressure, flexion, acceleration, magnetic field, and light sensors. The Kroonde receiver takes in data from up to 16 of these sensors and converts them either to MIDI or to OSC over UDP over 10 BaseT Ethernet.

Dan Overholt has built an interface called the *MATRIX* ("Multipurpose Array of Tactile Rods for Interactive eXpression") that consists of a 12x12 array of spring-mounted rods each able to move vertically. An FPGA samples the 144 rod positions at 30 Hz and transmits them serially to a PC that converts the sensor data to OSC messages [25, 26].

Newton Armstrong at Princeton has built prototype hardware (http://music.princeton.edu/~newton/controller.html) with 11 continuous and 40 switch analog inputs, which are digitized, converted to OSC messages, and sent as UDP packets via a built-in 10BaseT Ethernet port.

There are plans for the next version of IRCAM's *AtoMIC Pro* gesture-acquisition hardware [9] to output OSC rather than MIDI as it does now.

## 3. OSC-based Networking Applications

Here is a somewhat chronological survey of networked music applications that have been built with OSC. It is certainly not comprehensive; we encourage all users of OSC to inform us about their projects.

At ICMC 2000 in Berlin (http://www.audiosynth.com/icmc2k), a network of about 12 Macintoshes running SuperCollider synthesized sound and changed each others' parameters via OSC, inspired by David Tudor's composition "Rainforest."

The *Meta-Orchestra* project [16] is a large local-area network that uses OSC.

In Randall Packer's, Steve Bradley's, and John Young's "collaborative intermedia work" *Telemusic #1* [35], visitors to a web site interact with Flash controls that affect sound synthesis in a single physical location. The resulting sound is streamed back to the web users via RealAudio. This system was implemented before *flosc* and before Flash's *XMLSockets* feature existed, so data goes from Flash to JavaScript to Java to OSC.

In a project [17] at the MIT Media Lab, the analyzed pitch, loudness, and timbre of a real-time input signal control sinusoids+noise additive synthesis. The mapping is based on Cluster-Weighted Modeling and requires extensive offline analysis and modeling of a collection of sounds. In one implementation, one machine performs the real-time analysis and sends the control parameters over OSC to a second machine performing the synthesis.

Three projects at UIUC are based on systems consisting of real-time 3D spatial tracking of a physical object, processed by one processor that sends OSC to a Macintosh running Max/MSP for sound synthesis and processing.. In the *eviolin* project [13], a Linux machine tracks the spatial position of an electric violin and maps the spatial parameters to control a resonance model in real-time. The sound output of the violin is processed through this resonance model. In the *Interactive Virtual Ensemble* project [12], a conductor wears wireless magnetic sensors that send 3D position and orientation data at 100 Hz to a wireless receiver connected to an SGI Onyx that processes the sensor data. In this system, the Max/MSP software polls the SGI via OSC to get the current sensor values. *VirtualScore* is an immersive audiovisual environment for creating 3D graphical representations of musical material over time [11]. It uses a CAVE to render 3D graphics and to receive orientation and location information from a wand and a head tracker. Both real-time gestures from the wand and stored gestures from the "score" go via OSC to the synthesis server.

Stanford's CCRMA's *Circular Optical Object Locator* [14] is based on a rotating platter upon which users place opaque objects. A digital video camera observes the platter and custom image-processing software outputs data based on the speed of rotation, the positions of the objects, etc. A separate computer running Max/MSP receives this information via OSC and synthesizes sound.

*Listening Post* [15] is a networked multimedia art installation based on representing conversations in Internet chat rooms on a large number of video monitors and also with sonification via a 10-channel speaker system. A local network of 4 computers handle text display, text-to-speech, sound synthesis, and coordination of all these elements; all of the components of the system communicate with OSC. *Listening Post* is currently on display at the Whitney museum of American Art.

A research group at UCSB's CREATE has been developing "high-performance distributed multimedia" and "distributed sensing, computation, and presentation" systems [27]. These large-scale networks typically consist of multiple sensors such as VR head-trackers and hand-trackers, dozens of computers interpreting input, running simulations, and rendering audio and video, and multichannel audio and video output, all connected with CORBA and OSC. Another UCSB project [8] combines CORBA and OSC to allow a VR system with data gloves and motion trackers to send control messages to synthesis software written in SuperCollider.

In the *Tgarden* project [31], wireless accelerometers are sensed by a Linux machine and mapped via OSC to control sound and video synthesis in Max/MSP, SuperCollider, and NATO.

## 4. OSC Pedagogy

University courses teaching OSC include the following:

- Iowa State Music 448 ("Computer Music Synthesis")
- Princeton COS436 ("Human Computer Interface Technology")
- Stanford Music 250a ("HCI Seminar")
- UC Berkeley Music 158 ("Musical Applications of Computers and Related Technologies") and 209 ("Advanced Topics in Computer Music") and CNMAT's summer Max/MSP Night School.
- UC Santa Barbara Music 106 ("Interactive Electronic Performance and Synthesizer Design Using Max/MSP")

## 5. BENEFITS OF ORGANIZING REAL-TIME MUSIC SOFTWARE WITH OSC

This section describes some programming techniques that make use of OSC as the primary organizational scheme for building real-time performance instruments. Although our examples concentrate on the Max/MSP programming environment, the described techniques can be generalized to other platforms and aim in general to improve modularity and interconnectivity of software components.

### 5.1 A Module's OSC Namespace Is Its Entire Functionality

We propose a style of programming in which the entire functionality of each software module is addressable through OSC messages. Advantages of this style include the following:

1. The OSC namespace for each module explicitly names all of that module's features. This can enable software to be self-documenting and transparent in its functionality.
2. The entire functionality of each module is accessible via a single control mechanism: incoming OSC messages. In graphical languages such as Max/MSP and Pd, this allows even the most complex objects to have a single control inlet, reducing the clutter and confusion of connecting to multiple inlets (Figure 1). As a module's functionality grows, no structural changes (such as adding more inlets) are required; the programmer simply expands the module's OSC namespace
3. If the components of a complex system already communicate among themselves exclusively with OSC messages, then it becomes very easy to move some of the components to other computers to form a distributed local area network system.
4. Certain OSC messages can be standardized across different modules. For example, the message "/gain" with a floating point argument can be used in many different synthesis and processing components to change gain; the message "/namespace" can trigger any module to display its OSC namespace; the message "/go" followed by the argument 1 or 0 can be used to turn on and off the processing in the module; and the message "/init" can initialize a module.

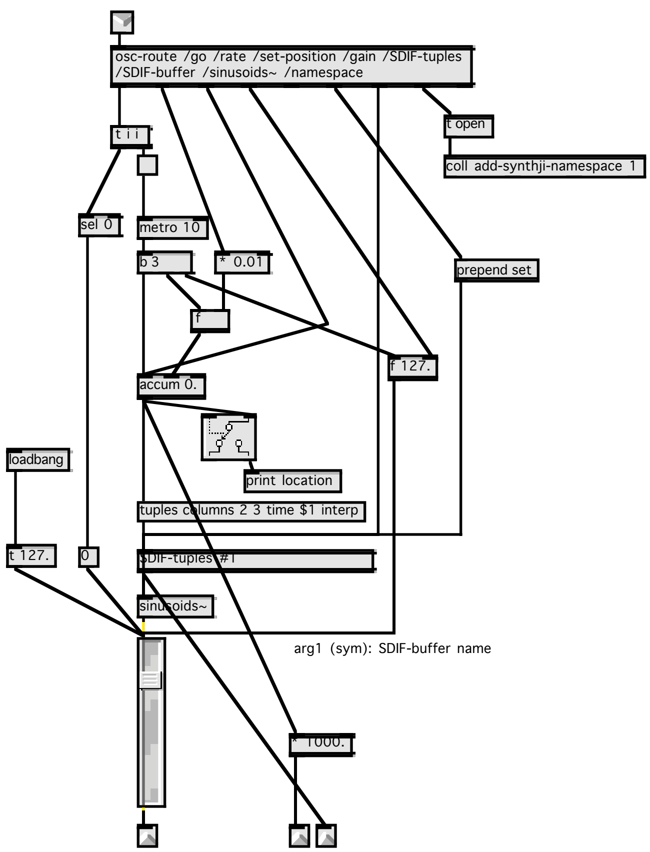



**Figure 1: A Max/MSP patch that performs additive synthesis in real-time using Sinusoidal Track data stored in an SDIF-buffer. Everything the patch does is accessible through a single inlet and is described by the list of OSC messages that the patch understands.**

By sending the "/namespace" message to the patch in figure 1 the user is presented with this list of OSC messages and can quickly learn how to control the patch:

1, /go _int_ turn processing on and off;
2, /rate _float_ play rate;
3, /set-position _float_ between 0 and 1 sets position in buffer from start to end;
4, /gain _float_ sets gain;
5, /SDIF-tuples _anything_ talks to SDIF-tuples;
6, /SDIF-buffer _symbol_ sets SDIF-buffer for playback;
7, /sinusoids~ _anything_ talks to sinusoids~;
8, /namespace _bang_ opens this collection;

## 5.2 Storing and Recalling Global Snapshots of Complex Software Components with OSC

In developing complex software instruments that perform many processes with many possible arrangements of parameters, the task of storing and retrieving complete snapshots of the system's state can be quite challenging. We propose a system of performing this storage and recalling that is based entirely on the usage of OSC messages as the communication scheme between the instrument's snapshot mechanism and its constituent modular components.

Once an instrument comprised of a set of components—all of whose functionality is addressable with OSC messages—is developed, it is possible to store and recall global settings of the entire system by collecting and dispensing OSC messages from and to the individual components. We propose a model where each module keeps track of its current state by remembering what OSC messages have been sent to it most recently. Note that since the OSC name for each function of the module is unique, this can be achieved by using the OSC message as an index whose value is replaced each time a new value is received. In order to collect a snapshot, we query each component for its current state. Each component answers the query in the form of a list of OSC message that will bring it back to its current state if sent to it at a later time. OSC messages from each component are then collected and stored in one central location. In order to recall a stored global snapshot of the system, one simply has to send out the list of OSC messages that each component submitted earlier.

This method was successfully employed in developing Ron Smith's work for orchestra and live electronics titled *Constellation* [20], as well as the collaborative dance piece of Carol Murota, Edmund Campion and Ali Momeni entitled *Persistent Vision* [4], a work for 16 dancers and live interactive sound installation.

## 5.3 Managing Polyphony With OSC

Many of the platforms for developing specialized real-time audio/video software include some tools for building polyphony, e.g., Max/MSP's *poly~* object and Pd's *nqpoly~* object. By using a simple abstraction that routes OSC messages to specific voices of a polyphonic component (figure 2), OSC's pattern-matching features can be used to address specific voices or sets of voices with great ease.

## 5.4 Dynamic Routing of Controller Data with OSC

We continue to advocate the use of OSC as the bridge between input data streams from gestural controllers and signal processing engines. This style of programming involves describing a complete OSC namespace for all output streams from a controller [34]. Intermediary patches dynamically map the control data to the OSC namespace for a

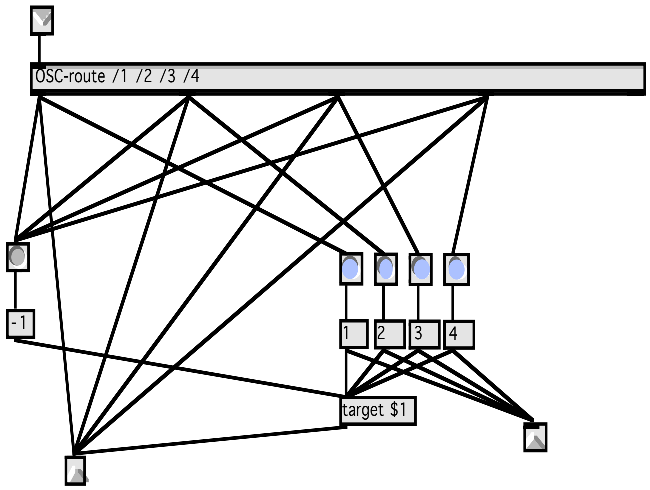



**Figure 2: A Max/MSP patch that routes OSC messages starting with the desired voice number to that voice of a 'poly~' object (which uses the 'target' message to address specific voices). The patch was created with Max/MSP's scripting mechanism and could have been made with any number of voices.**

specific signal processing engine. This allows the performer to change instruments by switching from one intermediary-mapping-patch to another, thereby directing his controller's OSC output to a different signal processing module.

In the past year, we have further developed our implementations of this approach to gesture mapping for a number of controllers including the Buchla Thunder, Wacom drawing tablets (via a new interface using Cycling 74's Jitter), and the game controller Cyborg 3D made by Saitek.

Finally, we promote the use of OSC for designing controller data streams that are *modal*. For example, the Saitek Cyborg 3D joystick provides an extremely flexible controller due to the number of buttons it has accessible to the performer in conjunction with its 4 continuous controllers (figure 3).


**Figure 3: The Saitek Cyborg 3D joystick has 13 buttons and 4 continuous controllers. The three buttons on the top, labeled 1 to 3, can be used to route the continuous controller values to different destinations.**

It is often desirable to control a number of processes with one controller, for instance multiple voices of a polyphonic engine. In the case of the Cyborg 3D, OSC messages in the form of '/button-number/continuous-controller value' can be constructed that will render the continuous contoller values *modally* addressable to different voices. For example, holding down button 1 and moving the joystick up and down would produce messages like, '/1/vertical value', headed to the first voice of our processing engine. Holding down buttons 2 and 3 while moving the vertical axis of the controller would produce both '/2/vertical value' and /3/vertical value' messages, thereby controlling the second and third voices of the processing engine. A similar technique can be applied to any combination of held buttons and manipulated continuous controllers to effectively turn the 4 available continuous controllers into a much larger number of control data sources.

## 6. FUTURE OF OSC

Here are some ideas for the future of OSC. Obviously, all implementations of OSC should be completed and made consistent, able to both send and receive the full OSC spec including type tags, bundles, time tags, etc. Full use of time tags requires solving the time synchronization problem; experiments must be done to see if NTP and SNTP will be adequate.

There is no reason that OSC should be so tied to UDP; more systems should support OSC via TCP, especially in situations where guaranteed delivery is more important than low latency.

OSC's query system is still more or less in an experimental stage; the community should standardize the syntax and semantics of a collection of useful queries.

Of course we would like to see more systems using OSC. On the day this paper was submitted we heard that Carlos Agon had completed an initial implementation of OSC in both OpenMusic [1] and Macintosh Common Lisp. The jMax [7] team is also planning to implement OSC.

The translation between OSC and XML used by *flosc* could be generally useful to the OSC community; we would like to see it become standardized.

## 7. ACKNOWLEDGMENTS

Amar Chaudhary, John Ffitch, Guenter Geiger, Camille Goudeseune, Peter Kassakian, Stefan Kersten, James McCartney, Marcelo Wanderley, David Wessel.

## 8. REFERENCES

[1] Agon, C., Assayag, G., Laurson, M. and Rueda, C. Computer Assisted Composition at Ircam : PatchWork & OpenMusic. (1999) Computer Music Journal. (http://www.ircam.fr/equipes/repmus/RMPapers/CMJ98/index.html)

[2] Bargar, R., Choi, I., Das, S. and Goudeseune, C., Model-based interactive sound for an immersive virtual environment, in International Computer Music Conference, (Aarhus, Denmark, 1994), ICMA, 471-474. (Current documentation and software: http://www.isl.uiuc.edu/software/software.html)

[3] Berry, M., GrainWave 3, 2002. (http://www.7cities.net/users/mikeb/GRAINW.HTM).

[4] Campion, E., Momeni, A. and Murota, C., Persistent Vision, 2002. (http://cnmat.CNMAT.Berkeley.EDU/~ali/documents/persisten-vision.html).

[5] Chaudhary, A., Freed, A. and Wright, M. An Open Architecture for Real-Time Audio Processing Software, Audio Engineering Society, Audio Engineering Society 107th Convention, preprint #5031, 1999. (http://cnmat.CNMAT.Berkeley.EDU/AES99/docs/osw.pdf)

[6] Chun, B., flosc : Flash OpenSound Control, 2002. (http://www.benchun.net/flosc).

[7] Dechelle, F., Borghesi, R., De Cecco, M., Maggi, E., Rovan, B. and Schnell, N. jMax: An environment for real-time musical applications. (1999) Computer Music Journal, 23 (3). 50-58.

[8] Durand, H. and Brown, B.D., A Distributed Audio Interface using CORBA and OSC, in Symposium on Sensing and Input for Media-Centric Systems (SIMS), (Santa Barbara, CA, 2002), 44-48. (http://www.create.ucsb.edu/sims/PDFs/Durand_Brown_SIMS.pdf)

[9] Fléty, E., AtoMIC Pro: a Multiple Sensor Acquisition Device, in Conference on New Interfaces for Musical Expression (NIME-02), (Dublin, Ireland, 2002), 96-101.

[10] Friedl, J. Mastering Regular Expressions: Powerful Techniques for Perl and Other Tools. O'Reilly & Associates, Sebastopol, CA, 1997.

[11] Garnett, G.E., Choi, K., Johnson, T. and Subramanian, V., VirtualScore: Exploring Music in an Immersive Virtual Environment, in Symposium on Sensing and Input for Media-Centric Systems (SIMS), (Santa Barbara, CA, 2002), 19-23. (http://www.create.ucsb.edu/sims/PDFs/Garnett_SIMS.pdf)

[12] Garnett, G.E., Jonnalagadda, M., Elezovic, I., Johnson, T. and Small, K., Technological Advances for Conducting a Virtual Ensemble, in International Computer Music Conference, (Habana, Cuba, 2001), 167-169.

[13] Goudeseune, C., Garnett, G. and Johnson, T., Resonant Processing of Instrumental Sound Controlled by Spatial Position, in CHI '01 Workshop on New Interfaces for Musical Expression (NIME'01), (Seattle, WA, 2001), ACM SIGCHI. (http://www.csl.sony.co.jp/~poup/research/chi2000wshp/papers/goudeseune.pdf)

[14] Hankins, T., Merrill, D. and Robert, J., Circular Optical Object Locator, in Conference on New Interfaces for Musical Expression (NIME-02), (Dublin, Ireland, 2002), 163-164.

[15] Hansen, M. and Rubin, B., LISTENING POST: GIVING VOICE TO ONLINE COMMUNICATION, in International Conference on Auditory Display, (Kyoto, Japan, 2002). (cm.bell-labs.com/cm/ms/who/cocteau/papers/pdf/icad2002.pdf)

[16] Impett, J. and Bongers, B., Hypermusic and the Sighting of Sound - A Nomadic Studio Report, in International Computer Music Conference, (Habana, Cuba, 2001), ICMA, 459-462. (91 page report: http://www.meta-orchestra.net/downloads/report.pdf)

[17] Jehan, T. and Schoner, B., An Audio-Driven Perceptually Meaningful Timbre Synthesizer, in International Computer Music Conference, (Habana, Cuba, 2001), 381-388. (http://www.media.mit.edu/hyperins/papers/Tristan_ICMC_2001.pdf)

[18] Kling, G. and Schlegel, A., OSCar and OSC: Implementation and use of Distributed Multimedia in the Media Arts, in Symposium on Sensing and Input for Media-Centric Systems (SIMS), (Santa Barbara, CA, 2002), 55-57. (http://www.create.ucsb.edu/sims/PDFs/OSCar_SIMS.pdf)

[19] La Kitchen Hardware, KROONDE: 16 sensors wireless UDP interface, 2002. (http://www.la-kitchen.fr/hardw/presentation_en.pdf).

[20] Madden, T., Smith, R.B., Wright, M. and Wessel, D., Preparation for Interactive Live Computer Performance in Collaboration with a Symphony Orchestra, in International Computer Music Conference, (Habana, Cuba, 2001), ICMA.

[21] McCartney, J., A New, Flexible Framework for Audio and Image Synthesis, in International Computer Music Conference, (Berlin, Germany, 2000), ICMA, 258-261. (www.audiosynth.com)

[22] McMillen, K., Wessel, D. and Wright, M. The ZIPI music parameter description language. (1994) Computer Music Journal, 18 (4). 52-73. (http://cnmat.CNMAT.Berkeley.EDU/ZIPI/mpdl.html)

[23] Mills, D.L., Network Time Protocol (Version 3) Specification, Implementation and Analysis, 1992. (http://www.faqs.org/rfcs/rfc1305.html).

[24] Mills, D.L., Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI, 1996. (http://www.faqs.org/rfcs/rfc2030.html).

[25] Overholt, D., The MATRIX: A Novel Controller for Musical Expression, in CHI '01 Workshop on New Interfaces for Musical Expression (NIME'01), (Seattle, WA, 2001), ACM SIGCHI. (http://www.csl.sony.co.jp/person/poup/research/chi2000wshp/papers/overholt.pdf)

[26] Overholt, D., Musical Mapping and Synthesis for the MATRIX Interface, in Symposium on Sensing and Input for Media-Centric Systems (SIMS), (Santa Barbara, CA, 2002), 7-10. (http://www.create.ucsb.edu/sims/PDFs/DanO_SIMS.pdf)

[27] Pope, S.T. and Engberg, A., Distributed Control and Computation in the HPDM and DSCP Projects, in Symposium on Sensing and Input for Media-Centric Systems (SIMS), (Santa Barbara, CA, 2002), 38-43. (http://www.create.ucsb.edu/sims/PDFs/Pope_SIMS.pdf)

[28] Puckette, M. Combining event and signal processing in the MAX graphical programming environment. (1991) Computer Music Journal, 15 (3). 68-77.

[29] Puckette, M.S., Pure Data, in International Computer Music Conference., (Hong Kong, 1996), ICMA, 269-272. (http://www-crca.ucsd.edu/~msp/Publications/icmc96.ps)

[30] Ramakrishnan, C., Java OSC, 2003. (http://www.mat.ucsb.edu/~c.ramakr/illposed/javaosc.html).

[31] Wei, S.X., Visell, Y. and MacIntyre, B. TGarden Media Choreography System, Technical Report, Graphics, Visualization and Usability Center, Georgia Institute of Technology, Atlanta, GA, 2003. (http://www.gvu.gatech.edu/people/sha.xinwei/topologicalmedia/papers/Sha-MacIntyre02.pdf)

[32] Wright, M., Implementation and Performance Issues with OpenSound Control, in International Computer Music Conference, (Ann Arbor, Michigan, 1998), ICMA, 224-227. (http://cnmat.CNMAT.Berkeley.EDU/ICMC98/papers-html/OSC-kit.html)

[33] Wright, M., OpenSound Control Specification, 2002. (http://www.cnmat.berkeley.edu/OSC/OSC-spec.html).

[34] Wright, M., Freed, A., Lee, A., Madden, T. and Momeni, A., Managing Complexity with Explicit Mapping of Gestures to Sound Control with OSC, in International Computer Music Conference, (Habana, Cuba, 2001), 314-317. (http://cnmat.CNMAT.Berkeley.EDU/ICMC2001/pdf/OSC-GestureMap.2.pdf)

[35] Young, J.P., Using the Web for Live Interactive Music, in International Computer Music Conference, (Habana, Cuba, 2001), 302-305.

[36] Zicarelli, D., An Extensible Real-Time Signal Processing Environment for Max, in International Computer Music Conference, (Ann Arbor, Michigan, 1998), International Computer Music Association, 463-466.

# Gestural Control of Music Using the Vicon 8 Motion Capture System

Christopher Dobrian
University of California, Irvine
303 Music & Media Bldg., UCI
Irvine, CA 92697-2775
(1) 949-824-7288
dobrian@uci.edu

Frédéric Bevilacqua
University of California, Irvine
1002 Health Sciences Road East
Irvine, CA 92612-1475
(1) 949-824-4104
fbevilac@uci.edu

## ABSTRACT
This article reports on a project that uses unfettered gestural motion for expressive musical purposes. The project involves the development of, and experimentation with, software to receive data from a Vicon motion capture system, and to translate and map that data into data for the control of music and other media such as lighting. In addition to the commercially standard MIDI—which allows direct control of external synthesizers, processors, and other devices—other mappings are used for direct software control of digital audio and video. This report describes the design and implementation of the software, discusses specific experiments performed with it, and evaluates its application in terms of aesthetic pros and cons.

## Keywords
Motion capture, gestural control, mapping.

## 1. INTRODUCTION
The Vicon 8 motion capture system[1] is recognized to be one of the best available systems for accurately recording three-dimensional movement, particularly—but by no means exclusively—movement of the human body. At the University of California, Irvine (UCI), a small group of artists and programmers is working in UCI's Motion Capture Studio, in conjunction with the Realtime Experimental Audio Laboratory (REALab), to develop software for the translation and mapping of Vicon motion capture data—which is normally used for animation or for biomechanics research—into control data for music, as well as for other media such as lighting, digital audio, and digital video. The working name of this software is MCM (Motion Capture Music).

The team is developing MCM concurrently on two programming platforms. One group is coding it in generic Java and C++ for optimal portability to any operating system, translating received motion capture data into directly transmitted MIDI data. A second group is coding extended functionality in Max[2]. The Max version of MCM is presently limited to the Macintosh operating system, but receives the realtime motion capture data via Open Transport UDP. The advantage of using Max is that it already provides comprehensive capabilities for the control of digital audio and video—via MSP and Jitter—and therefore allows direct mapping of motion capture data to those media as well as to MIDI.

The goal of the software development is to create a straightforward, useful tool for employing gestural motion to control audio-visual performance media with accuracy and reliability. Members of the team are conducting experiments with this type of control, and these experiments are guiding the software development as it is in progress. What follows is a report on this work.

## 2. DATA FROM THE VICON V8 SYSTEM
The details of the Vicon system are adequately described elsewhere[1,3], so here we discuss only the data it produces. The data derived from the captured motion are most commonly saved to disk, as a Vicon-standardized .C3D file. Captured data files are then normally used as input to an animation program such as 3D Studio Max for realistic generation of lifelike animated characters, or used for biomechanical studies of bodily motion (sports, physical therapy, ergonomics, etc.). In the .C3D format, each frame of information is represented as a list consisting of Cartesian *x, y, z* coordinates in 3D space for each marker. The Vicon 8 system at UCI reports up to 120 frames per second; more modern Vicon systems boast a much higher frame rate. The user determines the ordering of the markers in the list when recording the data.

### 2.1 Vicon Real-time Data
The Vicon system has recently become of interest for musical expression because of the availability of the Vicon Real-time Engine[4], which allows full realtime transmission of the motion capture data. For this project, Vicon Motion Systems provided their in-house RTEmulator software, which allows one to emulate the behavior of the Real-time Engine. RTEmulator reads data from a .C3D file and transmits it in the format of the Real-time Engine, allowing simulation and testing of realtime motion capture without having the Vicon Real-time Engine itself. Realtime access to Vicon data makes the system useful as a tool for musical expression.

## 3. INITIAL DESIGN OF MCM
The initial design[5] for MCM intended to make as simple—and as simple to use—a program as possible for mapping motion capture data to musical control data. The design allows for the user to select a marker (i.e. a position on the body), a coordinate (*x*, *y*, or *z*), and a range of space to in which to track that coordinate, and linearly map that data to any range of MIDI values for any MIDI channel message. The user can specify as many such mappings as desired. The intention was to make the most direct possible way for any motion capture parameter to be used to control a MIDI device. For more complex mappings, it was assumed that an additional program would mediate the transmission between MCM and the MIDI device(s).

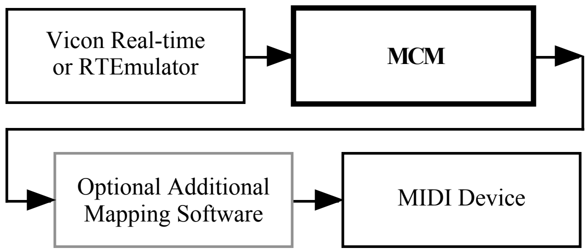



**Figure 1. Overview of Generic Usage of MCM**

## 3.1 Specifying Mappings and Translating Data

Per this initial design plan, the implementation of MCM consists of two parts: a user interface for specifying the desired mappings of marker coordinates to MIDI data, and a translator that uses those specifications to perform the desired mappings on incoming motion capture data and transmit the appropriate MIDI data. These two parts are implemented as separate programs, called MCMMap and MCMTranslate.

### *3.1.1 MCMMap*

MCMMap provides the user with panels (as many as desired) for inputting the desired mappings. The parameters to be specified by the user are marker, coordinate, coordinate range minimum and maximum, MIDI message type, MIDI range minimum and maximum, and MIDI channel. Depending on the type of MIDI message selected, other parameters may appear: e.g., controller number if a control change message is selected, velocity and duration if a note message is selected, etc. A session of such specifications is saved in a plain text file with a .mcm tag. The .mcm file is used by MCMTranslate to map motion capture data to MIDI data.

### *3.1.2 MCMTranslate*

MCMTranslate performs four tasks: it receives motion capture data from Vicon Real-time or RTEmulator, uses a .mcm file of mapping specifications to translate specific input values into MIDI messages, transmits the MIDI messages, and schedules note-off messages in the future to end any note-on messages it has transmitted. Since all of the mapping specifications are made in MCMMap, MCMTranslate requires nothing more of the user than to start it and select a .mcm file for the translation of the incoming data.

# 4. IMPLEMENTATION IN JAVA/C++

A relatively "platform-neutral" version of MCM is being implemented according to the initial design described above, using Java for MCMMap and C++ for MCMTranslate. The intention is for it to be maximally compatible with the operating system used by the Vicon system itself, Windows NT/2000, yet written with minimal reliance on OS-specific functionality, so as to be as easily portable as possible to any new OS. This implementation was begun by undergraduate students from the UCI Department of Information and Computer Science (ICS), Cayci Suitt and Gene Wie, and is being completed by ICS students Mark Magpayo and Maybelle Tan.

# 5. IMPLEMENTATION IN MAX

The authors are also implementing an extended version of MCM in the Max programming environment. The Max version (MCMMax) incorporates a number of extensions to the basic design that make it more versatile and useful for more complex mappings of gesture to music. Because Max is an inherently object-based "patchable" system of modules, MCMMax allows for easy redirection of input data to different types of mapping. Because Max already has considerable features for audio and video—MSP and Jitter—incorporated directly in the programming environment, it's a simple matter to create new modules for controlling these media as needed; the mapped data can thus control MIDI synthesizers and sound processors, digital audio, and digital video all within the same software system.

## 5.1 Design Extensions in MCMMax

MCMMax extends the initial design by permitting tracking of more types of input information, selection from a variety of different mapping schemes, and selection of different destinations for the mapped data.

### *5.1.1 Information Derived from the Received Data*

The motion capture data received from the Vicon system consists of *x, y, z* position values for each marker being tracked. In addition to the position of any marker in any dimension, MCMMax allows the user to track other information: marker velocity in one, two, or three dimensions; marker acceleration in one, two, or three dimensions; the distance between any two markers; and the angle formed by any two or three markers. As with the basic version of MCM, one can specify a range of input values to track.

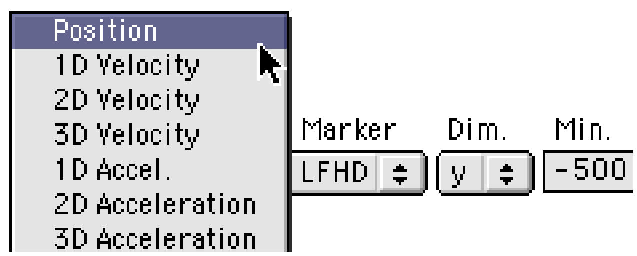



**Figure 2. Input Selection in MCMMax**

### *5.1.2 Mappings*

In addition to the linear mapping used in MCM, MCMMax also provides the choice of reversed, exponential, logarithmic, and non-linear (lookup table) mappings. When the user selects exponential or logarithmic mappings, a new field appears for entering an exponent or base; when the user selects non-linear mapping, the user may open a stored lookup table or draw an arbitrary mapping curve.

### *5.1.3 Destinations for Mapped Data*

In MCMMax, transmitting mapped data as MIDI messages is just one of the possible choices of destination. Because of the ease with which the data can be routed to different subroutines in Max, the data can just as easily be mapped to serve as control data for digital audio in MSP or digital video in Jitter. We have already performed experiments with mapping motion capture data to MSP parameters (e.g., filter cutoff frequency, etc.), and will continue to develop a variety of different mapping modules that can easily be plugged into MCMMax as new features.

### *5.1.4 Gesture Detection and Recognition*

Some of the research work in the UCI Motion Capture Studio has focused on the detection of individual gestures within the stream of motion capture data, and on recognizing particular kinds of gestures.[3] Such gesture detection has been demonstrated to be feasible and musically useful, and the results of those experiments—edge detection in acceleration curves to detect important changes of gesture (by ICS graduate student Jeff Ridenour), and principle component analysis to recognize particular types of gesture (F. Bevilacqua)—may also be easily encapsulated as plug-ins for MCMMax.

# 6. AESTHETIC DIRECTIONS

The numerical and musical problems of mapping captured gestural data to musical control have been discussed in writings by these authors and others.[3,6,7] We will point out here the primary challenges in the development of new musical performances using the Vicon Motion System as input for realtime gestural control of music.

### *6.1.1 Performing the Virtual Instrument*

A performer of this system directly produces and controls musical events with no tangible physical interface, and is thus performing a purely "virtual" instrument. There is no haptic feedback as there is with any physical device, and no precedent or restriction guiding or dictating the gesture. These facts can be viewed as challenges or obstacles to performance of deterministic composed music, but can be seen as opportunities for improvisation and for the discovery of "musicality" inherent in bodily movement. We are thus interested more in the immanent musicality of the human form than in a proof of concept of yet another alternative controller. This pursuit requires that performer(s) be skilled in both musical improvisation and movement, yet not be tied to traditional dance or music vocabularies. That is a significant aesthetic challenge, but one that encourages and requires the collaboration of composers, programmers, and performers.

### *6.1.2 Multiplicity of Parameters*

A single performer wearing a standard set of thirty markers, with three coordinates per marker, produces a stream of 90 simultaneous continuous parameters available for musical control. (To say nothing of other available information such as velocity, acceleration, distance, angle, etc., or of multiple performers.) This profusion of control data presents a management challenge for the composer, and a challenge of the limitations of awareness for the performer. To solve this problem by using only a small number of marker coordinates would be to ignore the unique power and potential of this system. The opportunity and challenge of this system is to devise strategies for mapping so very many degrees of freedom into a meaningfully expressive whole.

### *6.1.3 Lack of Portability*

The Vicon 8 system requires circular placement of its eight cameras, and takes some time to set up and calibrate, so it is not well suited to a traditional proscenium concert situation. The fact that the Vicon Real-time Engine is a TCP/IP server, however, means that the animation software, MCM, and any other performance components need not be in the same location as the Vicon 8 system. The performer may be in a remote location, and be seen by the audience only in the form of an animated, musical avatar. This presents opportunity for new formats of performance and interactivity.

# 7. FUTURE USES

The MCM project provides easy mapping of motion capture data to musical control. This will give Vicon users in the field of animation the ability to experiment with musical soundtracks ideas that are directly generated by the same data as is driving the animation. MCM provides researchers working on problems of gesture detection and recognition in the UCI Motion Capture Studio with a modular set of mapping tools into which they can interject motion analysis components. MCM makes the Vicon system into an instrument for musical expression, and gives the composer access to a remarkable affluence of simultaneous control data all coming from the bodily motion of a single performer with no physical interface.

# 8. ACKNOWLEDGMENTS

This project has been supported by the facilities of the Realtime Experimental Audio Laboratory and the Motion Capture Studio at the University of California, Irvine. Vicon Motion Systems has provided us with their RTemulator software to assist us in this software development. We acknowledge the contribution of Lisa Naugle to the original design, the work of programmers Cayci Suitt and Gene Wie in writing the design specifications with valuable advice and supervision by Andre van der Hoek, Ms. Suitt for programming the original mapping interface, and programmers Maybelle Tan and Mark Magpayo for their work in completing the mapping and scheduling software for PC.

# 9. REFERENCES


[1] Vicon 8 motion capture system, http://www.vicon.com/entertainment/technology/v8.html

[2] Max/MSP programming environment. http://www.cycling74.com/products/maxmsp.html

[3] Bevilacqua, F., J.Ridenour, and D. Cuccia, "Mapping Music to Gesture: A study using 3D motion capture data", Proceedings of the Workshop/Symposium on Sensing and Input for Media-centric Systems, Santa Barbara CA, 2002.

[4] Vicon Real-time motion capture system, http://www.vicon.com/main/technology/realtime.html

[5] Suitt, C. and G. Wie, MCM Design Document, http://www.solscope.com/~gwie/ics125/mcmdes06052001.pdf

[6] Dobrian, C., "Aesthetic Considerations in the Use of 'Virtual' Music Instruments" Proceedings of the Workshop on Current Research Directions in Computer Music, Institut Universitari de l'Audiovisual, Universitat Pompeu Fabra, Barcelona, Spain, 2001.

[7] Wanderley, M. and M. Battier, eds., Trends in Gestural Control of Music, Paris: IRCAM - Centre Pompidou, 2000.


[1] A demonstration of MCM, including audio-visual examples and explanations, is available online at http://music.arts.uci.edu/dobrian/motioncapture/

# Why Always Versatile?: Dynamically Customizable Musical Instruments Facilitate Expressive Performances

Kazushi Nishimoto
Japan Advanced Institute of Science and Technology
/ PRESTO, JST
/ ATR Media Information Science Laboratories
1-1, Asahidai, Tatsunokuchi, Nomi,
Ishikawa, 923-1292, Japan
+81 761 51 1812
knishi@jaist.ac.jp

Chika Oshima and Yohei Miyagawa
Japan Advanced Institute of Science and Technology
1-1, Asahidai, Tatsunokuchi, Nomi,
Ishikawa, 923-1292, Japan
+81 761 51 1812
{cooshi, y-miyaga}@jaist.ac.jp

## ABSTRACT
In this paper, we discuss a design principle for the musical instruments that are useful for both novices and professional musicians and that facilitate musically rich expression. We believe that the versatility of conventional musical instruments causes difficulty in performance. By dynamically specializing a musical instrument for performing a specific (genre of) piece, the musical instrument could become more useful for performing the piece and facilitates expressive performance. Based on this idea, we developed two new types of musical instruments, i.e., a "given-melody-based musical instrument" and a "harmonic-function-based musical instrument." From the experimental results using two prototypes, we demonstrate the efficiency of the design principle.

## Keywords
Musical instruments, expression, design principle, degree of freedom, dynamic specialization

## 1. INTRODUCTION
Musical instruments are tools for expressing our inner musical emotion. The more easily, directly, freely and perfectly we can express our musical emotion through a musical instrument, the more desirable the musical instrument is, not only for novices who have never or seldom performed any musical instruments but also for professionals. However, it is actually very difficult for us, in particular for novices, to express our musical emotion by using conventional musical instruments. Even after long and hard practice, we often cannot achieve satisfactory performances.

We think that this problem arises from the versatility of the conventional musical instruments. The conventional (acoustic) musical instruments are independent of the musical pieces to be performed while using them. Therefore, people can enjoy performing any musical piece of any genre with the identical musical instrument. Although this feature greatly benefits people, the wide applicability of conventional musical instruments requires a large degree of freedom in operation and generality. The large degree of freedom makes it unnecessarily difficult for the performer to perform a piece of music.

Therefore, it is necessary to reduce or eliminate the excessive degree of freedom in operation and to specialize the musical instruments for performing a specific piece. Now, we have high-quality sound synthesizers and PCs. These allow us to create new electrical musical instruments that we can freely, easily, quickly and dynamically customize as the occasion demands. By appropriately reducing the degree of freedom in operation and by specializing for a certain piece, such new musical instruments facilitate expressive performance not only for novices but also for professionals.

In section 2, we discuss the requirements of a musical instrument that facilitates expressive performances as well as how the excessive degree of freedom in operation should be reduced or eliminated and how the musical instrument should be specialized for a certain piece. Sections 3 and 4 describe two new types of musical instruments, i.e., a "given-melody-based musical instrument" and a "harmonic-function-based musical instrument," and we illustrate two prototype musical instruments, named the "Coloring-in Piano (CiP)" for typical classical music and "RhyMe" for Be-Bop style jazz improvisation. We also show experiments and their results and evaluate the advantages of the proposed new musical instruments in these sections. Section 5 compares our approach with the related works. Section 6 concludes this paper.

## 2. A DESIGN PRINCIPLE

### 2.1 Requirements
A musical instrument that facilitates expressive performances for everybody should satisfy the following requirements:

1. the initial barrier is low,
2. there is enough room for improvement by practice and study, and the improvement can be recognized by the player, and
3. the ultimately achievable quality of performance is not inferior to that by conventional musical instruments.

In order for novices to readily enjoy performing music, the initial barrier must be sufficiently low. If it is too high, most of the novices, in particular adult novices, would inevitably give up the idea of enjoying musical performance from the beginning. However, it is not practical to create a musical instrument with which a performer can immediately perform his/her ideal performance without any practice and/or study. This is because direct extraction of the performer's idea from his/her brain is impossible, and an established ideal performance does not initially exist in the performer's mind but is created through practice and/or study[1]. Accordingly, the room for improvement by practice and study is necessary. Additionally, from the novices' viewpoint, they would immediately lose interest in playing it if they cannot improve or cannot feel improvement in performance even after practicing hard. A feeling of steady and recognizable improvement toward ultimate excellent performances motivates players to continue practicing and performing. However, even if a musical instrument satisfies the above two requirements, professional performers would never use it if the ultimately achievable quality of performance were inferior to that by a conventional musical instrument.

## 2.2 How to satisfy the requirements

We think that the key to achieving musical instruments that facilitate expressive performances is dynamic customizability. When a performer performs a Thelonious Monk jazz piece with a musical instrument, the musical instrument should be convenient for performing it but need not be convenient for performing a classical Bach piece, and vice versa. Therefore, the musical instrument must be able to be dynamically, easily and quickly customized depending on what is performed with it. There are two aspects for customizing a musical instrument, i.e., reduction of unnecessary degree of freedom in operation and specialization of an interface depending on the piece to be performed.

### *2.2.1 Reduction of unnecessary degree of freedom*

There are many musical elements, e.g., pitch, timbre, volume and rhythm, that should be controlled in a musical performance. Most of the conventional musical instruments allow the performer to control all of the musical elements. However, such a huge degree of freedom is not always necessary. We can find that some musical elements require no (or less) degree of freedom when performing specific (genre of) pieces. For example, when performing "Fantasie Impromptu Op. 66" by F. Chopin, no degree of freedom in selection of pitch is allowed for the performer. The pitches of all of the notes are *a-priori* decided by Chopin. Therefore, when performing this piece, the degree of freedom in selection of pitch is redundant for the performer. The performer cannot express his/her own musical emotion in reproduction of the sequence of pitches specified in the score. However, the performer cannot skip this task. Furthermore, the task must be accurately executed because even a miss-touch is not allowed. Thus, such a redundant degree of freedom wastes the performer's cognitive and physical abilities.

What the performers of Chopin's piece should essentially do (and should devote themselves to) is expression of their musical emotion that lies beyond the reproduction task. If the performers could skip the reproduction task and directly tackle expression, they could concentrate their cognitive ability on expression. Consequently, if we could eliminate or reduce the unnecessary degree of freedom, the performer would be able to perform more expressively, and hence professionals as well as novices could receive some benefit.

### *2.2.2 Specializing an interface for pieces*

The layout of notes on the interface of a conventional musical instrument is based on the pitch of a note. That is, a certain pitch is always mapped on a certain position of the interface. For example, a C4 note is always mapped on the 24th white key from the leftmost key of a piano. We call this way of layout "pitch-based note mapping," and a musical instrument that employs it a "pitch-based musical instrument." Though this is a simple and intuitive mapping criterion, it is not always the best way of mapping. If the degree of freedom in selection of pitch is reduced when performing a specific (genre of) piece, the pitch-based note mapping actually becomes nonsense. Even if the degree of freedom is not reduced for any musical elements, some different mapping approaches often facilitate performance. In such a case, the layout should be changed based on another mapping criterion. However, it is indispensable to provide a comprehensible, consistent and definite criterion for mapping notes on the interface for the performer to precisely project images in his/her mind to externalized music.

## 3. GIVEN-MELODY-BASED MUSICAL INSTRUMENT

This section illustrates a "given-melody-based musical instrument" for performing musical pieces that require accurate reproduction of given scores, e.g., typical classical music, as the first example of a musical instrument based on the above design principle.

## 3.1 Basic Concept

The musical pieces that require accurate reproduction of given scores involve two types of elements from the performer's perspective: non-expressive elements and expressive elements. The pitches, pitch sequence and basic rhythm (that is, the time value of each note) are the "non-expressive elements." The performers must accurately reproduce them as the composer directed and hence they cannot demonstrate their expression. Therefore, control of the non-expressive elements is not an essential task for the performers, although they cannot skip this task when using conventional musical instruments. However, each note has many other attributes, e.g., *Dynamik* (varying and contrasting degrees of intensity or loudness in musical tones) and *Agogik* (a slight deviation from the main rhythm and/or the directed time values for accentuation purposes). We call these attributes "expressive elements." An individual performer's expression is reflected in how these expressive elements are controlled. Therefore, the control of the expressive elements is the essential task for the performers of this type of musical piece.

Consequently, in order to facilitate the performance of this type of music, we should reduce the degree of freedom in the non-expressive elements as much as possible. In this type of music, which notes must be performed at every point in a piece are definitely decided. Therefore, only the necessary pitches should always be mapped on the interface, and the mapped notes should dynamically change as the performance of the piece progresses. As a result, the performer can output the necessary notes without looking for them on the interface.

Owing to this way of mapping notes, the performer is freed from most of the nonessential tasks, and his/her cognitive and physical loads are alleviated. Accordingly, the initial barrier becomes low. On the other hand, the performer is still in charge of controlling all of the expressive elements for adding his/her original expression. Namely, the degree of freedom for expression is not reduced at all. Therefore, the given-melody-based musical instrument is as expressive as conventional musical instruments and provides much room for improvement of musical expression.

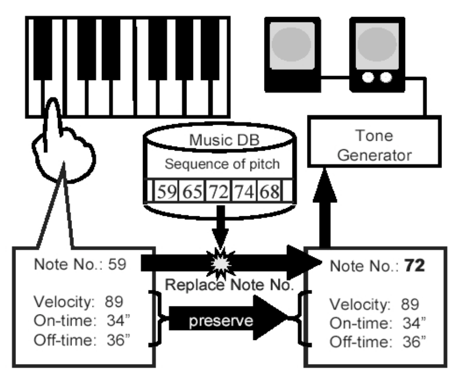



**Figure 1: Setup of CiP**

## 3.2 Coloring-in Piano

Here, we describe a prototype given-melody-based musical instrument named "Coloring-in Piano (CiP)."

### *3.2.1 Setup of CiP*

Figure 1 shows the setup of CiP. CiP consists of a MIDI (Musical Instrument Digital Interface) keyboard, a music-database, a function for replacing note numbers, and a tone-generator. Before performing, it is necessary to prepare a sequence of MIDI note-numbers (corresponding to pitches) of the piece to be performed in the music-database. While performing, the replacing function replaces the played note-numbers with the note-numbers registered in the music-database, based on the order in which they were input. Accordingly, the correct note number is always output by touching any key. On the other hand, the expressive elements, i.e., note-on (key down) velocity, note-off (key up) velocity, onset/offset timing, and pedal messages, are output as the performer plays. Finally, the replaced pitch numbers are input into the sound generator with the expressive elements preserved as they were performed. We implemented the above system on a laptop PC (OS: Windows 2000) using Delphi 6. We used a YAMAHA silent grand piano C5 professional model that outputs MIDI note-on/off, and pedal control messages. The piano was connected to the laptop PC.

### *3.2.2 Experiment to Evaluate CiP's Expressiveness*

This section describes experiments conducted to evaluate the potential expressiveness of CiP. In addition, we discuss how the interface of CiP should be designed.

#### *3.2.2.1 Method of evaluations*

The second author of this paper, who is a professional piano teacher, performed parts of two pieces on the conventional piano and on CiP. One of the pieces was "Tendre Fleur," which is one of the 25 Leichte etuden Op. 100 by F. Burgmuller. We called it "Piece-A." The other piece was "Grande Polonaise Brillante Op. 22" by F. Chopin, which was called "Piece-B." Both are examples of the style known as romanticism and include various articulations. Figures 2 and 3 show eight bars selected from each piece. She played only the melody without accompaniment. In the CiP case, the pieces were performed three ways, e.g., using only one finger for one key performance (CiP-1), using only two fingers for two-key performance (CiP-2), and using five fingers for all-key performance (CiP-5). All performances were recorded.

We asked twenty subjects who are experienced in piano playing, e.g., those who had finished the Bayer Manual, to evaluate the recorded performances. We let the subjects listen to four pairs of performances, i.e., pairs of a performance on a conventional piano and CiP-1, CiP-2, CiP-5 or the performance on the conventional piano. All of the evaluations were conducted under blind conditions. Therefore, the subjects did not know how a performance was recorded or which performances they were comparing even when they listened to the pair of the same performances on the conventional piano. We asked them to evaluate each performance from the perspective of whether it is musical (1: not musical to 5: very musical), where we explained that "musical" means "interesting" or/and "comfortable."

#### *3.2.2.2 Analysis of performance data*

Based on the performance data in the MIDI format, we calculated inter-onset interval (IOI) and gap time. The IOI is obtained as

$$IOI_i = t_{Non(i+1)} - t_{Non(i)}, \quad (1)$$

where $IOI_i$ is the $i$-th IOI, and $t_{Non(i)}$ is emitted time of the $i$-th note-on message $N_{on(i)}$. The gap time is obtained as

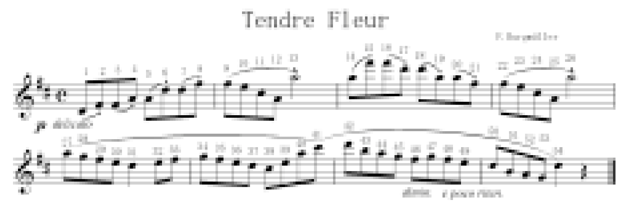



**Figure 2:** Bars 1-8 of "Tendre Fleur," which is one of the 25 Leichte Etuden Op. 100 by A. Burgmuller


**Figure 3:** Bars 220-227 of "Grande Polonaise Brillante Op. 22" by F. Chopin

$$gap_i = t_{Non(i+1)} - t_{Noff(i)}, \quad (2)$$

where $gap_i$ is the $i$-th gap time, and $t_{Noff(i)}$ is the emitted time of the $i$-th note-off message $N_{off(i)}$. Hence, if $gap_i$ is positive, the performer shortened the $i$-th note. Additionally, we extracted the velocity values included in the MIDI note-on message. The velocity of a note-on message shows the velocity of stepping down a key and nearly corresponds to the sound level of the note.

**Table 1: Average values of evaluations of musicality. An asterisk (*) indicates a significant difference of 1%.**

| Piece A | | | | Piece B | | | |
|---|---|---|---|---|---|---|---|
| | conventional | CiP-1 | t-value | | conventional | CiP-1 | t-value |
| 1 | 3.17 | 2.00 | 5.63* | 1 | 2.92 | 1.92 | 3.63* |
| | conventional | CiP-2 | t-value | | conventional | CiP-2 | t-value |
| 2 | 3.50 | 3.67 | 0.46 | 2 | 2.67 | 3.00 | 0.84 |
| | conventional | CiP-5 | t-value | | conventional | CiP-5 | t-value |
| 3 | 3.58 | 3.41 | 0.62 | 3 | 3.25 | 3.50 | 1.00 |
| | conventional | conventional | t-value | | conventional | conventional | t-value |
| 4 | 3.50 | 3.33 | 1.00 | 4 | 3.58 | 3.58 | 0.00 |

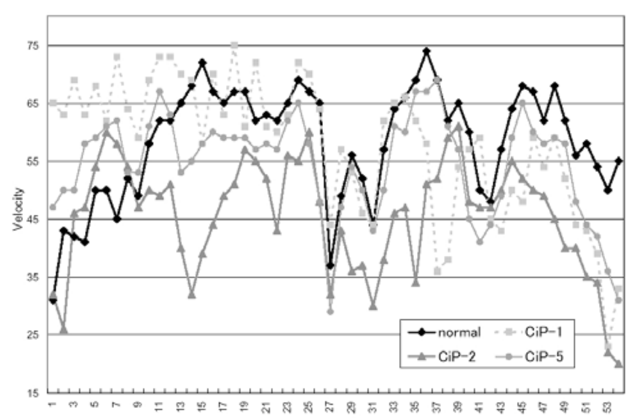



**Figure 4: Transition of average note-on velocity values of the four ways of performance of Piece-A.**

### *3.2.3 Results*

#### *3.2.3.1 Evaluation by the subjects*

Table 1 shows the average evaluation values of musicality by the twenty subjects. The reason why the values of "conventional" for each piece are different is that all of the evaluations were conducted under blind conditions. The results of a t-test indicate that CiP-2 and 5 performances are as musical as that of the conventional piano performance, though the musicality of the CiP-1 performance is significantly worse than that of the conventional piano performance.

#### *3.2.3.2 Performance data*

The *IOI*s of the four performance methods (CiP-1, CiP-2, CiP-5 and a conventional piano) are very similar for both pieces. Therefore, the performer could reproduce the basic structures of the pieces by any of the musical instruments. However, we found evident differences in gap time and velocity data. The gap time of CiP-1 was always positive. When performing CiP-1, the performer must always release the key. This means that the performer cannot perform a *legato* expression with CiP-1. Figure 4 shows the transition of the average note-on velocity values of the four performance methods for Piece-A. It is clear that the velocity of the CiP-2 performance decreases remarkably around the 14th note comparing to the other performances.

### *3.2.4 Discussion*

The results shown in Table 1 indicate that CiP, except for CiP-1, have enough potential for rich musical expression and are not inferior to the conventional piano. The reason why CiP-1 is inferior to the others is that CiP-1 reduced the degree of freedom in a necessary element, i.e., the overlap time between two consecutive notes, which is preserved in CiP-2 and 5. However, from the result shown in Figs. 4, we found that the pieces were expressed differently between, in particular, CiP-2 performances and the conventional piano performances. This might derive from the differences in fingering between two-finger use and five-finger use. For example, there is quite a large pitch gap between notes No. 12 and No. 13 in Piece-A (see Fig. 3). While the performer’s hand had to move a long way to the right when performing the part on the conventional piano, her hand did not need to move so far when performing the part on CiP-2 (only one-key distance). This motion difference should have affected the difference between the expressions (velocity, in particular).

We cannot easily conclude which expression is better. Please note that CiP is not a subspecies of the conventional piano but a new musical instrument. Therefore, CiP does not need to have the same expressiveness as that of the conventional piano. In fact, CiP-2 may permit novel expression that cannot be achieved by the conventional piano: CiP-2 always allows *legato* expression, however large the difference in pitch between consecutive notes is.

## 4. HARMONIC-FUNCTION-BASED MUSICAL INSTRUMENT

This section describes a “harmonic-function-based musical instrument” for performing musical pieces that require harmonic analysis of chord progression while performing a given piece, e.g., improvisational performance in the Be-Bop style of jazz music.

### 4.1 Basic Concept

In this section, we describe the basic concept of the harmonic-function-based musical instrument by using as an example the improvisational performance in Be-Bop style jazz (hereafter, simply called “jazz improvisation”). In contrast with the performance of classical music, the performer is required to concurrently execute two different tasks in jazz improvisation: composing melodies based on a given chord progression and externalizing the composed melodies while adding expression with a musical instrument.

Although it seems that the entire former task, i.e., composing melodies, is expressive, we think that there are still non-expressive sub-tasks involved. When composing melodies in jazz improvisation, a performer 1) reads the chord progression from a score, 2) analyzes it based on a harmonic theory, 3) obtains a function of each note, 4) chooses notes having functions that are suitable for the performer's desired expressions, and 5) finally composes melodies by concatenating the chosen notes with suitable rhythm. In other words, the performer translates the attribute of each note from “pitch” to “harmonic function” through theoretical analysis (steps 1 to 3) and then composes melodies based on the functions of the notes, not on the pitches of the notes (steps 4 and 5). In these five steps, we can say that the performer’s expressiveness is reflected only in steps 4 and 5. Though steps 1 to 3 can be executed mechanically based on established theory, these steps place a very high cognitive load on the performer. As a result, novices, in particular, cannot have the extra cognitive ability to compose good melodies.

In the externalization of composed melodies, the performer must look for notes based on their harmonic functions, not their pitch, on a musical instrument interface. Therefore, the notes should be mapped on the interface based on their harmonic functions. Each of the twelve notes in an octave has a different function in a certain harmony. By providing the twelve positions in an octave on the interface of a musical instrument, by assigning a specific function to a specific position, and by constantly mapping a note with a specific function to the corresponding position, we can construct a new musical instrument specialized for jazz improvisation. By operating a certain position, the performer can always immediately obtain a note with the required function, though the pitch of the obtained note changes along with the chord progression. Thus, even a novice can directly tackle the expressive tasks by skipping these non-expressive tasks, and a professional can concentrate more of his/her ability on expressive tasks.

### 4.2 RhyMe

Here, we describe a prototype harmonic-function-based musical instrument named “RhyMe.”

### *4.2.1 Setup of RhyMe*

Figure 5 shows the setup of RhyMe. The system consists of two modules: a chord-progression analysis module and a dynamic mapping module. The chord-progression analysis module analyzes the chord progression of a musical piece based on the Berklee theory, which was developed at the Berklee College of Music and is the most well known harmonic theory in Be-Bop style jazz improvisation, and then obtains available note scales for each chord. Using the obtained available note scale data, the dynamic mapping module dynamically maps the notes to an interface in the following manner. Usually, a scale consists of seven notes. Each note is named by a number relative to the position from the root note, i.e., I, II, III, IV, V, VI and VII, where I is the root note. These positions correspond to the functions of the notes. For example, the III note has the function of expressing tonality: minor or major of the chord/scale. Accordingly, we name the function of each note by the number of the note's position, e.g., the function of the III note is named “function-III.” For example, the F-mixolydian scale consists of F, G, A, B-flat, C, D, and E-flat notes. Therefore, the function-III note of this scale is A. The functions of the notes not included in the currently available note scale are also named based on the number of scale-notes, e.g., function-flat-II and function-sharp-IV. Finally, a note of a certain function is constantly mapped to a corresponding position on the interface. For instance, the note of function-III is always mapped to the position of function-III. As a result, all of the notes are mapped on the interface based on their harmonic functions. The mapping dynamically changes as the chord progresses (i.e., the available note scale changes). Moreover, we can choice notes based on their intervals by this way of mapping. For instance, if we want a note that is two degree higher than the function-II note, we can get it by the position of function-IV. Thus, the mapping of notes is specialized for Be-bop improvisation, while RhyMe does not reduce the degree of freedom of any musical element.

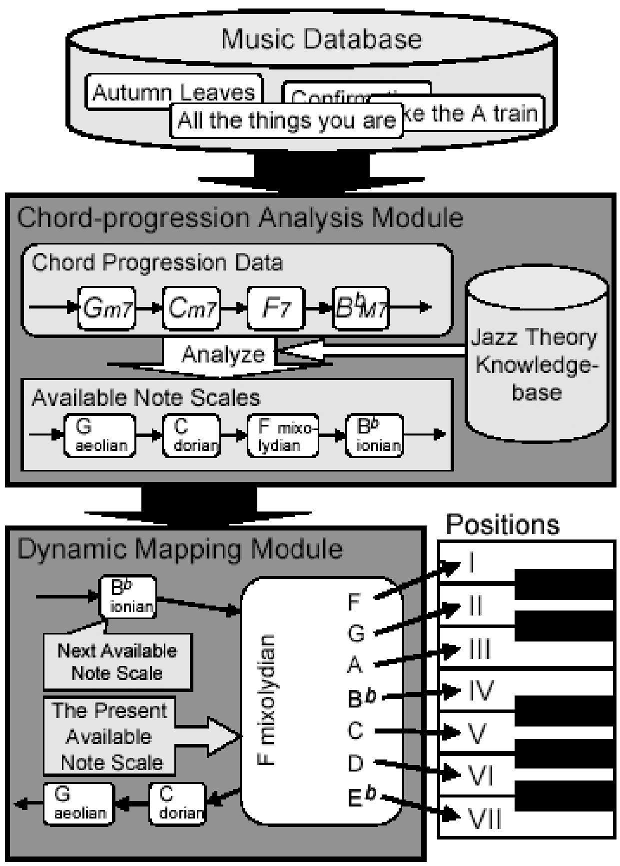



**Figure 5: Setup of RhyMe**

In the current implementation, we applied a MIDI keyboard (KORG M1) as the interface. Functions I to VII were assigned to the C to B keys, respectively. The out-of-scale functions were mapped on the black keys. In this prototype, a note that is a semi-tone lower than the white key to the right of the black key is mapped as an expedient. Because of this mapping method for the black keys, an in-scale note is assigned to a black key in some cases.

### *4.2.2 Experiments*

We conducted an experiment using RhyMe for jazz improvisation to evaluate whether novice users could easily perform jazz improvisation. We employed nine subjects and let them play improvisations of "Autumn Leaves" using RhyMe as well as using the same keyboard in normal pitch-based note mapping mode, i.e., a conventional keyboard as it is. Before beginning the experiments, we briefly explained how RhyMe works to the subjects. After the experimental performances, we posed the following questions to the subjects:

**Table 2: Results of eight-question inquiry for improvisational performances using RhyMe and a normal keyboard**

| No. | RhyMe | Normal | t | No. | RhyMe | Normal | t |
|---|---|---|---|---|---|---|---|
| 1 | 2.94 | 2.56 | | 5 | 2.50 | 1.94 | |
| 2 | 2.50 | 1.75 | * | 6 | 3.81 | 2.50 | * |
| 3 | 3.06 | 1.94 | * | 7 | 4.25 | 3.19 | * |
| 4 | 3.81 | 2.50 | * | 8 | 3.75 | 2.25 | * |

1. Was it easy to operate this instrument? (1: Very difficult, 5: Very easy)
2. Was your performance good? (1: Very bad, 5: Very good)
3. Was your performance jazzy? (1: Not jazzy, 5: Very jazzy)
4. Did you enjoy performing this instrument? (1: No, 5: Yes)
5. Do you think you could perform as you wanted to perform? (1: No, 5: Yes)
6. Do you think you could improve your performance with practice? (1: No, 5: Yes)
7. Do you want to continue to perform on this instrument? (1: No, 5: Yes)
8. Do you think you can play a session performance with other performers using this instrument? (1: No, 5: Yes)

Table 2 shows the average values of each inquiry for RhyMe and for the normal keyboard. The asterisks in the "t" column show the results of t-tests comparing the average values in each inquiry: an asterisk indicates a significant difference of 2%. From these results, we confirmed that RhyMe was evaluated more highly than the normal keyboard for all of the questions. Furthermore, RhyMe scored significantly better than the normal keyboard for all of the questions except Nos. 1 and 5.

### *4.2.3 Discussion*

The harmonic function of notes is not such a common concept, particularly the novices. However, though the subjects did not understand the harmonic function well, they could satisfactorily perform on RhyMe. This fact proves that RhyMe does not initially require a deep understanding of the harmonic-function. Therefore, this novel way of mapping does not have a bad influence on the alleviation of the initial barrier. On the contrary, it makes it easy for the subjects to perform jazz improvisation, as the results in Table 3 show. We could not find any significant difference between RhyMe and the normal keyboard in question No. 1. We think this result relates to the fact that both musical instruments had the same interface device. Namely, it is assumed that the instrument's ease of operation is strongly dominated by its interface device. However, despite the interface problem, RhyMe was evaluated as being much better than the normal keyboard in all other questions. In particular, the high scores of RhyMe in questions 6, 7 and 8 suggest that the subjects had great expectations of their future ability to enjoy performing music with the harmonic-function-based musical instrument, though they did not expect such enjoyment with the conventional instrument.

## 5. DISCUSSION AND RELATED WORKS

Recently, various instruments for novices' entertainment have been developed. The "Two Finger Piano"[2] is a system that allows the user to coarsely handle tempo and *Dynamik* for each "beat" or for each half-beat (but not for each note) by using two fingers. Therefore, it is impossible to control *Agogik,* which requires note-level control. The CASIO LK-40 Lighted Keyboard [3] is equipped with a similar function to CiP that always only outputs available pitches of a piece by hitting any key. However, this system outputs constant velocity values. Therefore, *Dynamik* cannot be expressed by this system. "MusPlay"[4] and the "any key play" mode of the

Yamaha EZ-20 and 30 keyboards[5] are very similar to CiP. In addition, MusPlay allows us to play a two-handed polyphony performance. As for the mechanism, CiP is not so advanced from these systems. However, it is not assumed that professional musicians use these musical instruments. Therefore, no experiments have been conducted to evaluate their potential for expression. On the contrary, we focused on a mechanism that could benefit professionals as well as novices, and we proved that CiP is as expressive as conventional musical instruments in classical music performances.

The "adlib-musician" function of CASIO CT-647 keyboard is similar to RhyMe. However, in the CT-647, only notes included in the presently available note scale are mapped on the keys. In addition, the way of mapping notes is different from RhyMe: a note whose pitch is the same or the nearest neighbor of the original pitch of a key on the normal keyboard is mapped to the key. "INSPIRATION"[6] also changes input notes to theoretically correct notes automatically. Therefore, in these systems, neither pitch nor the harmonic-function of a note mapped to a certain key is always stable. Consequently, though the performer can perform improvisation as if he/she had become an experienced musician, the performer cannot intentionally compose melodies by using these systems: there always remain unexpected factors. On the other hand, RhyMe always provides an evident and stable criterion of mapping of notes: the harmonic-function-based note mapping. Therefore, the performer can intentionally compose melodies by considering the harmonic functions of notes. In addition, all twelve notes are (basically) always available on RhyMe for allowing a fully expressive and profound improvisation by using even incorrect notes (i.e., dissonant notes).

Thus, the three requirements, i.e., low initial barrier, enough room for improvement, and potential for rich expression, have not yet been satisfied in any ordinary (electric) musical instrument. However, we showed that they could be satisfied in one musical instrument by considering the degree of freedom of musical elements and specializing the interface of the musical instrument for a specific (genre of) musical piece.

Hunt et al.[7,8] showed that a multiparametric interface is more useful and expressive than a simple one-to-one mapping between each control input and each musical parameter for most people. We also empirically thought so. Therefore, we employed multi-parametric interfaces for CiP and RhyMe. However, we think that it is not always necessary to integratedly control all the musical parameters and that only "expressive elements" should be integratedly controlled. Our experimental results would support this concept. Hunt et al.[9] discussed the importance of mapping between the way of input and that of output in electric musical instruments. They focused on the importance of a kind of *affordance* of musical instruments, while we focus on the importance of an evident criterion of mapping. Chadabe[10] pointed out that the electric musical instruments have been freed from tight and fixed relationships between controller and sound generator and that the instruments should employ more flexible mapping between them. We strongly agree with this position and believe the two musical instruments described in this paper, i.e., the "given-melody-based musical instrument" and the "harmonic-function-based musical instrument," represent incarnations of this concept.

## 6. CONCLUSIONS

In this paper, we discussed a design principle of musical instruments that are useful for both novices and professional musicians and that facilitate musically rich expression. We pointed out that versatility causes difficulty in performance. By eliminating the unnecessary degree of freedom in operation and by appropriately specializing a musical instrument for performing a specific (genre of) piece, the musical instrument becomes more useful for performing the piece and facilitates musical expression. Based on the proposed principle, we developed two new types of musical instruments, i.e., a "given-melody-based musical instrument" and a "harmonic-function-based musical instrument." Using two prototypes, we demonstrated the efficiencies of the proposed principle based on the experimental results. By exploiting this principle, we can create new musical instruments that serve as an introduction to musical performance as well as a tool for exploring the entire world of expressive musical performance, without changing the instruments.

This study was supported in part by the Telecommunications Advancement Organization of Japan.

## 7. REFERENCES

[1] Schon, D. A.: *The Reflective Practitioner: How Professionals Think in Action*, Harpercollins, 1983.

[2] Ueda. K., Hirai. S., Katayose. H. and Inokuchi. S., The improvement of Two Finger Piano, IPSJ Symposium Series Vol.2000, No.4., 2000. (in Japanese)

[3] http://www.casio.com/musicalinstruments/product.cfm?section=35&market=0&product=4057

[4] http://www.cs.utk.edu/~plank/plank/music/musplay/

[5] http://www.yamaha.com/cgi-win/webcgi.exe/DsplyModel/?gPPK00005EZ30

[6] Yatsui, A. and Katayose, H.: An Accommodating Piano which augments intention of inexperienced players, *Proc. Intl. workshop on Entertainment Computing*, 2002.

[7] Hunt, A., Wanderley, M. M., and Kirk, R.: Towards a Model for Instrumental Mapping in Expert Musical Interaction, *Proc. International Computer Music Conference (ICMC2000)*, 2000.

[8] Hunt, A., and Kirk, R.: Mapping Strategies for Musical Performance, *Trends in Gestural Control of Music*, Ircam, 2000.

[9] Hunt, A., Wanderley, M. M., and Paradis, M.: The importance of parameter mapping in electronic instrument design, *Proc. The 2002 Conference on New Instruments for Musical Expression (NIME-2002)*, 2002.

[10] Chadabe, J.: The Limitations of Mapping as a Structural Descriptive in Electric Instruments, *Proc. The 2002 Conference on New Instruments for Musical Expression (NIME-2002)*, 2002.

# Block Jam: A Tangible Interface for Interactive Music

Henry Newton-Dunn
Interaction Laboratory,Sony Computer Science Laboratories, Inc.
3-14-13 Higashigotanda
Shinagawa-ku, Tokyo 141-0022, JAPAN
henry@csl.sony.co.jp

Hiroaki Nakano
Creative Development Group,Sony Design Center
6-7-35 Kitashinagawa
Shinagawa-ku, Tokyo 141-0001, JAPAN
nakano@dc.sony.co.jp

James Gibson
Human Interface Design Group,Sony Design Center Europe
The Heights, Brooklands
Weybridge , Surrey KT13 0XW, UK
gibson@dc.sony.co.jp

## ABSTRACT

In this paper, we introduce Block Jam, a Tangible User Interface that controls a dynamic polyrhythmic sequencer using 26 physical artifacts. These physical artifacts, that we call *blocks*, are a new type of input device for manipulating an interactive music system. The blocks' functional and topological statuses are tightly coupled to an ad hoc sequencer, interpreting the user's arrangement of the blocks as meaningful musical phrases and structures.

We demonstrate that we have created both a tangible and visual language that enables both the novice and musically trained users by taking advantage of both their explorative and intuitive abilities. The tangible nature of the blocks and the intuitive interface promotes face-to-face collaboration and social interaction within a single system. The principle of collaboration is further extended by linking two Block Jam systems together to create a network.

We discuss our project vision, design rational, related works, and the implementation of Block Jam prototypes.

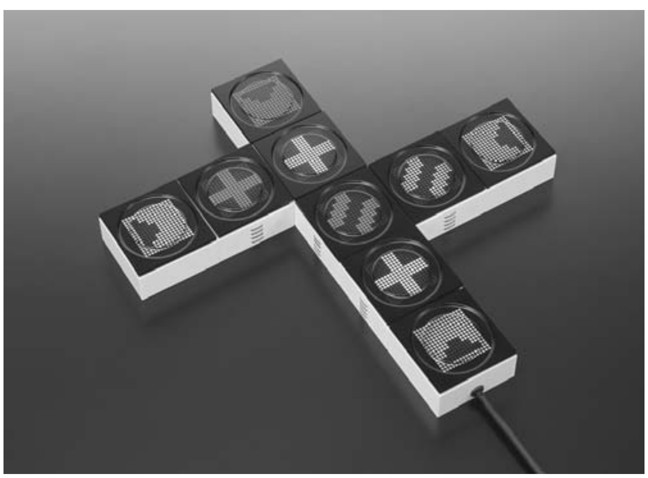


**Figure 1. A cluster of blocks, note the mother block on the bottom right**

## Keywords

Tangible interface, modular system, polyrhythmic sequencer.

## 1. VISION

We believe in a future where music will no longer be considered a linear composition, but a dynamic structure, and musical composition will extend to interaction. We also believe that through the introduction of such media the divisions of composer, performer, and audience will be blurred.

Our second aim is to put the group experience back into music. We understand that the musical experience changes with technology. Musical technology provides greater control, more possibilities and greater access to the beginner or novice.

The way we receive and listen to music is also changing, not just in terms of Low-Fi to Hi-Fi or Phono to Tape to CD to MD to MP3, but also in terms of experience, music is moving from a social experience to a personal experience, from campfire to orchestra to living room to Walkman. Degrees of separation have occurred between the composer and the performer, the performer and the audience. This trend continues. Inversely, technology is moving towards community, towards the group, towards the network.

Block Jam is the first in a number of projects aimed at addressing this disparity within the technologically mediated musical experience.

## 2. DESIGNING BLOCK JAM

This project was initially created in December 2001 as an exploration into both new types of musical experience and novel tangible interaction techniques. The project was a collaboration between designers from Sony CSL Interaction Laboratory and designers from the Sony Design Center. Decisions were weighted towards functional relevance, with a view towards an aesthetic outcome. Designing the project fell into two distinct areas, designing the musical experience and designing the tangible interface. Though separately described below – these processes were actually designed concurrently, decisions made in one area often affecting the other.

### 2.1 Designing the Musical Experience

Through a musician's physical gesture, his vocal or instrumental performance, we are able to unlock sensations, emotions and thoughts. Musicians develop these skills through formal study and practical application. Music is not difficult to appreciate or enjoy, but difficult and often frustrating to create or play, especially for the untrained.

Interactive music has the potential (and the promise) to help release us from this difficulty and frustration. In 1970, Mathews and Moore created the first interactive music program called *Groove* that proved to be experientially "almost irresistible"[[3]]. Since Groove, there have been many attempts to design and implement new computationally mediated musical systems and interfaces. It should be noted that this paper is not concerned with computer music systems, but rather systems for interacting with music.

These systems and interfaces fall into many categories; from improvisational or continuator systems [[4]], Hyperinstruments [[5]], to simpler systems that are usable by the novice. There are a host of commercial available experiences like those created by new media companies such as *Hi-Res*[17], or software titles such as *Rez* [15] or *Parapa the Rappa* [16] for the PlayStation 2 platform.

Our goal for this project was to create an interactive music system that would both engage the novice user and the musically trained, and promote collaboration and communication among the participants.

We can think of music as a communicative conduit, as a performance, and as a mono-directional expression. We prefer to consider music as bi-directional (or even omni directional) since music is about teamwork and group interaction. It is a promoter of communication, excitation, and mediation, and most importantly to us, creative interplay. This notion can be identified in the compositions of artists such as Arnold Schoenberg, Karlheinz Stockhausen and notably John Cage. Umberto Eco speaks of the Poetics of the Open Work [2], in which each layer of interpretation adds to the completeness. Therefore, he suggests that a piece of music remains incomplete (or open) until interpreted from the score by a musician, then in turn by the audience.

Bearing this in mind, we decided on two key points to be used as anchors for our design of a musical experience:

1. The collaborators have a common musical frame of reference (much like Schoenberg's twelve tone system) within which they can each assume a different role.
2. The musical frame of reference should be compos-able, and must remain open (interpretable).

These translated into an asymmetric network of interactive music applications, emulating the structure of a band (e.g. a drummer app, guitarist app, bassist app, etc).

This suggested a need for a collaborative interactive music framework. Such a framework would allow composers a new creative space for themselves and their audience – the participants - to explore.

Block Jam is the physical realization of an application that could be used for such an interactive music framework. As a starting point, we divided the potential applications into two categories – sequencer type applications and gestural type applications.

Sequencer type applications are comprised of modular musical elements that are arrange-able and interchangeable within a sequential context to create a novel musical outcome such as Toshio Iwai's Composition on the Table [[9]].

Gestural applications by contrast are based on the modulation of tone, pitch or timbre to create an expressive output. An example is Ivan Poupyrev's *Augmented Groove* [[8]], where a user modulates and mixes techno compositions by manipulating real LP records tracked by a PC using visual markers. Effects, filters, and samples are triggered according to a record's movements (up-down, rotation, tilting, and shaking).

We considered that a necessary limitation to our design was that it must be stylistically neutral. Unlike Augmented Groove, we did not want to tie the applications or interface to be organized around a particular genre, particularly if our longer-term goal is to build a compose-able interactive music framework.

For our first application, Block Jam, we chose the sequencer type. We felt that it would be less problematic to create functional mappings and far easier for it to remain neutral and open.

## 2.2 Designing the Tangible Interface

When it came to designing the interface for Block Jam we started by looking at interactive toys and sound devices currently available for children, good examples being *SoundBlocks*, *Musini* and *Phonics Tiles* available from Neurosmith[[6]] (who produce children's products based on research into linguistics and cognitive science).

Children's toys tend to be physically organized and actuated, have meaningful use of shape and color, are iconic and often include sound. Children's toys also have immediacy. These were all characteristics we hoped to include and emulate in our design, the major difference being that our target users were adults, who require a much greater sophistication than children in order to be experientially engaged.

Other toys of particular note were Friedrich Froebel's Gifts (#2 to #6), which are sets of wooden blocks that vary in complexity, shape and color. The sets were designed to help children learn, explore, and create. The shapes, being primitives, hold no direct meaning, but when combined they can create endless iconic forms such as houses, castles, towers and bridges. The forms act as a mechanism for eliciting experience. This suggested the possibility of a programmatic equivalent - a modular tangible interface whose rules are simple enough to be easily understood, but whose outcome is potentially complex enough to be continually engaging. Perhaps we could use blocks to program sound.

The idea of using modular tangible blocks for building programmatic structures is not new. We formed two approximate categories for our assessment of tangible interfaces, functionally heterogeneous and functionally homogeneous. Functionally heterogeneous tangible interfaces are where different physical artifacts are used to represent different functions. Functionally homogeneous tangible interfaces consist of a single type of physical artifact with a single function – typically, many of these artifacts could be interlocked to produce programmatic outcomes.

### *2.2.1 Examples of Functionally Heterogeneous Modular Tangible Interfaces*

FitzMaurice identified the possibility of using tangible objects (Graspable User Interface) in *Bricks* [1] to extend the Graphical User Interface (GUI). Different tangible artifacts (as metaphoric transducers) could be used to represent functional elements of a GUI. This promoted the notion of a *space-multiplexed* interface, as opposed to a *time-multiplexed* interface (such as the mouse).

Brygg Ullmer's *MediaBlocks* [10] further extended this work by using tangible objects for the containment, transport and manipulation of a digital media system. Wooden blocks known as *phicons* (physical icons) were used "as a seamless gateway" to control the GUI.

Jun Rekimoto's *DataTiles* [[12]] integrated both the graphical and the physical user interface. It promoted the use of tagged transparent objects as interaction modules, which mixed visual feedback with physical interactions and created a physical language for combining multiple tiles to create a "sentence". This system relied on screen used as a base on which the transparent tiles were placed.

Mitchel Resnick's *Behavior Construction Kit* [7], which allowed children to build behavioral machines using sensor bricks (e.g. IR) and output bricks (a motor) connected to a programmable brick. Interestingly the project was envisioned as a means of making ubiquitous computing accessible to children, but was later realized as Lego Mindstorms, which is sold as a "Robotics Invention System"[[11]].

### *2.2.2 Examples of Functionally Homogeneous Modular Tangible Interfaces*

Frazer, J.H. et al identified the first modular tangible systems in 1982 [[13]] with two systems, *Tree Searcher* and notably *Intelligent Beer Mats*. The Intelligent Beer Mats were used to describe 2D topological configurations of flat square physical units to a host PC. The system was intended as a prospective application for the architectural building process.

Mathew Gorbet's Triangles [[14]], similar to the Intelligent Beer Mats consisted of a set of identical flat, plastic equilateral triangles, each with a microprocessor and unique ID, again, allowing a host PC to rebuild a given topology. The triangular shape was chosen because it held no meaning, thus the Triangles were applied to varying media oriented applications. Unfortunately, the Triangles were limited by a lack of an integrated output and input interface

## 2.3 Interface Requirements for Block Jam

Many of the tangible interfaces that we looked at were:

1. Reliant on a separate GUI [1, 9,12]
2. Had stateless interface artifacts, (they just acted as passive handles) [1, 9, 13, 14]
3. Had no, or limited means of direct digital interaction with an artifact itself [1, 9, 13, 14]

A clear advantage of a digital artifact is that it can be functionally dynamic. Our primary requirement for the Block Jam interface was that each tangible artifact would have a dynamically change-able state. This in turn suggested a need for a means of displaying the state, and a means of changing it.

Early sketches included a large variety of possible functions/states [Figure 2]. These were grouped and then mapped to a variety of shapes, to create a visual/shape interaction language. The language included different types of input mechanisms and function groups.

The input mechanisms included a variety of state and stateless dials, wheels, buttons, and sliders. The functions were grouped into effect (DSP) modulators, route functions – including route splitters and route mergers, sound triggering functions, and more compositionally algorithmic functions, such as automatic key changing.

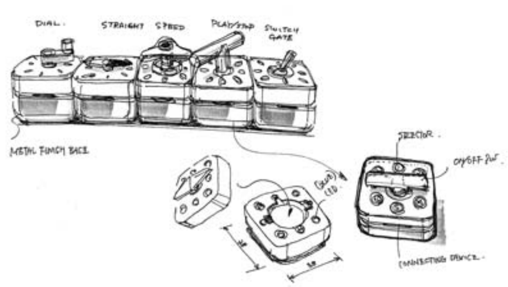



**Figure 2. Early sketch (describing a more complex interaction language)**

However, it soon became apparent that the potential complexity of assigning separate functions to separate objects could become overwhelming for the user within the context of our design. We found that designing an interface that makes music easily controllable at a higher level did not necessarily mean adding new functionality (as is normally the case with interface design), but rather it meant refining and rationalizing existing functionality.

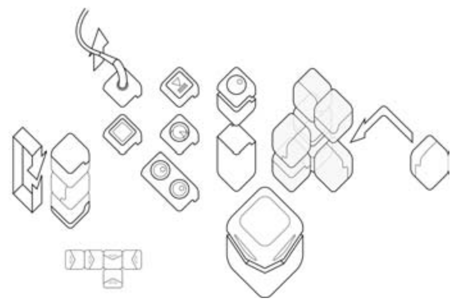


**Figure 3. Later sketch (exploring the ramifications of stacking blocks in 3D)**

We started a continuous process of removing functionality, attempting to boil it down to the absolute bare minimum. During this process, we realized that the tangible artifacts' different functional states, did not have to be mapped to different shapes. We had so few functions left that the original shape groupings became meaningless. Each artifact's state or function could be represented by its display mechanism alone. One primitive shape with a common set of input and output mechanisms could be used.

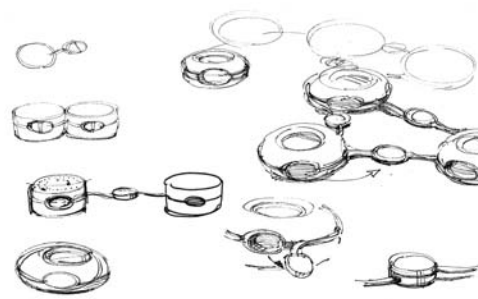


**Figure 4. Sketch of a non-tessellating structure**

We explored tessellating and non-tessellating structures and different stacking, and layering solutions. Tessellating shapes were attractive because they're easy to assemble, input mechanisms are accessible, and connected shapes could be easily read. However, they're best suited to 2 dimensional arrangements – 3 dimensional arrangements (though exciting) had problems of visual and input mechanism occlusion [Figure 3]. Conversely, non-tessellating structures [Figure 4] were equally suited to 2D and 3D, especially when self organizing [Figure 5], but suffered from bad input mechanism accessibility and poor shape readability.

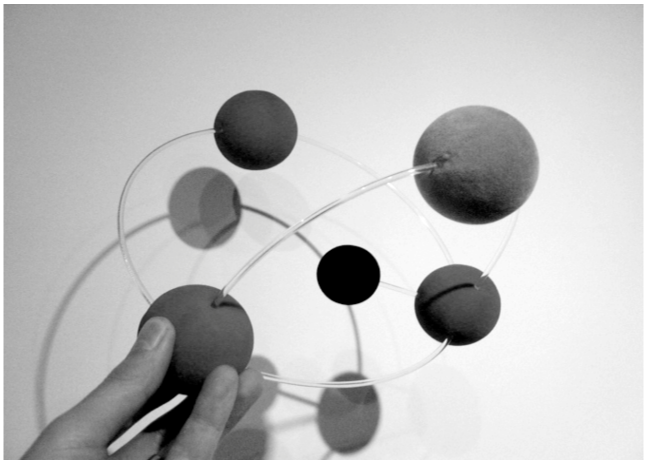


**Figure 5. Mock-up of a self organizing structure using plastic rods to hold the nodes in tension**

Finally, we chose a form factor based on a square shape rather than a triangular, or a circular shape because it implied directionality. A message arriving at one side could easily be imagined passing through to the opposite side. A 3-color LED matrix (16*16) was chosen as the display mechanism for its simplicity, and two input mechanism were added, a button for toggling the state/function, and a dialing gesture for choosing sound [Figure 6].

**Figure 6. The final block design, note the milled groove in the surface and the connectors on the side**

### *2.3.1 The Anatomy of a Block*

Externally, a block consists of a white ABS resin box, with a black acrylic top [Figure 6]. It has connectors on the side, which connect power and pass serial data. The black acrylic top has a circular milled groove in it's surface, and an LED matrix below its surface that is visible through the acrylic when illuminated. The LED can light up red, green, or orange. Incidentally, the black acrylic was chosen because it helps with the LED visibility.

The blocks can sense two types of input, a click, and a dialing gesture. The click is measured via a simple sprung button placed on the under side of the circuit/LED matrix, so, when the acrylic top is pressed so the button is pushed. The dialing gesture is sensed via an array of eight infrared optical reflectors arranged two to a side along the path of the milled groove. As the user dials, his/her finger is guided by the groove.

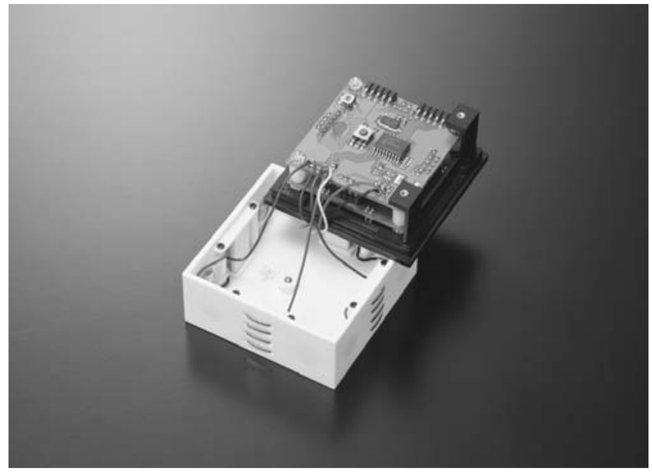


**Figure 7. Inside a block, note the button for detecting the click**

The blocks physically connect to each other by means of two hidden magnets on each side of the box. Each block has a unique ID, allowing the blocks to communicate over a common bus. Data is sent from a block when it is connected, when a neighbor has been disconnected and when a block has sensed a user's interactions. Information received updates a block's status and tells it which icon to display on the LED matrix.

The data passing between the blocks and the PC is handled by a PIC microcontroller [Figure 9] in each block communicating over a common bus. The LED matrix, the click input and the array of optical reflectors used to detect the dialing gesture were all handled by second PIC microcontroller. The only calculation performed by the blocks was the dialing to keep the data flow on the common bus to a minimum (otherwise risking too much noise). All other functions were handled by the controlling PC, which told the blocks what to display and when to display it.

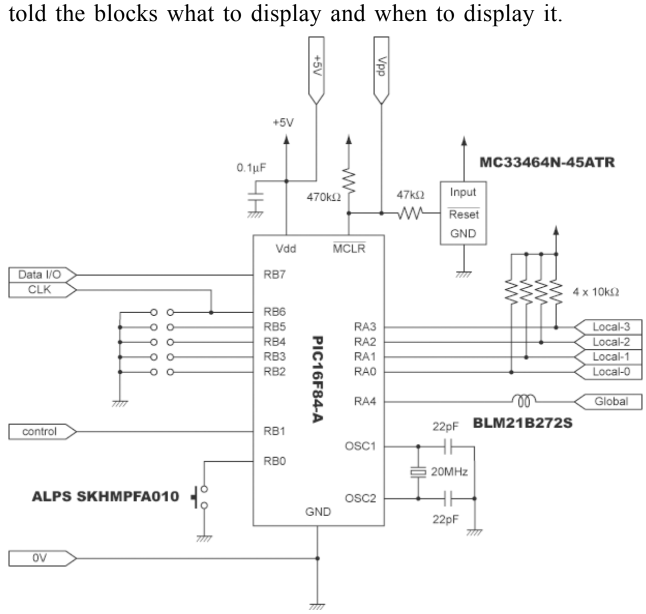



**Figure 8. Schematic showing a blocks I/O system**

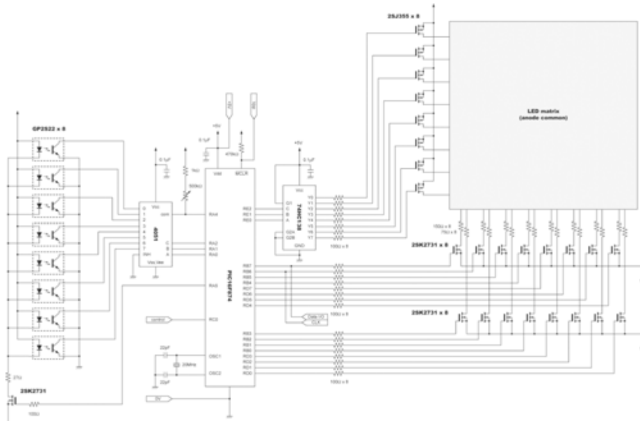


**Figure 9. Schematic showing the input system (click and dial) and the LED matrix control**

# 3. INTERACTING WITH THE BLOCKS

We created two types of block, *play blocks* and *path blocks*. Play blocks start, stop and control the speed of a sequential instance or marker known as a *cue ball*. Path blocks control the route that a cue ball travels. A play block can only be connected on one side (towards which the play icon is pointing), where as a path block can be connected on four sides.

As the blocks are added together, they form a *cluster* [figure 1]. A cluster is connected to a PC (for computation only) by a tethered play block (known as the *mother block*). The tether (wire) provides common power and a common serial connection to the cluster.

A cue ball *bounces* from one block to another within a cluster according to rules determined by the state of each path block. Every block metaphorically contains a sound, so as a cue ball (or cue balls) bounce from block to block it determines a sequential composition, creating music.

## 3.1 Starting and Stopping a Cue Ball

Clicking a play block toggles a cue ball instance on or off. Therefore one click starts a cue ball and another stops it (in addition to toggling, a timer is measuring the length of the clicking action). This measurement allows for different speeds of the cue ball to be started. A quick click starts a quick cue ball, a medium length click starts a medium speed cue ball, and a slow click starts a slow cue ball (the amount of time a click is held down while stopping a sequence has no functional relevance – the cue ball just stops).

Since events in our system are quantized to a fixed beat/grid, this translated as a click of less than one second starting a cue ball that bounced every beat. A click that lasted from 1 to 2 seconds launched a cue ball that bounced every two beats. A click longer than 2 seconds launched a cue ball that bounced every four beats.

Multiple play blocks can be added to a cluster at the same time, and so multiple cue balls can run concurrently. Experientially, concurrent cue balls, especially when they are of different speeds (e.g. two slow and one fast cue ball), create musical layering and complexity. A play block only controls the cue ball that it has launched.

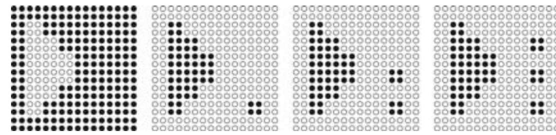


**Figure 10. Play Icons displayed on the 16*16 LED matrix**

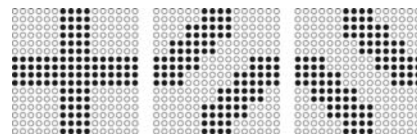


**Figure 11. The straight and corner function icons**

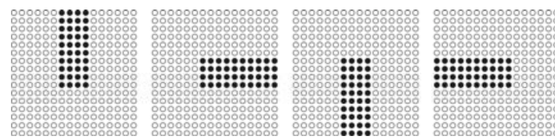


**Figure 12. The four rotated states of the gate icon**

## 3.2 Controlling the Path of a Sequence

Clicking a path block toggles between various path changing functions. The path changing functions determine which side of the block a cue ball will exit from relative to the side through which it entered. The path changing functions are:

1. The *straight* function, which bounces the cue ball straight along the given axis
2. The *corner* function, which bounces the cue ball at 90° relative to the given axis
3. The *gate* function, which jumps the cue ball in the direction indicated by a changing icon displayed on the LED surface, the graphic rotates 90° every time it is triggered

When a cue ball comes to the end of a route, i.e. it has nowhere to jump next, the cue ball loops back to the play block from which it originated. If a cue ball bounces onto a play block, its route is reflected 180°, returning in the opposite direction.

The corner function was often used to build large loops, but they could also be used as a means of isolating one area from another. For example in Figure 13, a cue ball launched from the top red play block would never reach the bottom two green blocks.

The gate function was designed to act as a type of counter, sending a cue ball in a given direction every fourth count. More often, it was used as a randomizer adding variation to a cue ball's route. When a large number of play blocks were gathered together and all set to the gate function, with several play blocks initiating several cue balls, the system constantly varied, never sequentially looping, much like the behavior of cellular automata. Although fascinating to watch and consider, this did not produce interesting sound. The output, though quantized, felt structure-less, and without flow or variation. It generally produced an un-engaging din when used with harmonic sounds (surprisingly), but was more palatable when we used rhythmic sounds.

## 3.3 Choosing a Sound

Milled into the top surface of a block's display is a circular groove. If a user places their forefinger into the groove and follows it rotationally, the block senses a dialing activity [Figure 13].

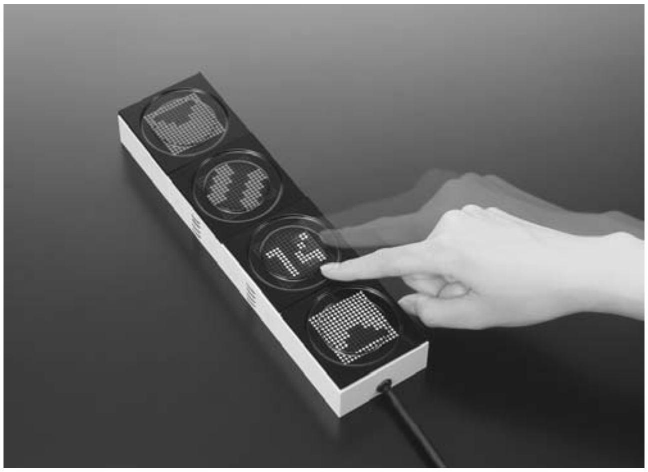



**Figure 13. Dialing a new sound**

Dialing changes the block's sound ID. As the user dials, a number representing the sound ID on the display counts up or down from 0 to 15. Dialing clockwise counts up, and dialing anticlockwise counts down. The numbers were grouped into three colors, so sound IDs 0-5 are displayed red, 6-10 are displayed orange, and 11-15 are displayed green. (More colors were possible, but we limited the number to three for easy readability)

After the user has finished dialing the chosen sound ID the block's display reverts back from the shown number to the functional icon. The icon retains the color of the sound ID.

For example, if we take a play block that has a sound ID of 1, the play icon shown on the block's display will be red. If a user then dials in a sound ID of 12, after the user has finished dialing the display reverts to the play icon, except that this time the icon is green.

The different color groups allow the user to approximately know visually which block has which sound ID. In our prototype, we mapped all the guitar and organ type sounds to the red group, vocal sounds to the orange group and percussive sounds to the green group (sound ID 0 is never mapped to a sound, it can be used as a musical pause by a user). The iconic use of color greatly assisted a users ability to keep track of the structures they had created, and which sound they had put where. This was not something that we had originally considered - our initial prototype only had red LEDs, which led to user confusion and frustration. The addition of the color groups allowed the users to play more freely, they no longer had to remember which sound was related to which ID, but instead could say "I want an orange sound here, and a green sound there".

The resulting sound that a block played was dependant on two variables:

1. The block's sound ID
2. The speed of the cue ball that triggered it.

This meant that a block doesn't just contain one sound, it actually contains three; one sound for a slow cue ball, one sound for a medium speed cue ball, and one sound for a fast cue ball. Consequentially, a single block can play one of 45 possible sounds from 15 sound IDs.

Having the combination of different cue ball speeds and sound IDs, in conjunction with the modular tangible interface, though apparently simple, allowed the user an enormous number of musical possibilities, in an intuitive, easy to use interface without any prior musical knowledge.

## 4. OVERVIEW OF THE BLOCK JAM SYSTEM

The blocks are connected to a *mother box* via the mother block. The mother box mediates serial data from the blocks to a PC. The PC is connected via MIDI to a sound module to render the output.

The software architecture on the PC comprised:

1. The topological model
2. The functional model
3. The mapping
4. The sequencer

As a block is connected to the cluster, it is activated. It sends a message to the PC telling the topological layer its unique ID and a number identifying which side (0-3) and the ID and connecting side of the adjoining block(s). From this information, a topological model of the blocks is deduced.

Attached to the topological model is a functional model. The functional model tracks all the users interactions and changes each block's state accordingly. A metronomic event triggered by the sequencer tells the functional model to send a list of output events to the mapping layer if there are one or more active cue balls. The sequencer finally sends the appropriate MIDI data to the sound module.

### 4.1 Mapping the Sound to the Sequencer

Every time a cue ball bounces from block to block, it triggers an event. The trigger event is then mapped to an array of MIDI events called a *part*.

The part is then passed to an ad hoc sequencer, which quantizes the MIDI events (according to a time stamp) and passes them to a MIDI sound module, which renders the resulting sound out put. A user experiences the sound coming from a block metaphorically. We use a part instead of playing a sample directly because it allows us a to do much more compositionally – after all, MIDI can control much more than a sampler.

If two or more cue balls trigger the same block at the same time then all of the resulting parts will be passed to the MIDI sequencer. If the cue balls are of different speeds, for example one fast and one slow cue ball, then the cue ball will render the two different parts (i.e. the block's sound fast part + slow part). If, on the other hand the cue balls are the same speed, there will be no additive effect.

### 4.2 Composing for Block Jam

Because we are using MIDI and a sequencer, music can be easily composed for the Block Jam system. During the Block Jam's development, we worked closely with musicians, in an effort to maintain parity between our system and how a modern musician might want to compose their music. It became apparent early in our prototype development that we needed to create a simple authoring structure that the musicians could use. The structure needed to cover two areas, authoring the possible 45 parts (time stamped arrays of MIDI messages), and determining the setup for the MIDI sound module.

After assessing many different sound modules, we chose the "Reason 2" virtual sound studio and software synthesizer because of its high quality output and its modular patching system. The modular patching system allowed the musicians to rapidly plug together a number of virtual instruments and create the rendering context they required. We then created a grid that the musicians could follow in the Reason sequencer, with a space allocated for each of the 45 parts. A part being a time stamped MIDI array, is essentially a small sequence – so to export the parts to the Block Jam system a musician simply exported the entire MIDI song. Because of the grid, we knew where each part was located in the song and could easily parse the data. The same rendering context (Reason file) was used to output the music for Block Jam. This structure allowed the musicians to rapidly compose music for us, which meant that we could try a variety of mappings and types of sound.

Semantically the length of the sounds rendered from a part should have some parity to the speed of the sequence. In the case of our prototype, fast parts tended to last one beat, medium parts two beats long and slow parts one bar (4 beats in 4/4 time). This assists the user in connecting the resulting sound to the activity of the cue ball.

Of course, what is mapped and how long its duration is, is entirely up to the composer who is authoring the music for the system. It is also worth noting that a part does not even have to contain sound triggering (note On) MIDI data, it could just contain sound altering data, such as control change or pitch bend messages.

An obvious limitation to the Block Jam system (in terms of composition only) is that any sound can be played with any other sound at any give moment – which means that all the sounds should work together. The easiest way to achieve this is to keep the instruments distinct (timbre), and play all the notes in the same key.

A way around this limitation would be to create an algorithmic layer (as previously mentioned) that could automatically handle global events such as key changing. Realistically, this would have to be very carefully mapped to Block Jam, giving greater compositional variety without the user losing their sense of control.

## 5. USER RESPONSE

We were hoping to elicit an engaging or thought provoking response from users using Block Jam; therefore, we relied on anecdotal information and our own observations of participatory demonstrations, to assess usability and understanding of the system. Participatory demonstrations were made to members of the Sony community and outside visitors to our laboratory. Block Jam was also demonstrated at SIGGRAPH2002 in the Emerging Technologies Section [[15]]. Response from users was overwhelmingly positive, satisfying our aim to design an engaging system.

Users required a minimum of instruction, and would usually spend 3 or 4 minutes "working it out". Users tended to have the greatest initial difficulty with the most popular feature – dialing. A conflict can occur because the acrylic surface is used for dialing and clicking, so when a user pushes down while dialing (thus clicking) at the same time, they're also inadvertently toggling a blocks function. This could be easily fixed by making the round area in the center of the surface move independently of the surrounding milled groove. Once a user understood the conflict, it no longer interfered with their interaction.

Another observed confusion arose when users tried to select different speeds of cue ball – the play icon (indicating the speed) is displayed after the click interaction has been displayed – so user feedback is after the fact. One user suggested that we should animate the play block icon while it is being clicked – so the change in the speed indicator can be watched as it's being held down.

Users often built similar structures, the abstract complexity of the structure reflecting their understanding of the how the Block Jam system functioned. The first structure built was usually a single row of blocks. Then a user would discover the corner function; this was usually followed by the question "can I build a big loop?" From the "big loop" structure users tended to build a series of 3 rows of different lengths with a play block at each end creating a system for easily observing the timing in relation to the different cue ball speeds. After this point, it was very much up to the individual – some wanting to build visual patterns, massive structures, organized structures, and playful random structures. Once a user started building, it was quite hard to make them stop. We had initially been a little concerned about how quickly a user might get bored of the system – how wrong we were! Users loved to play. Some wanted to use it as a tool for performance, others wanted to collaborate – "It would be great for parties" was a typical response. Everyone wanted to take them home.

Block Jam is not a musical instrument; it is an alternative means of controlling a sequencer. It has no means of continuous control or gesture, and all interactions have a musical latency inherent to quantized sequencers. Conveying expression on such a system might be construed difficult if not impossible. However, users were able to control the musical output expressively through structure and timing, from gentle harmonies to complex beats and rising crescendos.

## 6. WE JAM

In addition to promoting face-to-face collaboration by using a tangible interface, we hoped to promote and explore remote collaboration through a network. We extended the Block Jam prototype by adding a second mother block and PC, to create two nodes and named the new prototype *We Jam*. The nodes were interconnected via MIDI as a means of simulating a real-time network. Though pertinent, we decided to avoid issues of latency normally associated with networks and decided to focus on the issues pertaining to the interaction alone.

A key element in our vision was the notion that different users within a framework interact using different applications – creating a functionally *asymmetric* dynamic. We realized that we had an opportunity to emulate this within We Jam, by allocating each node a different sound set. The first node was allocated a rhythmic sound set running at a speed of 120 beats per minute (bpm). The second was allocated a more harmonic sound set running at a speed of 60 bpm. So when interacted with individually, the nodes provided the users with very contrasting musical experiences, one fast and edgy, the other slow and melodic.

The sound sets were composed so that when the users played together they would produce a pleasing result. We found that inexperienced users, at first tended to play erratically trying to discern which sound belonged to whom, but after a few minutes would start to play cooperatively. Experienced users would play cooperatively, improvisationally reacting to the other user's actions.

## 7. CONCLUSION

In this paper, we have presented the design and implementation Block Jam. We have demonstrated that the Block Jam system succeeds as both a tangible interface and musical application – eliciting a positive experience and provoking a collaborative response in users.

By spatially-multiplexing a sequencer through the use of a tangible interface have created novel interactive possibilities. The notion of collaboration was further extended by the addition of a network to create We Jam – our latest prototype to date.

We hope to continue this work; exploring alternative types of tangible interface, different types of application (looking at gesture in particular), and further explore the use of a network. In our next project we hope to create a collaborative interactive music system for remote users using hand held devices.

We are also interested in applications for the Block Jam tangible interface other than music. After all, nearly all media is sequential, so the system could easily be adapted for visual applications, or creating dynamic narrative structures.

## 8. ACKNOWLEDGEMENTS

We would like to thank the Sony CSL Interaction Lab members, particularly Dr. Jun Rekimoto and Ivan Poupyrev for their help and support. We would also like to thank members of the Sony Design Center, particularly Kei Tostuka and Yutaka Hasegawa. We are also indebted to Ryota Kuwabuko - who designed the blocks' integrated circuit and display mechanisms, and to Kenjiro Mastuo who composed the music used for testing the system and the music for the SIGGRAH demo.

## 9. REFERENCES

[1] Fitzmaurice, GW. (1995) Bricks: Laying the Foundations for Graspable User Interfaces. CHI95 Proceedings ACM Press.

[2] Umberto, E. Translated by Cancogni, A. (1989) The Open Work. Harvard University Press.

[3] Mathews, M.V & Moore, F.R. (1970) Groove – A Program to Compose, Store, and Edit Functions of Time. Communications of the ACM, vol. 13, no.12, December 1970.

[4] Pachet, F. (2002) Playing with Virtual Musicians: the Continuator in Practice. IEEE Multimedia, 2002.

[5] http://www.media.mit.edu/hyperins/

[6] http://www.neurosmith.com/

[7] Resnick, M. (1993). Behavior Construction Kits. Communications of the ACM, vol. 36, no. 7, pp. 64-71, July 1993.

[8] Poupyrev, I. (2000) Augmented Groove: Collaborative Jamming in Augmented Reality. SIGGRAPH 2000 Conference Abstracts and Applications, ACM Press.

[9] Iwai, T. (1999) Composition on the Table. SIGGRAPH'99, Electronic art and animation catalog July 1999.

[10] Ullmer, B et al. (1998) MediaBlocks: Physical Containers, Transports, and Controls for Online Media. SIGGRAPH'98, Conference Proceedings 1998.

[11] Lego Mindstorms, http://www.legomindstorms.com

[12] Rekimoto, J. et al (2001) Datatiles: A Modular Platform for Mixed Physical and Graphical Interactions. CHI2001, Conference Proceedings 2001.

[13] Frazer, J.H. (1982) Three Dimensional Input Devices, Computer/Graphics in the Building Process, March 1982

[14] M.G. Gorbet, M. Orth, and Hiroshii Ishii. Triangles: Tangible Interface for manipulation and exploration of digital information topography. CHI'98 Proceedings, pages 49-56.

[15] Newton-Dunn, Nakano, and Gibson. Block Jam. SIGGRAPH2002, Conference Abstracts and Applications, page 67. ACM Press

[16] http://www.u-ga.com/rez/

[17] http://www.us.playstation.com/games/SCUS-97167/parappa1024_win.html

[18] http://www.hi-res.net/

# Posters

# The Gluiph: a Nucleus for Integrated Instruments

Sukandar Kartadinata
glui
Hagenauerstr. 6
10435 Berlin, Germany
sk@glui.de

## ABSTRACT
In this paper I present the gluiph, a single-board computer that was conceived as a platform for integrated electronic musical instruments. It aims to provide new instruments as well as existing ones with a stronger identity by untethering them from the often lab-like stage setups built around general purpose computers. The key additions to its core are a flexible sensor subsystem and multi-channel audio I/O. In contrast to other stand-alone approaches it retains a higher degree of flexibility by supporting popular music programming languages, with Miller Puckette's pd [1] being the current focus.

## Keywords
Musical instrument, integration, single-board computer (SBC), embedded system, stand-alone system, pd, DSP, sensor, latency, flexibility, coherency.

## 1. HISTORY

### 1.1 Sensor to MIDI - The SensorLab
While work on the gluiph in its current form started in 2001, the original idea goes back to the early 90s at STEIM, Amsterdam. At that time their main line of development were instruments based on the SensorLab [2] which basically is a single-board computer that converts sensor signals to MIDI. The interesting point was that its CPU was not just used to implement a mere data pump but to run an interpreter with an event-driven programming language called SPIDER, which was just powerful enough to render an additional PC on stage unnecessary, instead driving the MIDI gear directly. It was also small enough to be clipped on a belt, so the sensor lines could be kept short for better noise immunity.

### 1.2 Integrating Sound – The SoundLab
The early 90s were also the time when Motorola's 56k DSP chip was at its peak, being used in many commercial synthesizers, samplers, and effect processors as well as academic projects like CNMAT's Reson8 [3], CCRMA's Frankenstein [4], or the extended HMSL [5].

STEIM was introduced to the 56k in '93 by Steven Curtin [6] which lead to the idea to expand the SensorLab with such a DSP, so that one could design sound patches on a development PC and upload them to a stand-alone box in a similar way to the above SensorLab concept, thereby also getting rid of the MIDI rack. This project was dubbed the SoundLab, where, after Curtin left, I took responsibility for adapting the hardware while SPIDER was supposed to evolve to DSPider.

### 1.3 NSP – The PowerMac
However, with '94 came the PowerMac and its promise to do all sound processing on a general purpose computer. While initially not quite up to it one was right to assume that eventually it would, which for STEIM meant to abandon the SoundLab project. Since then native signal processing (NSP) has slowly become the main choice for most people to develop their instruments. Simpler Sensor to MIDI devices were released as now the PC could do the event as well as the sound processing.

### 1.4 Modern DSPs - hardMAX & mtronix
Still, for many years, embedded system held their advantages, and so certain advances in DSP development combined with on-going problems of PC-based systems lead to the hardMAX proposal on the Max/MSP mailing list in '99, with the basic idea of porting this software to an embedded device. However, this was largely rejected as G3 PowerBooks had become the center of stage setups, and hardware been declared as fetishism.

Due to the high development costs work on the SoundLab was stopped at this point, but only until in '01, during some projects for the Berlin company mtronix, a new strategy emerged that seemed feasible. There, the access to development tools and the shared system requirements allowed the design of a new musical instrument platform – the gluiph.

Before covering this in more detail though, I think it is necessary to explain the motivation in pursuing all this.

## 2. MOTIVATION

### 2.1 A DSP-PC Comparison
The justification for building embedded systems seems to have gotten more difficult. Ten years ago one could have come up with the following lists of advantages: latency/timing, processing power, size/weight, reliability. Today, things look different.

Latency and OS-related timing issues have largely been solved, provided one makes careful choices about audio interface and operating system, and maintains a certain amount of "hygiene" on the system. The MIDI bottleneck can still be a problem though, but newer sensor hardware has been released that communicates thru better interfaces (USB, OSC over Ethernet [7][8]), which should at least solve bandwidth problems. Only if zero-latency is required like in digital mixing environments, DSPs can hold their ground.

For processing power things have changed dramatically. At a time when PCs were still confined to non-realtime work, a 56k could easily shoulder reverb algorithms or many voices of sample playback. Now, CPUs with up to 3GHz make it possible to run complex instruments or complete studio setups on a standard PC. DSP-based systems have to use a (massive) parallel processing approach to defend their performance edge (e.g. Kyma [9], Scope [10]).

But then those devices are hardly smaller than a PC or even require an additional one for control purposes. At the same time laptops have replaced the desktops and their big monitors, so touring the clubs got even easier than with the then ubiquitous 19” rack. Only if even smaller systems are required, DSPs have an edge.

The reliability issue is debatable of course. After all we are dealing with complex systems in both cases, so there is always room for unexpected behaviour. And in fact, digital mixers, for one example, do crash at times. Still a point can be made that the additional complexity of a full-fledged operating system, that for the most part was not designed with musicians in mind, increases the likelihood of problems. At least I hear far less about catastrophe scenarios involving hardware equipment than with PCs. Concerning hardware reliability, embedded systems can be designed to provide better stability, e.g. sturdier connectors where you don’t break the motherboard when someone yanks on the cable.

Still, the bottom line seems to be that the advantages of DSP based systems have become somewhat marginal or confined to niche applications. So what is really left to justify the by no means small development effort invested into the gluiph project? The answer lies in the possibility of integration.

## 2.2 What defines an instrument?

I would like to start this discussion with a picture of a typical stage setup.

**Figure 1. Two laptops, two audio interfaces, a master keyboard, two fader boxes, two MIDI interfaces, a mixer, 8 power supplies, 20+ cables**

So is this what I want to call “my instrument”? Some might ask why not, but to me it looks more like an experiment from a laboratory. While there is sure nothing to say against being experimental in creating and performing one’s music, it is the goal of the gluiph project to derive at more mature instruments with a stronger identity. Integrating the defining components into a single physical entity is an important step in this direction, and not just a matter of aesthetics and easier installation (although those are by no means to be underestimated).

Flexibility, the key advantage of PCs, is a powerful concept for exploring the vast possibilities of today’s sound offerings. However, if a musician eventually wants to master his instrument, there comes a point when a decision has to be made about consolidating the existing setup, rather than getting lost in an ever-shifting environment. This reasoning follows the notion that reduction is an important phase in defining one’s means of expression. And that power has to be traded with nuance.

Of course, even a violin needs a bow, and an electric guitar its amplifier and loudspeaker. So not everything can and should be integrated. However, there are designs where the further integration of speakers can be advantageous (e.g. Nic Collins’ trombone [11]). In any case this exemplifies how the loudspeaker marks one of the key differences between electric and acoustic instruments, for which the “interface”, the “processing”, and the “speaker” are all intrinsicly embedded within their physical structures. To achieve such a degree of coherency for electronic instruments is one of the main goals of the gluiph project.

## 2.3 Related developments

Before describing the gluiph’s technology a few parallel development efforts are discussed that in one way or the other are related to its approach. Concerning the main concept of having a stand-alone music computer that receives its programming from a host PC, there are quite a number of commercial devices available:

Clavia’s Nord Micro Modular [12] also shares the gluiph’s size, however its sound generation is limited to virtual analog, i.e. mostly subtractive synthesis. More flexible are Creamware’s Noah [13] and Manifold’s Plugzilla [14] with their broader range of available DSP modules, where the latter relies on VST plugins. Both are 19” rack modules though, so they are hardly smaller than a laptop. Another device of that format is Sound-Art’s Chameleon [15], however it requires 3rd-party developers to write modules in DSP assembler. The same applies to the Death Synth by Noah T. Vawter [16] which again is smaller and was designed with PDA control in mind.

However, none of the above aims at integration but expects outside control thru MIDI, with all its limitations.

Others have suggested that, after the laptop, the PDA itself will be the next device to reach performance levels that will allow sound processing. However, currently the key problem is their missing floating point support which makes it difficult to run existing music applications. Wether this will change in the near future is rather unclear. The same holds true for the majority of commercially available Linux-SBCs.

# 3. TECHNOLOGY

## 3.1 Hardware

As mentioned in the introduction the idea to pursue the gluiph project was renewed while working for mtronix, a small Berlin company which mainly develops precision measurement instruments. They serve an industry where PC-based solutions are out of the question mainly because of reliability issues, operability requirements, and size constraints. Add to this the need for a powerful processor and realtime capability and one gets pretty close to the requirements of a musical instrument.

The key components of their boards were a Philips TriMedia CPU, SDRAM memory, Flash memory, programmable logic, PCI interface, and various sensor sections depending on the project. One interesting thing about the TriMedia is that it was designed as a multi-media processor with set-top boxes in mind, so this seems like a strange choice for measurement devices. For mtronix though it made sense as image analysis is one of their main applications, so they could make good use of the integrated video co-processing units. Video analysis, of course, is also of interest for controlling music software.

Another important reason for choosing the TriMedia was the fact that its core employs a very modern VLIW implementation with multiple execution units that are scheduled at compile time to keep the design light-weight yet powerful. As a consequence assembler programming is no longer feasible due to all the complex scheduling decisions to be made. Of course this approach requires a capable compiler, and in fact the one provided leaves nothing to ask for, while also complying to the full C/C++ ANSI standard and packing a well-tuned set of libraries.

Finally, and this with regards to the gluiph, the TriMedia features a 2in/10out-channel audio interface, so not much was missing to attempt a modification of the mtronix design towards a musical instrument platform. This was basically implemented by adding converters for the audio section and choosing a sensor subsystem. In addition the PCI connector was replaced with a USB port for a more appropriate host connection.

Also worth noting is the role of the programmable logic chip (CPLD). While many sensors provide an analog signal that can simply be connected to the sensor ADCs, others transmit digital data thru serial interfaces using a variety of protocols and baud rates. The CPLD can easily be adapted to those so no extra micro controllers are required. It can also be told to drive displays and relays or include precise timers for measuring ultrasound time lags. Towards the TriMedia it connects thru a high-speed interface that can be operated synchronously to the audio frame rate. MIDI, of course, is totally out of the game.

## 3.2 Software

The main premise in designing the software concept for an embedded music system was to do away with assembler programming. While in the past I had enjoyed tweaking DSPs for maximum performance, this was clearly no feasible approach any longer, mainly because of the lack of flexibility and the maintenance effort required.

The second decision was to avoid writing yet another patcher or even coming up with a whole new music programming language. Instead the plan was to port existing software, not only to reduce the development effort but also to benefit from the associated user bases. This, apart from the integrative concept, is the main difference between the gluiph and the other DSP based solutions mentioned above.

The third aspect emerged after the hardMAX proposal and its rejection, which was to rely on open-source software, thereby also following the general movement that was starting to grow at the time. The first candidate here was CSound as it could be operated from a command line in a non-realtime mode, which facilitated first tests for general operability. In fact, the port was completed literally in less than a day, and displayed performance levels comparable to a G3 PowerMac which ran at twice the CPU clock.

Rather than adapting CSound for realtime operation, it was then decided to port pd, partially because of CSound's more restrictive license, but mainly for personal taste and because Miller Puckette was encouraging the project. The main modifications to the source dealt with the MIDI implementation which was replaced with a direct link to the sensor subsystem. Other (minor) changes included glue code to the audio drivers as well as to the file system. Finally some performance optimizations were necessary, e.g. in handling the double precision arithmetic required by pd.

However, no special sensor-related external objects were written, rather the sensor signals are relayed to pd thru its standard send/receive objects.

## 3.3 IMPLEMENTATION EXAMPLES

### 3.3.1 *The SKRUB (for Richard Barrett)*

The SKRUB at first glance might not be the most spectacular sensor instrument, as it is "just a keyboard". However, it illustrates the effect of integration that can be achieved with the gluiph as it basically boasts the same functionality as the setup shown in Figure 1. There the PowerBooks run Max/MSP resp. LiSa patches that are triggered from a MIDI master keyboard and further controlled by two fader boxes as well as some foot pedals.

For the SKRUB the fader boxes were replaced with a row of trackpads, that have the advantage to be within easy reach from any position on the keyboard. They also add two dimensions of control – sideway movement and pressure (actually coverage).

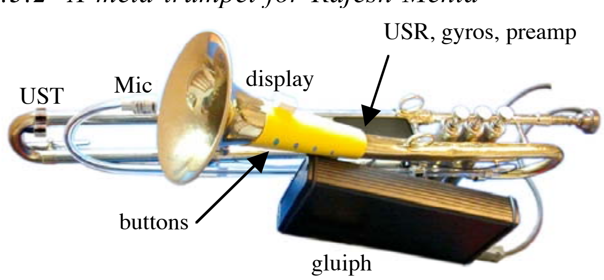
The keys themselves differ from ordinary ones by supplying continuous vertical position information for every key, extending the "trigger with velocity and aftertouch" scheme of standard keyboards. Thus a key can be used to e.g. scrub along a sample. Or a set of keys to knead thru a graphic equalizer.

For both sensor systems the CPLD proved very helpful. For the trackpads a 9-channel PS/2 interface was implemented, while the linear position sensing for the keyboard required a 61-channel frequency counter to measure the frequency deviation of the inductive sensor circuits. All key positions are sampled at 345Hz (fs/128) while trackpad coordinates are measured at 115Hz (fs/384, limited by PS/2 speeds).

The SKRUB also includes a 4-channel audio output for which the PowerBook setup requires external interfaces. Finally reverb was integrated, although not as software within the gluiph but thru an additional chip.

The control interface was kept at a minimum, providing only one LED per trackpad, its brightness resp. color reflecting the current levels of the parameters it controls. An additional button is used to switch to patch/bank select mode, however no extra number keys are used for selection, rather the keyboard keys take over this function.

### 3.3.2 *A meta trumpet for Rajesh Mehta*




This trumpet is an acoustic/electronic hybrid instrument. It is the first gluiph-based design and only employs a small number of sensors. The center box contains a 2-D gyro sensor that is used to measure yaw and pitch. The yellow leather sleeve around the horn houses a small display and a set of simple touch buttons that select different operating modes. The trumpet also stands out thru its trombone-like slide that allows a pitch change of a minor third. This extension is measured with an ultra-sound system.

An important aspect for Mehta was wireless operation so he would be free to move around on the stage. With the gluiph and an integrated RF unit, he now doesn't even need a "tech table" but can transmit directly to the house system. However, the need for a battery with its considerable weight didn't allow the gluiph to be mounted right next to the trumpet, so an extra belt-mounted box became necessary. On the other hand this will make it easier to handle his other trumpets that we plan to equip with sensors.

### *3.3.3 Other developments*

Currently under construction is a virtual drum that will highlight the possibility for further integration by incorporating the loudspeaker. Its design is based on a subwoofer that has been covered with different materials (leather, velvet, foam pads) to achieve varying deceleration behaviour when hitting it with the accelerometer equipped gloves. The latter also carry small speakers for the higher frequency ranges. This also adds an additional spatial aspect to the instrument apart from the position sensors.

Finally, a DJ mixer is currently being modified to allow for spatial crossfades across a multi-channel output. Other instruments are in the planning phase.

## 4. THE FUTURE

So where will the gluiph project go from here? While the production of the mainboard seems to be in safe hands with mtronix, the first two projects showed that the amount of customization required is the main factor in determining the amount of time and money involved in designing a gluiph based instrument. With the current one-man workshop situation it is getting increasingly difficult to pursue this.

Therefore it would be desirable to establish an environment where an institution could cover the custom development as well as provide logistic support. Efforts to create such a place in Berlin have proven difficult in the past but carry on.

As far as technology is concerned, the gluiph motherboard will be updated on a regular basis following mtronix' product cycle. A new board with a faster CPU and expanded I/O possibilities is due out for the end of the year. On the software side a port of the recently open-sourced SuperCollider is under consideration.

## 5. ACKNOWLEDGMENTS

My thanks to Miller Puckette for his initial encouragement and developing pd in general. I'd also like to acknowledge the commitment of mtronix as well as the patience of my first users.

## 6. REFERENCES

[1] Puckette, M. "Pure Data." Proceedings of the ICMC, 1996.

[2] http://www.steim.org/steim/sensor.html

[3] J.-B. Barriere, P.-F. Baisnee, A. Freed, and M.-D. Baudot "A Digital Signal Multiprocessor and its Musical Application," Proceedings of the ICMC, 1989.

[4] Putnam, W. & Stilson, T. "Frankenstein: A low cost multi-DSP compute engine for the music kit". Proceedings of the ICMC, 1996

[5] Burk, P. "The Integration of Real Time Synthesis into HMSL", Proceedings of the ICMC, 1991.

[6] Curtin, S. "The SoundLab: a wearable computer music instrument", Proceedings of the ICMC, 1994

[7] Avizienis, R., A. Freed, T. Suzuki, and D. Wessel "Scalable Connectivity Processor for Computer Music Performance Systems". Proceedings of the ICMC, 2000.

[8] http://www.la-kitchen.fr/hardw/kroonde.html

[9] Hebel, K. and C. Scaletti. "The Software Architecture of the Kyma System". Proceedings of the ICMC, 1993.

[10] http://www.creamware.de/en/Products/SFP/default.asp

[11] Nicolas Collins, "The Evolution of Trombone-Propelled Electronics," Leonardo Music Journal, Vol. 1, 1991.

[12] http://www.clavia.se/nordmodular/index.htm

[13] http://www.creamware.de/de/products/Noah/default.asp

[14] http://www.plugzilla.com/

[15] http://www.soundart-hot.com/english/index.htm

[16] http://www.gweep.net/~shifty/death/

# Pointing Fingers: Using Multiple Direct Interactions with Visual Objects to Perform Music

Jean-Michel Couturier
LMA-CNRS
31, chemin Joseph Aiguier
13402 Marseille Cedex 20, France
0033 491 16 42 10
couturier@lma.cnrs-mrs.fr

Daniel Arfib
LMA-CNRS
31, chemin Joseph Aiguier
13402 Marseille Cedex 20, France
0033 491 16 42 10
arfib@lma.cnrs-mrs.fr

## ABSTRACT

In this paper, we describe a new interface for musical performance, using the interaction with a graphical user interface in a powerful manner: the user directly touches a screen where graphical objects are displayed and can use several fingers simultaneously to interact with the objects. The concept of this interface is based on the superposition of the gesture spatial place and the visual feedback spatial place; it gives the impression that the graphical objects are real. This concept enables a huge freedom in designing interfaces. The gesture device we have created gives the position of four fingertips using 3D sensors and the data is performed in the Max/MSP environment. We have realized two practical examples of musical use of such a device, using Photosonic Synthesis and Scanned Synthesis.

## Keywords

HCI, touch screen, multimodality, mapping, direct interaction, gesture devices, bimanual interaction, two-handed, Max/MSP.

## 1. INTRODUCTION

In computer music, different strategies are possible to control sound processes. A first one consists in using the computer properties of calculation power and flexibility in the design phase of an instrument. Today, many researches are conducted to create powerful digital musical instruments. To design them, a critical part of the work consists in the mapping between the gestural devices and the sound processes to control [1]. Those instruments tend to reproduce the "instrumental link" [13] that is intrinsic to the acoustic instruments and that has often disappeared in the electronic and numerical systems.

A second strategy consists in using the computer for its powerful interaction trough a graphical user interface (GUI). Regarding today musical softwares, they essentially use a mouse and a keyboard with a current GUI: all sound parameters are controllable via graphical objects that generally represent real objects like piano keyboards, faders, etc. Complete studios equipments and electronic instruments emulators are now integrated in the computer. The GUIs tend to reproduce on the screen an interaction area close to the real one, like front panels of electronic instruments. The aim of such interfaces is to give the user the impression of real objects in front of him. Nevertheless, with a single mouse, the interaction process is poor: the gesture space (the place where is the mouse) is separated from the interaction space (the screen) and only one object can be manipulated at one time. This explains why many software programs are configured to use "external" devices like MIDI controllers, software-specific control surfaces or alternative controllers. In this case, the full system is similar to those of the first strategy; the graphical objects, which are designed for interaction, are only used for visual feedback or not used at all.

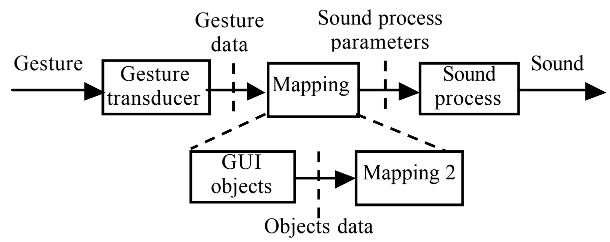



**Figure 1. Mapping chain in a digital musical instrument. The first strategy links gesture data to sound parameters; the second strategy adds an additional step in the mapping: the graphical objects are linked to sound parameters and the gesture device can control any graphical object.**

The system we introduce in this article enables the control of graphical objects in GUI's like real objects and rather follows the second strategy. This new powerful multimodal system, the *Pointing Fingers*, performs a direct control on GUIs with a multi-touch touchscreen-like device, designed for musical control. Section 2 introduces the interaction principle; section 3 describes the gesture device and section 4 the software implementation. Finally, in section 5, some musical examples of what is possible with such a system are exposed.

## 2. A NEW APPROACH IN INTERACTIVE SYSTEMS

The system is based on the combination of two crucial features: the superposition of both gesture spatial place and visual feedback spatial place and the ability to have multiple simultaneous controls when using a GUI. Some systems that have these two features already exist; one of them was developed to control musical processes: the Audio Pad [10], based on tangible interfaces [7] in which the objects to manipulate are real and interact with graphics. Our system is closer from current GUIs because the objects to manipulate are the virtual graphical objects displayed on screen.

This type of system provides the most direct and intuitive interaction possible: our fingers are manipulating graphical objects as if they were real objects. There are no material constraints on the objects: they can change in position, size, shape and function. It is possible to display some information beside the objects to help the user. It is a very efficient system to control virtual copies of real objects. Finally, interaction situations that are impossible in the real world can be implemented here, like manipulating moving objects, as it will be demonstrated in the section 5.

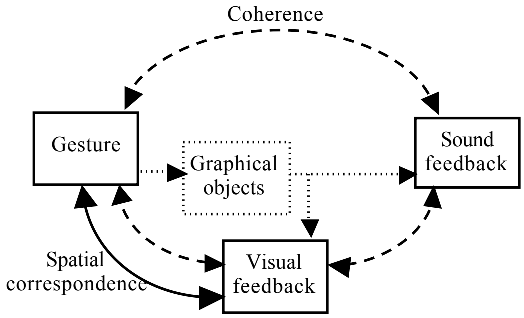



**Figure 2. In this system with multimodal feedbacks, the relationship between action and perception is coherent; this coherence is reinforced by a spatial superposition of the gesture area and the visual feedback area. Those features add strong presence to the system.**

In interaction with a real object, this object provides some haptic feedback: the contact with the object shape, the force it needs to be manipulated, the degrees of freedom it offers and the spatial limits of its displacement. This feedback is so important that the user could manipulate an object with the eyes closed. With our system, the haptic feedback is reduced to the contact between fingers and screen. Sight and hearing are fully used; sight permits to locate the position of the objects in the screen and hearing can reinforce sight when an object is manipulated, through the effect of manipulation on the sound.

The GUI of our system is close to those using a mouse to control graphical object; the differences are that the object needs a bigger size, because a fingertip is bigger than a mouse pointer. The screen area contains different interaction zones; each zone will have its own interaction mode and connection to the sound process parameters.

Different types of gestures are necessary to act in a zone: selection gesture to select the chosen zone among several zones, modulation or continuous gesture to modify the parameters that are associated with the zone, and decision gesture to stop the interaction. For example, if the user wants to manipulate the graphical object " fader", he selects this fader with one of his fingers, manipulates it, and then he lifts his finger off the screen area.

## 3. THE POINTING FINGERS SYSTEM

We want a device that follows our requirements: having multi-touches and interacting directly with the interface. Commercial touchscreens fulfill the second point, but unfortunately, they does not allow multi-touches. Many other solutions have been developed in different labs, as the following examples: the SmartSkin [11] system combines a prototype of multi-touch surface with a video projection; the vision-based finger tracking [6] determinates the fingers' positions through a video analysis; Mulder's system combines two CyberGloves and two Polhemus position/orientation sensors enables to find the position of the fingertips [9]. The device we have developed represents a simple alternative to all that exists.

### 3.1 The Gesture Device

The device we introduce now is a first prototype we have made to perform multi-touches on a screen. It consists of 2 semi-gloves (recovering the thumb and the index) with two 3D position/orientation sensors and two switches per hand (see Figure 3). This device is close to Mulder's CyberGloves and Polhemus system [9], but is less expensive in hardware and simpler to implement.

**Figure 3. The gesture device uses *flock of birds* sensors with 4 *birds* (receivers). The *birds* are fixed on the thumb and the index of each hand so that no motion is possible between the *birds* and the fingertips. Switches are fixed on each fingertip and can be used like a mouse click button.**

This device can give the position of 4 digits (the thumb and the index of each hand) with approximately 1 mm accuracy and the on/off values of the switches (an equivalent of the mouse click button) localized at the extremity of the fingers; those switch buttons indicate if the fingertips are physically touching the screen or not. All the data of the sensors are processed in the Max/MSP environment. The *flock of birds* [3] is a commercial device composed of a transmitter and several receivers, called *birds*; the device communicates with the computer trough a serial interface and a serial/USB converter. We use the *serial* object of Max to receive the data. The switches are connected to the electronic of an USB joystick and we receive its data, using the *insprock* object.

However, this device has some limitations. The *flock of birds* device introduces some latency: we have not measured it but we estimate it to be approximately 30 ms with four sensors; this lag is too important to create really reactive instruments, but is acceptable for our experiments and applications with modulation-like instruments. Another problem is the choice of the screen: CRT screens are disturbed by magnetic fields, and some LCD screens disturb the magnetic field of the sensor.

### 3.2 Converting the Data of the 3D Sensors

We have developed a specific C object for Max to transform the data of the *birds* and find the fingertips coordinates in the screen base. The sensor gives the absolute position and orientation of the 4 *birds* in space, relatively to the transmitter; with this data, the object calculates the position of the tips using the rotation matrix between each *bird* base and the transmitter base. This coordinates are then rotated and translated to the screen base and rescaled in order to obtain the position of the tips in pixel, which is the mouse coordinates unit (Figure 4). A calibration procedure calculates the screen position and size in the transmitter base and then determines the screen base.

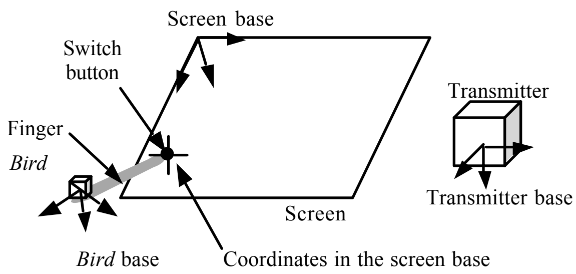



**Figure 4. For one finger: the fingertip position is known in the *bird* base, and the *flock of birds* gives the *bird* relative-to-transmitter position/orientation. With the screen position, the program calculates the coordinates of the fingertip in the screen basis.**

The object returns the (X,Y) coordinates of the four fingertips in the screen base. The switch button state, given by the *insprock* object, is added to the corresponding list.

## 4. CONTROLLING GRAPHICAL OBJECTS

In this section, we develop how the data of our gesture device or any equivalent device will be processed. Indeed, in our approach, we try to build modular systems. So the control of the graphical object is completely independent from the gesture device: we consider that any gesture devices that can give us lists with the point number, (X,Y) coordinates in the screen basis and the value of a on/off button can be used instead of the Pointing Fingers. For this reason, we will call *pointer* a point on the screen that is given by the gesture device. Our gesture device gives simultaneously 4 pointers.

We used the Max/MSP environment and we created a specific Max object to manage the data for a given zone of the screen: multipoint provides some confusion problems that did not exist with the only mouse. The object receives all data lists from all pointers. The delimitations of the object action zone are given by sending specific instructions to it. Figure 5 describes how this Max object manages multiple points for a given zone.

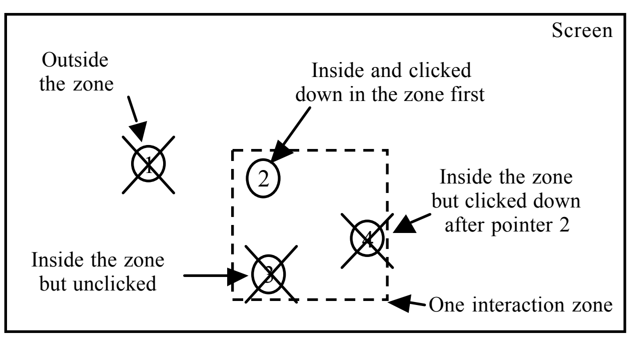



**Figure 5. The Max object returns the coordinates of a pointer only if this pointer is clicked down in the zone of the object and if the zone does not contain another active pointer.**

The outputs of this max object can be connected with many graphical objects, taking care of the coherence between the visual effects of the interaction on the graphical object and the position of the pointer on the screen.

This implementation is simple but sufficient to perform numerous things. Firstly, lots of Max graphical objects can be used with our Max object and can be manipulated simultaneously. Secondly, many original graphical objects or interaction zones can be created and used with our system, as the section 5 will show, and we can imagine multipoint interaction zones using several units of our Max object.

## 5. EXAMPLES

### 5.1 Control of the Photosonic Synthesis with the Pointing Fingers Device

Created by Jacques Dudon, the Photosonic instrument is an optical musical instrument based on the following principle: a solar photocell receives the rays of a light that are intercepted by a rotating disk and an optical filter (see figure 6). The electric current of the photocell is the audio signal that the instrument produces. An optical comb-filter can be placed on the trajectory of the light to modify the sound of the disk. The sound of the instrument depends on the position of the light, the waves inscribed on the photosonic disk, and the position of the filter.

We have made an emulation of the instrument in the Max/MSP environment, which was presented in NIME 2002 [2]. This digital version of the instrument uses a graphical tablet with a mouse and a pencil to control the position of the light and the filter. Now we will introduce an implementation in which the Photosonic synthesis is controlled by the *Pointing Fingers*.

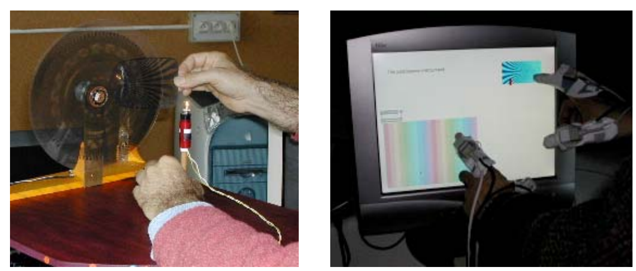


**Figure 6. The optical instrument and its digital emulator's interface. The emulator proposes the same control than the real instrument, where the user moves a light in front of the disk and a filter between the disk and the photocell.**

We have created a graphical interface close to the optical version of the instrument. This interface contains two interaction objects, like the real instrument: the light and the filter. The filter is the same than the optical one, with a rectangular shape. A circle represents the light and can be displaced below the disk, which is represented in a rectangle divided in several parts corresponding to the rings of the Photosonic disk. The interaction principle is here similar to the interaction with real objects: when one object (the light or the filter) is *clickdown* or selected on the screen, it follows the displacements of the fingertip. When the finger switch is *clickup*, the activation zone is the new position of the object. This interaction mode provides a digital instrument that is really close to the real optical instrument in terms of handiness.

### 5.2 String Control in Scanned Synthesis

Scanned Synthesis was developed by Verplank, Shaw and Mathews [12] and enables the generation of sound thanks to the slow movements of mechanical systems, which shape is used to create dynamical wavetables. We have implemented this technique with a circular string model in finite differences for the mechanical system. A C object that provides a high level control of the Scanned Synthesis was created [5] for the Max/MSP software and was used for the realization of a complete musical instrument demonstrated at NIME 2002 [4]. In these previous works, the string was put in motion by forces or by throwing it from a pre-definite shape. The string shape was only displayed as graphical feedback. However, The link between sound and the string visual representation is direct: the motion of the string is perceptible with the eyes and its speed corresponds to the speed of the human body gestures. Because of these features, we felt like interacting with the string directly using our fingers, in real time. Scanned Synthesis is the perfect synthesis technique to use with our new gesture device.

We then modified our C object in order to enable direct interaction with the string and we create a GUI that displays the string shape, an area to control the pitch (that was implemented in our instrument and controlled by a graphical tablet) and other controls.

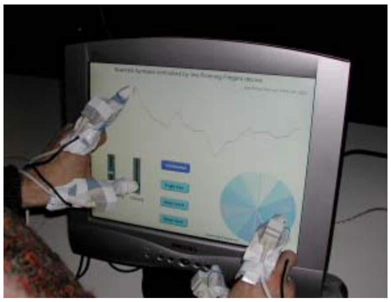


**Figure 8. Direct manipulation of the string of the Scanned Synthesis algorithm. One can simultaneously interact with the string, control the pitch and change string parameters.**

On the string, the interaction principle is the following: a finger makes a selection gesture up or down the string; according to its initial position, the finger pushes the string up or down. The pitch control is localized in another area and uses the angular frequency control developed by Kessous [8]. Two sliders are modifying the string damping and stiffness; 4 buttons enables to stop the sound and to choose to play a single note or chords. The sliders and the buttons are standard Max graphical objects that receive data of our system.

Scanned Synthesis is a complex method that disposes of a high number of parameters. Nevertheless, controlled by our system, this synthesis technique becomes easy to use and gives remarkable presence to the interaction.

## 6. CONCLUSION - PERSPECTIVES

The computer often seems to be a powerful creature inside a closed box. Its screen shows us marvelous worlds, but interacting with a mouse is frustrating, especially when we want to perform music. As the two examples shows, our system will help to design musical instruments that benefits of the advantages of the computers' universality and flexibility, through a powerful control of Graphical User Interfaces.

Our works on this system have just established its basis; in the future, we will develop new objects, implement other synthesis techniques and improve the system to provide a complete environment to create new digital musical instruments.

## 7. REFERENCES


[1] D. Arfib, J.M. Couturier, L. Kessous, V. Verfaille, “Mapping strategies between gesture control parameters and synthesis models parameters using perceptual spaces”, Organised Sound 7(2), Cambridge University Press, pp. 135-152, 2002.

[2] D.Arfib, J. Dudon: “A digital emulator of the photosonic instrument”, Proceedings of the 2002 Conference on New Instruments for Musical Expression (NIME-02), Dublin, Ireland, May 24-26, 2002.

[3] Ascension Technology Corporation, http://www.ascension-tech.com/

[4] J.M. Couturier, “A scanned synthesis virtual instrument”, Proceedings of the 2002 Conference on New Instruments for Musical Expression (NIME-02), Dublin, Ireland, May 24-26, 2002.

[5] JM. Couturier, Scansynth~: Scanned Synthesis object for Max/MSP, 2002. http://www.lma.cnrs-mrs.fr/~IM/en_telecharger.htm

[6] C. von Hardenberg, F. Bérard, “Bare-Hand Human-Computer Interaction” Proceedings of the ACM Workshop on Perceptive User Interfaces, Orlando, Florida, USA, Nov. 15-16, 2001.

[7] H. Ishii and B. Ullmer, “Tangible Bits: Towards Seamless Interfaces between People, Bits, and Atoms”, Proceedings of Conference on Human Factors in Computing Systems (CHI ‘97), ACM Press, pp.234-241, 1997.

[8] L.Kessous, “A two-handed controller with angular fundamental frequency control and sound color navigation”, Proceedings of the 2002 Conference on New Instruments for Musical Expression (NIME-02), Dublin, Ireland, May 24-26, 2002.

[9] A. Mulder, S. Fels and K. Mase, “Design of Virtual 3D Instruments for Musical Interaction”, Proceedings of Graphics Interface '99, pp. 76-83, Toronto, Canada, 1999.

[10] J. Patten, B. Retch, H. Ishii, "Audiopad: A Tag-based Interface for Musical Performance", Proceedings of the 2002 Conference on New Instruments for Musical Expression (NIME-02), Dublin, Ireland, May 24-26, 2002.

[11] J. Rekimoto, “SmartSkin: An Infrastructure for Freehand Manipulation on Interactive Surfaces”, Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 113-120, Minneapolis, USA, 2002.

[12] B.Verplank, M. Mathews, R. Shaw, "Scanned Synthesis", "Proceedings of the 2000 International Computer Music Conference", p: 368-371, Berlin, Zannos editor, ICMA, 2000.

[13] M. Wanderley, “Performer-Instrument Interaction: Applications to Gestural Control of Music”, PhD Thesis, Paris, University Pierre et Marie Curie - Paris VI, 2001. http://www.ircam.fr/wanderle/Thesis/Thesis_comp.pdf

# LEMUR GuitarBot: MIDI Robotic String Instrument

Eric Singer
LEMUR
New York, NY
eric@ericsinger.com

Kevin Larke
LEMUR
New York, NY
kevin@soundball.com

David Bianciardi
LEMUR
New York, NY
david@av-controls.com

## ABSTRACT
This paper describes the LEMUR GuitarBot, a robotic musical instrument composed of four independent MIDI controllable single-stringed movable bridge units. Design methodology, development and fabrication process, control specification and results are discussed.

## Keywords
Robotics, interactive, performance, MIDI, string instrument.

## 1. DESIGN GOALS
In early 2001, LEMUR set out to create an ensemble of robotic musical instruments controllable by MIDI and playable by human musicians or from a computer. GuitarBot, one of several instruments created by LEMUR, was designed to be a responsive robotic stringed instrument controllable via MIDI for the performance of live, generated or sequenced musical works in concert or installed settings.
To help satisfy our goal that it could adopt any number of configurations and aesthetic treatments, the instrument is comprised of 4 identical modules which can be mounted and arranged to suit the needs of a given work. Each module is a monochord under tension suspended between 2 fixed bridges. Pitch variation is achieved by a motorized servo-positioned movable bridge which travels along the length of the string, in a manner similar to a slide guitar. The string is plucked, struck, bowed and otherwise excited by a system of plectra and other mechanical and electromagnetic actuators. A solenoid damper is employed to inhibit string vibration under control.
The sound of the string vibrating is electrified by an electromagnetic pickup positioned above and adjacent to the string, and processing and amplification are used to deliver the instrument's performance to the listener.

In order to maximize GuitarBot's usefulness to the greatest number of users, we adopted the MIDI standard for communication between the modules' onboard microcontroller and the outside world. The instrument module can be set to several MIDI modes which range from offering an external sequencer complete access over all parameters and functions of the robot, to condensed command sets suitable for control of the instrument by a live performer using a keyboard or other, more exotic controller.

General design principles also dictated that the instruments be fabricated in a manner as precise and repeatable as possible, with the goal of producing a series of durable, serviceable, and upgradeable devices. To this end, the iterative design process made heavy use of CAD to model and revise the instruments prior to milling and assembly.

## 2. DEVELOPMENT
The current version of the GuitarBot is the result of three design iterations. The first prototype was created experimentally, crafting the basic design from aluminum and investigating various mechanisms for the slide and picking systems, modifying and refining parts until we had a working proof of concept.

This unit was then modeled to scale in 3D using the Vectorworks CAD package which allowed us to explore design scenarios before committing to and machining the next version. This second version was built to spec from the CAD model and tested. Additional changes were made to the CAD model, further refining the accuracy and playability of the instrument. As a reflection of our growing understanding of the milling, machining and assembly processes, design enhancements at this stage also served to increase the ease of manufacturing and servicing the unit.

Following this round of changes, the third and current version of GuitarBot was constructed and is described below.

## 2.1 Mechanical
Each Guitarbot module is assembled on a 36" x 4" aluminum base. A steel electric guitar or bass string, with a diameter range of .01" to .02" for plain strings, and .02" to .1" for flatwound strings, is stretched between 2 fixed bridges and tensioned with a worm drive guitar tuning peg. Open string pitches from E at 41.2Hz through E at 392.6Hz are within the nominal range of the instrument. The effective maximum length of the string is 27.03", occuring when the moving bridge is furthest from the base fixed bridge. The moving bridge has a travel length of 21.44" inches, or 79.3% of the effective length of the string, thus ensuring a pitch range of over 2 octaves.

The moving bridge assembly travels in a ball bearing slide track and is positioned by means of a drive belt affixed to the

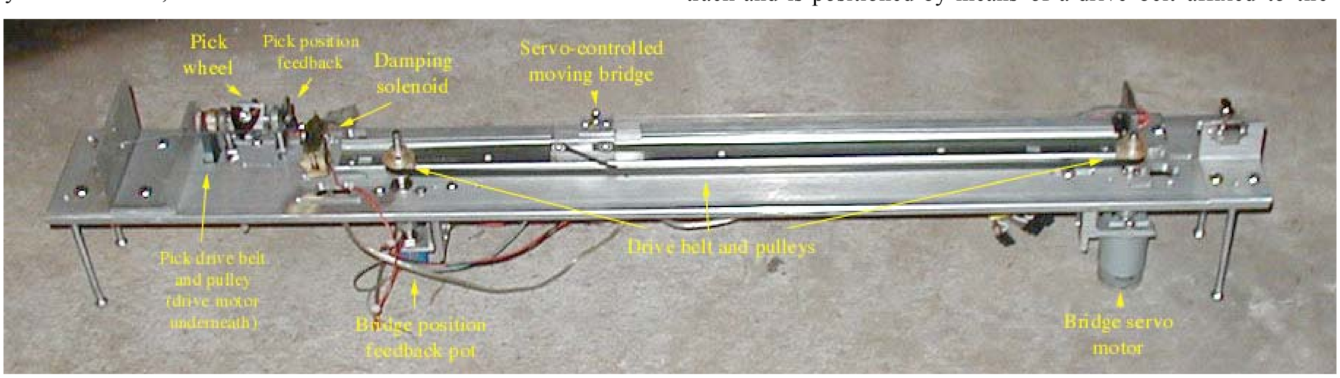



**Figure 1. GuitarBot prototype**

bridge and riding on pulleys. The drive pulley is driven by a DC servo motor, and the idler pulley is coupled to a 10-turn rotary potentiometer. End-to-end travel time of the slide assembly is under 250ms.

In contrast to previous mechanical guitar-like instruments like Sergi Jorda's 6 string hammer-on fretted robot[1], the string is excited by means of a plectrum mechanism consisting of 4 nylon guitar picks mounted on a block that rotates on a shaft. The shaft is belt and pulley driven by a DC servo motor on the underside of the assembly. Position feedback is derived from an encoder wheel mounted on the end of the shaft, which passes between the emitter and detector of a photosensor module. The closed loop servo motor system guarantees that the pickwheel assembly can be quickly and accurately rotated so that a note can be struck and the next pick brought into position to await the next trigger. Additionally, the pickwheel can be rotated at various speeds to produce humanly impossible feats of tremolo.

A "clapper" solenoid is used as a damper which, when activated, closes on the string and stops vibration. This allows the instrument to respond to note-off requests and also permits a playing technique in which the string is picked while dampened, producing a muted "thumping" and more percussive sound without sustain.

An electromagnetic pickup of our own design is affixed to a flexible metal arm which clamps to the base. The design of the arm permits the pickup to be positioned at various places along the length of the string to take advantage of variations in tone. We used a 1/4" diameter rare earth magnet as the base of our hand wound single coil pickup which, when complete, proved to have excellent gain and noise rejection characteristics when compared to off the shelf systems with which we initially experimented.

## 2.2 Electronics

Each GuitarBot module has onboard circuitry to handle all actuators, motor control and feedback locally. Custom control boards were designed around a Microchip PIC16F87x-series microcontroller and contain two DC servo amps, MIDI I/O circuitry and power supply and microprocessor support components. The board additionally features assorted diagnostic and status LEDs and in-circuit serial programming connections.

Inputs to the microcontroller include opto-isolated MIDI-format serial, analog values from the slide position potentiometer and digital input from the pickwheel encoder. The slide position pot values are sampled at 10-bit resolution, which, as we are using approximately 95% of the pot range, results in an effective resolution of almost 1000 linear positioning steps over the 2 octave range of the instrument.

Outputs from the microcontroller include MIDI-format serial, PWM motor control, and digital control of the damper solenoid.

## 2.3 Software and Control

The GuitarBot embedded software system can be broken into three components: Input, Configuration and Control. The Input component is responsible for receiving, filtering and dispatching MIDI messages to the other components. The Configuration component is used to save, restore and implement control routing presets and the system tuning table. The Control component is responsible for driving the bridge and pickwheel motor and managing the damper state.

The bridge positioning subsystem is the most complex piece of the Control component. The primary objective of this subsystem is to position the bridge smoothly and accurately at a user definable speed. The following is a summary of the structure and operation of the bridge subsystem.

The inputs to the bridge positioning subsystem:

1. The current position of the bridge as measured from the idler pulley rotary potentiometer and converted to a digital value via the microcontroller's on-board analog-to-digital converter (ADC).
2. A user defined target position and travel velocity. The travel velocity determines how fast the bridge should travel to the target position.

The outputs calculated by the bridge positioning subsystem:

1. The bridge motor voltage. This output directly determines the speed of rotation of the motor and therefore the speed of motion of the bridge.
2. The bridge positioning motor direction. This output determines whether the bridge moves toward higher or lower pitched notes.

The bridge positioning subsystem periodically generates a new set of output signals by running the following algorithm:

1. Determine the current bridge position by reading the rotary pot ADC.
2. If the bridge target position is not the same as its current position generate the best motor voltage and direction for moving the bridge towards the target position at the user defined travel velocity.
3. Wait until the next read time occurs then go to 1.

The obstacles to achieving smooth and accurate control of the bridge can be demonstrated by considering how the physical bridge system responds to repositioning requests at very high and very low speeds. For example at high speeds the bridge will tend to overshoot the target position. The overshoot is a result of the momentum the bridge may have gained during travel and the inevitable delay between when the rotary pot ADC is read, when the next motor voltage is computed, and how quickly the motor is able to respond to that new voltage. At low speeds the physical system has a different set of characteristics. For example it will tend to move sporadically as it overcomes static friction or encounters changing forces due to the slight twisting produced by the side mounted drive belt.

After trying several ad-hoc motion control schemes we arrived at a satisfactory software solution to the bridge positioning problem by implementing a Proportional-Integral-Derivative (PID) based algorithm. The PID algorithm accepts as input an error value and three user definable coefficients referred to as P,I and D. In this case the error value is the difference between the current bridge position and the target bridge position. The algorithm directly computes the bridge motor voltage by scaling the error, the time derivative of the error, and the time integral of the error by P,I, and D respectively and then summing the three scaled values. We arrived at values for the P,I, and D coefficients through interactive experimentation.

The Control component also implements two pickwheel operational modes. In the first mode the user can set when the pickwheel is triggered and how many times the the string is plucked per trigger. In the second mode the pickwheel spins continuously and the user can control the speed of rotaion.

The Control components damper control subsystem manages a set of user selectable trigger/release algorithms. These include a mode which allows applying the damper directly, a mode which keeps the damper on the string for a predetermined amount of time after each damper trigger and

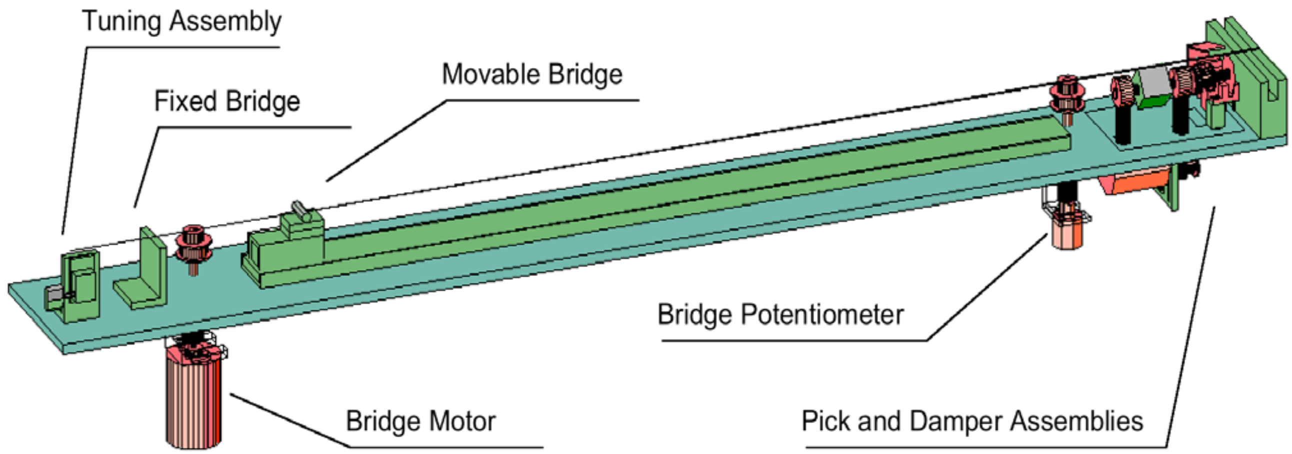



**Figure 2. CAD model of current version**

then automatically lifts it, and a mode where the damper is lifted just prior to the pick wheel pluck and then put down manually or automatically after an elapsed time.

All of the control component variables used to determine the bridge position, pickwheel triggering and damper state are user programmable via MIDI. The Configuration component is responsible for deciding which MIDI messages are routed to each of the control component variables. This logical distinction between control and configuration allows for maximum flexibility in defining user interface modes. By default Note-on pitch values determine the center pitch, pitch bend messages determine pitch offset above and below the center pitch, and note-on velocity messages control bridge velocity. This setup allows a MIDI keyboard to easily control the instrument in the expected way. However the routing configuration can be easily changed to more unusual schemes. For example if the MIDI pitch wheel is routed to the center pitch, the MIDI modulation wheel to the pick wheel velocity, and the note-on/off gate to the damper trigger, the pitch and the pluck rate can be controlled quickly and continuously for interesting sonic results.

The Configuration component also manages the storing and recalling of configuration presets and the system tuning table to non-volatile memory. The default tuning table supports the traditional Western twelve notes per octave equal-tempered system however it can be changed by the user to contain any arbitrary values.

## 3. FUTURE

Future aesthetic and functional improvements are planned. The picking system is designed to be interchangeable, and different actuation systems can easily be swapped in and out. Other mechanisms under consideration for playing the string include bouncing action, bow-like action, rubber and glass wheels and electromagnetics (i.e. EBow).

Software improvements will include the addition of a load balancing and dynamic allocation algorithm, allowing a designated master unit to distribute monophonic and polyphonic note data to the best available unit or units.

By making the units of the instrument modular, we are able to place them into different sculptural and aesthetic contexts. Future designs concepts in which to incorporate the guitar units include a pyramidal structure with integrated speakers and amplification and an anthropomorphic humanoid “Guitar God” robot. Also, on-unit MIDI-controllable lighting effects will be implemented.

## 4. ABOUT LEMUR

LEMUR – League of Electronic Musical Urban Robots – is a New York City based group of musicians, artists, engineers and technologists dedicated to producing robotic musical instruments (http://ericsinger.com/LEMUR). Founded by Eric Singer in 2000, LEMUR received a $30,000 Rockefeller Foundation grant awarded in 2001 to create an orchestra of musical robots. Under this development grant, LEMUR completed four instruments in 2002: the string-based *GuitarBot*, and three percussion-based instruments, *ShivaBot*, *!rBot* and *TibetBot*.

LEMUR is affiliated with Harvestworks Digital Media Arts Center, a non-for-profit organization founded in 1977 to cultivate artistic talent using electronic technologies (http://harvestworks.org).

LEMUR research and development is headquartered at the Madagascar Institute, a Brooklyn-based collective of art stars, geeks, pyromaniacs, insurgents and other misfits dedicated to keeping NYC art spontaneous, hazardous and exciting by means of guerilla street performance, techno-art, carnival rides, kinetic sculptures and a whole lot of stuff blowing up. (http://madagascarinstitute.com).

## 5. ACKNOWLEDGMENTS

In addition to the authors, LEMUR members include Kyle Lapidus, Jonathan Huggins, Clay Lacefield, Jeff Feddersen, Milena Iossifova, Michelle Cherian, Brendan J. FitzGerald, Ahmi Wolf and Chad Redmon

LEMUR would like to thank the following people for their support and assistance: Joan Shigekawa (Rockefeller Foundation); Carol Parkinson (Harvestworks); Chris Hackett and Maureen Flaherty (Madagascar Institute); Red Burns, Tom Igoe and Greg Shakar (NYU Interactive Telecommunications Program); Chico MacMurtrie (Amorphic Robot Works); Scott Wolynski; Dave Liatti; Will Bowen; Heather Martin.

## 6. REFERENCES

[1] Jorda, Sergi. “Afasia: the Ultimate Homeric One-man-multimedia-band” In proceedings NIME02, Dublin, Ireland

[2] Control Tutorials for Matlab. http://www.engin.umich.edu/group/ctm/

[3] Davies, Bill. Practical Robotics: Principles and Applications. CPIC Technical Books, 1997.

[4] McMaster-Carr Supply Company Catalog #108. New Brunswick NJ, 2002.

[5] MSC Insdustrial Supply Company Catalog 2002/2003. Melville NY, 2002.

[6] Walker, John R. Machining Fundamentals. Goodheart-Willcox Co. Inc., Tinley Park IL, 2000.

**Figure 3: Completed GuitarBot in its native urban environment**

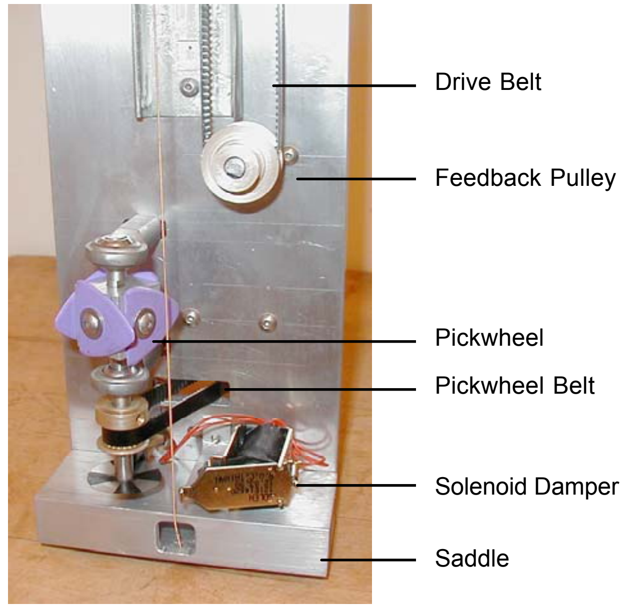



**Figure 4: Detail of Pickwheel and Damper assemblies**

# An Interface for Real-time Classification of Articulations Produced by Violin Bowing

Chad Peiper†, David Warden†, Guy Garnett‡
†Department of Computer Science, University of Illinois Urbana-Champaign
‡School of Music, University of Illinois Urbana-Champaign
{peiper,warden,garnett}@uiuc.edu

## ABSTRACT
We introduce a software system for real-time classification of violin bow strokes (articulations). The system uses an electromagnetic motion tracking system to capture raw gesture data. The data is analyzed to extract stroke features. These features are provided to a decision tree for training and classification. Feedback from feature and classification data is presented visually in an immersive graphic environment.

## 1. INTRODUCTION
The CyberViolin project seeks to capture aspects of violin playing in order to better understand how expert violinists achieve the effects they are heard to achieve. Ultimately, this information will be used to facilitate human-computer interaction.

There have been several initiatives involving the use of sensor data to describe musical performance. In [8], a model is described for the mapping between gestural primitives and performer processes. A partial survey of instruments created which allow sensor data to manipulate the sound characteristics produced is given in [6]. Virtual musical instruments provide a gestural interface for the mapping of movement to sound [7]. In the eviolin project, a synthetic instrument is created, efficiently mapping gestural data to the production of particular sounds [9]. In [4], resonance model filter parameters are changed based on motion sensor data, allowing an instrument to be played through the resonance model of another sound. An instrument model allowing alternate input devices is described in [10]. In contrast to systems using sensor input to map gestures to sound, the cyberviolin project records gesture data for analysis in order to provide objective feedback to the user. The user may then interactively adjust his/her performance.

Bow movement is one significant part of violin playing. Characteristics of this gesture include bow pressure, velocity, and position. Used together, these parameters result in different articulations; controlling these characteristics is essential to the production of good violin tone.

In order to analyze these properties a module allows the computer to identify and measure different articulations of bow strokes. Additionally, this technology is designed to be adapted to a pedagogical mode. Ideally, a violinist in an immersed virtual environment would be able to interact with a virtual world that responds to the articulations of the player. As a pedagogical tool, a violinist in the practice room could be monitored by a virtual teacher who could identify incorrect articulations in real-time.

## 2. DESCRIPTION OF THE SYSTEM
The software system for the classification of violin bow strokes (articulations) uses an electromagnetic motion tracking system with two sensors. One sensor is attached to the back of an acoustical violin and the other is attached to the frog of the bow. There are a variety of articulations that have developed historically as aspects of violin performance. These include the following: *détaché, martelé, staccato, spiccato,* and *legato*. The difference between these various strokes is the result of subtle variations in gestures, which are discrete and measurable.

*Détaché* is the most common bow stroke in string playing. It is usually played in the middle part of the bow, with one note per stroke. Although the spelling of the term suggests a detached articulation, the bow never stops between strokes. *Legato* is defined by the slurring of two or more notes on one bow stroke. Contrary to the continuous bow motion of the *legato* and *détaché*, *martelé* and *staccato* require decisive attacks and releases.

*Martelé* is characterized by a percussive stroke produced by pinching the string with the bow before drawing the bow. The pinching or *martelé*-accent results from the weight of the bow completely resting on the string with additional weight applied by the relaxed, hanging arm. In each stroke, almost all of this weight is suddenly released from the string, but only for a moment while the bow is in motion. Upon completion of a quick stroke (bow speed), the weight of the bow and arm is again pressed against the string resulting in discontinuity between strokes.

The *spiccato* is performed by a natural bouncing of the bow from the string in the middle portion of the bow. The exact location varies from bow to bow dependent on the location of the "bouncing point" on the stick. Performing an accelerating *détaché* in the middle-third of the bow will result in *spiccato*. Note that both *détaché* and *spiccato* are performed with a continuous bow motion (i.e., no stoppages).

The term *staccato* denotes two or more short strokes on the same bow; performing two *martelé* strokes on the same bow results in a *staccato*.

The CyberViolin system for analysis of bowing techniques consists of the following components: bow stroke characterization, decision tree induction, and bow stroke classification. Decision tree induction allows classification rules to be easily generated and compared with expectations from domain experts.

Bow stroke identification involves a representation of each stroke in terms of a dataset that translates physical activity into various classes to facilitate computer recognition. The next step is the creation of a tree structure that enables the computer to recognize the various streams of data that constitute the successful performance of each stroke. The final step is the sifting of the data through the tree, a process that results in the real-time identification by the computer of performance gestures.

The system can be used in two modes: training and classification. The two modes of operation are illustrated in the following figures:

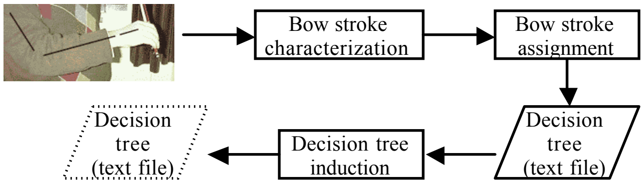



**Figure 1: Training process**

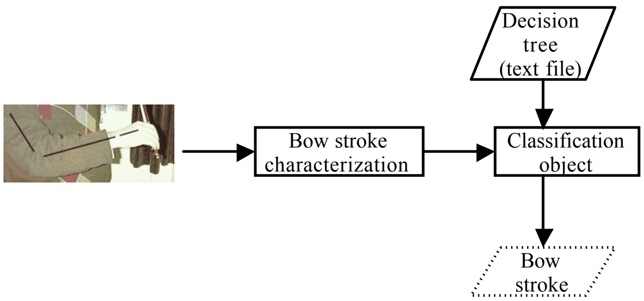



**Figure 2: Classification process**

The training mode is used to construct the decision tree, which the computer will use in the real-time classification. In order to facilitate the computer's ability to classify different articulations, a series of datasets that characterize each stroke must be provided. This is an example of supervised learning in that the programmer must provide the definitions for each aspect of a given stroke. This is the necessary prerequisite to enabling the computer to assess any given input. The more training data we provide, the greater the accuracy of the classification. In addition, there are algorithms designed to improve the structure of the tree. For example by combining branches we can create a more compact structure, and by removing unwanted branches created as a result of noise in the dataset, we can decrease the depth of the overall tree. In the classification mode, we traverse the tree to identify a class label.

Decision trees provide a natural mapping from feature data to classification rules. Unlike other real time classification methods, such as neural networks, these rules can be compared with expectations from domain experts. Since the decision trees generated in our system are relatively small, they can be easily modified to further simplify the tree and improve the classification accuracy. One of the major disadvantages of the decision tree approach is that tree induction algorithms have a poor scalability in terms of both the size of the training set and the number of attributes. While additional data would be beneficial, the relatively small number of attributes under consideration reduces the requirement for large data sets. In addition we have the option of manually editing and adjusting the tree as we see fit.

## 3. IMPLEMENTATION OF THE SYSTEM

### 3.1 Bow Stroke Characterization

For the purposes of our classification system we define a bow stroke as the path the sensor on the bow travels between consecutive changes in bow direction. The most fundamental classification of bow stroke is “up-bow” or “down-bow”. Since the violin sensor is mounted along the axis of the strings, when the bow-sensor, mounted near the performer's hand, approaches the strings the distance between the two sensors is small. Therefore, an up-bow is defined by a decrease in the distance between the bow and violin sensors, while the down-bow is the reverse. Whenever movement is detected in the direction opposite of the current stroke in magnitude greater than a threshold of 0.4in a bow change is detected.

Initially the parameters used to characterize the bow strokes were: the distances between the two sensors at the beginning (D1) and at the end (D2) of the stroke, the length of the path of the bow sensor (L) computed from D1 and D2, and the average speed of the sensor (V). The most accurate measurement possible, given this sparse set of parameters, was approximately 73%.

Adding additional characteristics of bow movement has increased the accuracy of this evaluation: frequency of bow change, acceleration or deceleration within a stroke, continuity of motion between strokes, bow position (middle, upper, lower), number of changes in a single coordinate (stroke similarity), lack of movement within a stroke (stoppage). The system's design allows for the easy addition and removal of characterization parameters.

In order to add additional features, code specific to that characteristic must be written. Several articulations must then be performed in order to record how this characteristic varies among articulations. Once this is done, numerical or nominal data can be presented to the decision tree program. The ability of the additional parameter to distinguish among articulations under consideration can be determined and can then be evaluated. An increase in accuracy indicates the addition was successful.

Other parameters were considered but did not lead to an increase in accuracy during our experiments. These characteristics included the change in azimuth, elevation, and roll relative to the violin.

The bow stroke characterization module is implemented in an ANSI-C++ program using the FreeVR library [2]. The library provides a programming interface to more easily obtain calibrated sensor data and render three-dimensional objects. The data is gathered from position sensors on the bow and violin in the NCSA CAVE™ Autonomous Virtual Environment [11].

The CAVE™ consists of a 10'x10' area with images projected in front, to both sides, and below the user. Shutter glasses allow images to appear in three dimensions. The special tracked glasses and CAVE™ wand are both attached to electromagnetic sensors that allow the computer system to know where within the CAVE™ each is located and what its orientation is. It supports the use of pulsed DC magnetic sensors which report position as well as azimuth, elevation, and roll. The environment is free of metal objects that would reduce sensor accuracy.

The raw position data can be written to a text file for later analysis. The feature data is also sent to a file (training mode) or sent to the classification object (classification mode). Each row in the file created as part of the training mode is a comma-separated list of decimal numbers that characterize a single bow stroke. The user must identify (supervised learning) each type of bow stroke manually at the beginning of each file of the training set.

### 3.2 Decision Tree Induction

The decision tree induction module is a stand alone C++ program, which, given a text file containing the training dataset, builds a decision tree and stores it in a separate text file. The bow stroke classification object accepts the file as input and uses the decision tree to classify the bow strokes. The decision tree induction algorithm is a version of the classic ID3 algorithm, which works only with discrete values and creates very shallow trees (the original ID3 creates binary trees and works with continuous values). The shallow decision trees have very high classification performance and are easier for manual modifications. The algorithm is described in detail on page 56 in [1] and page 285 in [2]. "Information gain" is used as the attribute selection measure. Because this algorithm only works with discrete values, the range of values for any given stroke must be precisely defined before running the algorithm. Entropy-based discretization [1] [2], is used because it works best with the decision tree induction algorithm being employed.

With each iteration, the samples are split across value of the attribute which results in the least randomness among the resulting partitions. The process is repeated until the attribute list is exhausted or until a partition has only members of a single class. The nodes are labeled with the most common class present in that partition.

The attributes are selected from among the extracted features (such as velocity), and the classes correspond to the different articulations.

The command line parameters of the decision tree induction program are:

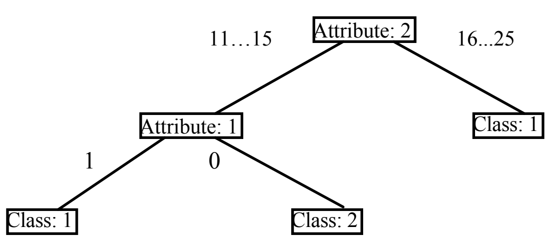
dtree input_file [output_file ] [delta]

- input_file is the file containing the training set.
- output_file is the resultant decision tree exported by the function.
- delta represents a decimal number between 0 and 1 which is used in the stopping condition of the entropy-based discretization. The smaller the value, the smaller the ranges created.

The program evaluates the classification accuracy. 2/3 of the training set is used for decision tree induction and 1/3 is used to test the classification accuracy of the constructed tree.

## 3.3 Bow Stroke Classification Object

The bow stroke classification object (figure 3) acquires the extracted feature data and processes it through the decision tree. The classification object then returns the corresponding articulation. Upon instantiation, the classification object is initialized with a text file containing the decision tree. Each row in the file represents one node (figure 3). This module is implemented in C++ and is integrated with the bow stroke characterization code.




```
Attribute: 2
11…15: Attribute 1
          1: Class: 1
          0: Class: 2
16…25: Class 1
```

**Figure 3: Textual representation**

## 4. FEATURE DETECTION PROCESS

In order to gather the required data, two sensors are used in a CAVE™ Autonomous Virtual Environment. Each sensor provides information on its location in 3 dimensional space as well as the azimuth, roll and elevation to which it is oriented. In the CAVE™, this information is reported approximately 30 times per second, with an accuracy of approximately 1/4in.

Once the information has been captured from the sensors it is translated from the absolute coordinate system into a system relative to the location and orientation of the violin. This allows the bow movement to be correctly recorded and compared even as the user moves the violin about the area. If the bow is not within range of the violin, data is not recorded.

Raw data is recorded in a temporary data structure until an entire bow stroke can be analyzed. Since a stroke may stop in the middle, as in the case of a *staccato*, a stroke is not delineated until the next stroke is begun. This is determined from movement in the opposite direction beyond a predetermined threshold sufficient to account for any unintentional movement and sensor jitter.

Most basic feature information is determined at the stroke level. This information includes the bow direction (up or down bow) velocity, minimum and maximum position in each dimension, the time in the stroke when respective minima and maxima are achieved, the number of times the bow is stopped during the stroke, continuity, and acceleration. Acceleration is recorded for several discrete segments of bow used in any particular stroke. A violin simulator that reads in raw position data from a file aids in feature development.

The features of several bow strokes are also recorded in a temporary structure. Additional features can be reported based on the similarity of stroke properties over a sequence of several strokes. Strokes are also paired together before being sent to the decision tree for classification.

As there are limitations on the sensor and characterization system, the attribute characteristics may differ to some degree from expectations of characteristics provided by experts in the domain. Experiments were conducted to determine the benefit of various attributes by examining the performance of the decision tree when these attributes are considered.

Once information for all features is gathered, those features or derivations, which provide the most information, are selected for the decision tree. Those features typically include, the velocity, change in distance between the frog of the bow and plane of the violin strings, number of stoppages, continuity, acceleration for first and middle segments, and similarity of bow usage. A conversion program takes the features and arranges the output as a list of comma-separated values.

The decision tree generator takes a set of sample data and produces a hierarchical system for classifying a stroke into an articulation based on the values of provided attributes. The accuracy of classification (based on withheld data) on data sets containing several different articulations is shown in Figure 1. While some of the data sets have modest accuracy, there is an explanation for the results. When only *détaché* and *martelé* articulations are considered, 100% accuracy is obtained. The attributes considered by the decision tree help to discriminate between these articulations. The essence of *détaché* is that the bow never stops between strokes (continuity) whereas *martelé* contains discontinuity.

All articulations have characteristics that are derived from either the *détaché* or *martelé* stroke. For example the *legato* and *spiccato* are both continuous in the manner of *détaché* whereas the *staccato*, as mentioned earlier, is essentially multiple *martelé* stokes contained within a single bow.

There are two problems that result from such similarity of articulations. First, the decision tree can misclassify similar articulations when devoid of distinguishing attributes; it also requires a greater amount of data to differentiate articulations that share common attributes. Second, since a stroke is defined as a change in bow direction, it is difficult to take advantage of distinguishing attributes within a single stroke. It is also difficult to consider attributes aggregated over several strokes such as the number of bow changes over time. This results in an insufficient amount of data available to the decision tree.

*Staccato* should be identifiable in a single stroke; the number of stoppages uniquely distinguishes this articulation. However, when several elements of a *staccato* articulation are aggregated to the point of a bow change, it appears similar to a *martelé*. Likewise *détaché* and *Legato* overlap with the exception that there will be fewer *Legato* strokes over time, which is only reflected to some degree in the velocity. Multiple stroke attributes such as examination of the similarity of bow usage among strokes increases accuracy, however the amount of data required by the decision tree is also increased. Given these constraints and the ability of the system to identify the two most general articulations the results are reasonable.

**Table 1: Articulation classification accuracy**

| Articulation | Accuracy |
|---|---|
| *martelé, détaché* | 100% |
| *martelé, détaché, legato* | 85% |
| *martelé, détaché, spiccato* | 81% |
| *martelé, détaché, staccato* | 76% |
| *staccato, spiccato* | 75% |
| *martelé, détaché, spiccato, legato, staccato* | 71% |

Once a decision tree is generated, it can be supplied to the violin program. Given the same set of attributes supplied to the decision tree generator, the classification routine will provide the classification reported by the tree. If desired, the tree could be modified by hand. The reported classification can be used by visualization code (see section 5) or logged into a file. In order to reduce false reports of articulation changes, the reported classification is an average over the five most recent individual stroke classifications preformed by the decision tree.

The feature extraction process only requires simple arithmetic operations be performed whenever new sensor data is received. An advantage of decision trees is their high performance after the training process is complete. Each bow stroke only requires a number of comparisons at most equal to the depth of the tree, which is in turn at most the equal to the number of distinct features extracted. For these reasons, all computation can be done without observable performance latency. In the CAVE™ new data was received thirty times per second, and strokes would be completed after several seconds depending on the articulation.

## 5. CYBERVIOLIN APPLICATION

The cyberviolin application provides a player an environment for the visualization of performance data. The program reports feature data in real time. With a sufficient number of strokes, real-time classification data is also available. All data is archived for later presentation or analysis.

Currently the application offers two modes of operation: record and playback. In either of these modes, graphical feedback is available. For each attribute the respective graph illustrates the value with its corresponding bow stroke. This allows the user to observe how the attributes, interrelated, form articulations.

Static graphs representing pre-recorded data, while informative, are of no real benefit to the player during the actual performance. The aim of this application is to provide the user with real-time feedback. Providing this feedback in the CAVE™ environment creates a virtual interactive mirror allowing the performer to see him/herself objectively. The additional insight afforded in such a system allows the user to adjust and improve their performance in real time.

As an example, consider a user intending to perform a passage with a *spiccato* articulation. If the player mistakenly performs a *détaché* we would like the system to provide feedback to communicate this error and the means to correct it. The report of the detected articulation indicates an error but is devoid of the details necessary to influence a change. Presentation of the distinguishing attribute values to the user can alert the user to the requisite corrections.

The application is currently under development and we hope to report on later versions supporting greater interaction.

## 6. LIMITATIONS AND FUTURE WORK

There are several sources that limit the accuracy of this system. One weakness is the limited amount of information that is available from a single pair of position sensors. If sound analysis data or pressure sensors on the bow were available, they may provide valuable information for feature extraction and subsequent classification.

The precision of the hardware sensors as well as their refresh frequency also has a limitation on the ability to detect features. In testing the CAVE™ environment the precision was clearly superior to other environments with less sensor precision. Sensor error in such environments made feature detection substantially less accurate and difficult to develop. Further improvements could allow for more precise detection of stoppages, discontinuity, and shorter strokes, as they are most dependent on sensor precision. The size, weight, and positioning of the bow sensor challenge the player's ability to perform articulations. A smaller, lighter, wireless sensor would result in a more natural performance, and perhaps more distinct articulations.

Feature detection could be improved with additional identifying features. Features that compare several strokes are one area of development that may lead to improved accuracy. By computing information over an aggregate of strokes, the time period for information provided to the decision tree is increased. This allows for increased accuracy, as an articulation may not be evident for several strokes. Reliance on these features would require articulations long enough to provide data on groups of strokes.

In our current implementation, some feature information is not entirely accurate. Whenever there is a discontinuity or stoppage, there is some period of time when the system is uncertain if playing has resumed due to the time lag used to determine motion. This appears to lead to observable error in some instances correlated with stoppages. One such articulation is the *martelé* stroke, which is generally defined as a percussive bow stroke produced by "pinching" the string with the bow before starting the stroke. When detecting bow changes where the bow is stopped at the end of the stroke, the velocity values were not consistent on the up and down-bow, even when the bow speeds of both strokes were performed similarly. The exact effect of this error on the classification is unknown. It may be possible to reduce this error with more accurate sensors in the future.

The availability of more data would aid in the construction of the decision tree. Since the tree must speculate on unknown information, the more performances the tree is exposed to during training, the more accurate it will be. Spurious or unusual data would also be factored out. While a decision tree is capable of very accurate classification, other AI based classification methods, such as neural networks could also potentially be used.

The user interacts with the program by using the bow in place of the traditional wand pointing device. A graphical interface is projected within the CAVE™ allowing the user to select desired options by directing the bow at the respective location.

Even with future improvements, there will be a level of classification beyond which additional accuracy is difficult to obtain. This is due to the fundamental properties of the underlying music, and limitations in the performer's ability. However, even with these limitations, the cyberviolin project is able to perform classification with a useful level of success.

## ACKNOWLEDGMENTS

Our thanks go to Camille Goudeseune, Integrated System Laboratories (ISL), and William Sherman, NCSA, at the University of Illinois at Urbana-Champaign.

## REFERENCES

[1] Mitchell T. Machine Learning. ISBN 0-07-042807-7, McGraw-Hill, 1997.

[2] Han J. and Kamber M. Data Mining: Concepts and Techniques. ISBN 1-55860-489-8, Morgan Kaufmann, 2000.

[3] http://www.freevr.org

[4] Garnett G., Goudesuene C., Johnson T. "Resonant Processing of Instrumental Sound Controlled by Spatial Position" to appear, 2003.

[5] Trueman, D., P. Cook. "BoSSA: The deconstructed violin reconstructed", Proc. 1999 Intl. Computer Music Conf. San Francisco: Computer Music Association 232-239.

[6] Trueman, D. "The Infinite Virtual Violin:" in Reinventing the Violin, http://www.music.princeton.edu/~dan/rtv/, Chap 3.

[7] Goto, S. "Virtual Musical Instruments: Technological Aspects and Interactive Performance Issues", Trends in gestural control of music, ed. M. Wanderley and M. Battier, Paris: Ircam Centre Pompidou, 217-230.

[8] Choi, I. "Gestural Primitives ad the context for computational processing in an interactive performance system", Trends in gestural control of music, ed. M. Wanderley and M. Battier, Paris: Ircam Centre Pompidou, 139-172.

[9] Goudesuene, C. Composing with Parameters for Synthetic Instruments, Ph.D. dissertation. Dept. of Music, University of Illinois at Urbana-Champaign, 2001.

[10] Serafin, S., Dudas, R., Wanderley, M., and Rodet, X. "Gestural Control of a Real-Time Physical Model of a Bowed String Instrument", Proc. Intl. Computer Music Conf, 375–378, 1999.

[11] http://cave.ncsa.uiuc.edu/

# Convolution Brother's Instrument Design

Zack Settel
Zeep.com
4319 Esplanade, suite 1
Montral QC, CA
zack@zeep.com

Cort Lippe
Department of Music
Hiller Computer Music Studios
222 Baird Hall
Buffalo, NY, USA 14260
lippe@buffalo.edu

## ABSTRACT

The subject of instrument design is quite broad. Much work has been done at Ircam, MIT, CNMAT, Stanford[1] and elsewhere in the area. In this paper we will present our own developed approach to designing and using instruments in composition and performance for the authors' "Convolution Brothers" pieces. The presentation of this paper is accompanied by a live Convolution Brothers demonstration.

## 1. INTRODUCTION

To be sure, instrument design has always been a rich source of inspiration and discovery for the authors. We continue to be fascinated by the relationship between instruments and the music written for them. The piano is worth considering in this light. It was born from an idea: an improvement to the harpsichord, which introduced the concept of note-independent dynamic range to keyboard music. Repertoire for the instrument only emerged once the piano had been in the hands of performers and composers for a certain time, after its appearance around 1700. In short, the sound and nature of the piano inspired or induced both pianists and composers to discover new kinds of music for the instrument: "piano music" (i.e. the kind of music written by Mozart, Chopin or Debussy). One could say that the instrument had a particular innate musical potential, based on its design, just waiting to be tapped. The same appearance of instrument-inspired music followed the introduction of the saxophone, or electric guitar. The playing and music of Charlie Parker and Jimi Hendrix cannot be separated from their respective instruments.

Harry Partch and John Cage are two excellent examples of composers whose work deeply integrated instrument design in the compositional process. Harry Partch produced compositions that integrated new scales, notation, and instruments (including playing techniques). The resulting music remains quite distinctive and refined; his music is easily identified and consistent with his other works. Even more impressive is Cage's works for prepared pianos, where Cage takes the piano as an instrumental point of departure, and builds a completely new instrument with an extremely wide expressive range (discussed later). Very little of the piano's original sound is retained. Rather, the instrument is completely revisited, based on the concept of an ensemble of percussion instruments under the control of a single musician. Cage's works for prepared pianos represent an unparalleled example of creative expression, innovation and elegance in composition. While the "instrument" and resulting compositions are in many ways (starting with the timbre) unprecedented, much of the underlying compositional "implementation" (i.e. instrumental technique, and writing/scoring) was squarely based on the tradition and practice of piano music. From our point of view, in terms of compositional approach, the prepared piano instrument serves as a vehicle for discovery, a source of ideas, inspiration and compositional curiosity. It is at the root of the inspiration of the music composed for it. In our approach to composing, the choice (or development) of instrument can be at the root of compositional inspiration, and is a crucial stage in the rendering of the piece. It is as if the music for a piece were "reverse-engineered", given the instrument it is composed for. Otherwise said, in inventing an instrument, one invents a piece. Thus, we attach a great deal of importance to instrument design, and have developed an approach to composing in this way.

[1]The following individuals are quite active in this area: B. Rovan (IRCAM), T. Machover (MIT), M. Mathews (Stanford) and D. Wessel (CNMAT).

## 2. DEVELOPING INSTRUMENTS

One of the most important considerations we have come to recognize in designing instruments for composition is what we would call "instrumental expressive range"; i.e. to what extent the designed instrument is able to transmit the "musical message" of the performer, and to what extent the instrument is able (versatile enough) to handle a wide range of musical messages. The ideal instrument would be totally responsive and completely transparent, consistently translating the musician's playing actions instantly into sound, arbitrarily without interference or distortion. Though not ideal, it is worth referring to Cage's prepared piano instrument; sure, it lacks the stylistic versatility of the clarinet, but it is remarkably transparent, capable of transmitting even the finest playing gestures (based on piano technique, acquired over the years).

### 2.1 Instrumental expressive range

The expressive range of a given instrument tends to determine its musical potential with respect to style (i.e. types of musical expression that it lends well to), or compositional application. While the piccolo is a fine ensemble instrument, it is rarely chosen by composers for solo works; to imagine an evening of piccolo music is terrifying. The violin, on the other hand, lies at the other (complex) end of the "instrumental complexity" continuum, and would be a fine instrument to be marooned on a deserted island with. The number of music cultures around the world that include violin, or violin-like music is substantial. Based on its design, the violin is capable of a vast range of musical expression. The same can be said of other instruments of a similar degree of complexity, such as the saxophone or clarinet. Cage's prepared piano, mentioned above, would occupy a position on the instrumental complexity continuum (shown below) much closer to the violin than to the claves.

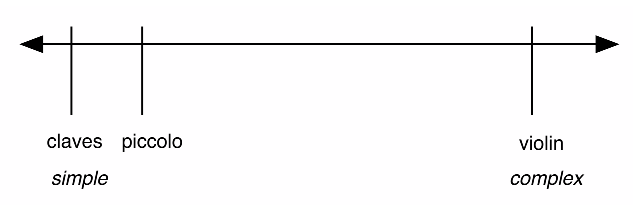



**Figure 1: Instrumental complexity**

## 2.2 Instrument structure

In general, a complex instrument holds the greatest potential for musical (including, but not limited to compositional) discovery-especially instruments that are unusual, since the music they "embody" tends to emerge without being subject to habits of hearing or musical expectation. What makes for a complex instrument?

1. High resolution and wide range with respect to dynamics, timbre and pitch (when applicable), though a wide range is less important for pitch.

2. "Instrumental depth" or responsiveness: the degree to which the instrument can consistently render a sonic response specific to a particular instrumental manipulation.

For example, the claves don't qualify as a complex instrument. While they have excellent dynamic range and resolution, the timbre never really changes. You can bang on them any which way you please, but the resulting sound is pretty much always the same.

The instruments we design for the "Convolution Brothers" pieces tend to be complex, and typically involve acoustic instruments with electronic extensions for signal analysis, signal processing, and sound generation. In each case, the acoustic instrument is used as a sound source, and a controller providing control signals derived via event detection. Here's how it looks on paper:

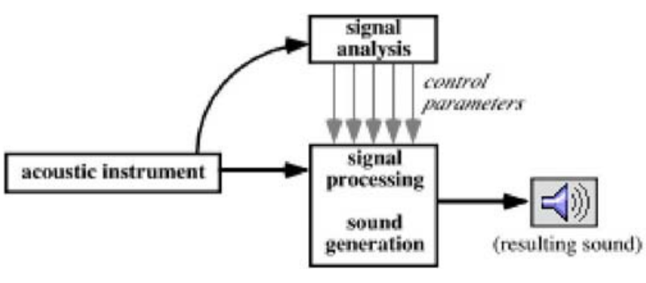



**Figure 2: Electroacoustic instrument**

The use of acoustic instruments in the design provides a "built-in" degree of instrumental complexity and richness, and a high-resolution "musical interface" for controlling the instrument via event detection. It's a good way to get around the limitations of (usually low and) finite-resolution controllers (i.e. all electronic controlling devices, and particularly those with 7-bit MIDI implementations). In this way, the control resolution can approach that of the audio (e.g. the envelope follower, that produces a 32-bit intensity parameter). Additionally, the acoustic instrument's dynamic spectrum (or fundamental aspects of it) is often present in the instrument's output, usually transformed. Thus, the player's technique and initial expression, as manifest in the dynamic spectrum, are retained. An excellent simple example of this is singing into a ring modulator; while the output sound is completely transformed, all changes (no matter how fine) to the singer's spectrum at the input of the ring modulator will produce corresponding degrees of change in the modulator's output spectrum. From this point of view, the processing technique is highly transparent to the singer. Finally, the acoustic instrument's spectrum can be combined with additional sound sources, whose behavior is analogous in one or more ways to the acoustic instrument's behavior, and a hybrid sound results. For example, an FM generator's frequencies can be mapped from the acoustic instrument's detected pitch, while the FM generator's dynamic spectrum is spectrally gated (as in vocoding) by the acoustic instrument's spectrum. The resulting sound's behavior is quite similar to that of the acoustic instrument. However the spectral content can be radically different.

## 2.3 The instrument and its context

The higher the degree of an instrument's sophistication (i.e. the instrument's ability to produce a wide range of timbral responses, based strictly on the musician's input), the less it needs to be updated via "external" (non-player based) parameter or configuration changes. Typically, these kinds of "external" updates are based on the player's position in the score (or musical situation), and executed via manual triggering, control tracks, or score followers. With the ability to design a sophisticated instrument and predict how it will respond (sound) to a performer's particular musical input, the performer's score can be written with the instrument's response in mind, and material can be written in the score to provoke certain responses from the instrument when played by the performer. No score following is required. Instead, the performer's score serves as a sort of control track, containing a sequence of events that will ultimately determine (via event detection and parameter mapping) how the instrument is to sound . This relationship between the instrument and the music intended for it is central to our approach to instrument design and score crafting (or improvisation applications).

# 3. IMPROVISATION

Designing instruments for use in specific compositions is already a lot of work. However, the realm of possibilities is limited by the demands of the given composition during the course of its execution; the requirements of the instrument are "defined" as a single sequence specifying the instrument's behavior and state at any given moment during the piece. The sequence itself often consists of references to patches to be activated at a particular time, with associated context-specific data, such as harmonizer transpositions or what have you. Score following is a well-known technique which is employed for this kind of functional sequencing, and can be effective as long as the sequence of functionality for the instrument is pre-defined-and as long as there's someone watching the score follower. Designing instruments for use in compositions with "structured improvisation" is more work; the instruments are not designed for use in a highly circumscribed through composed musical situation. Rather, they are designed to be "run away with", in the "hands" of a virtuoso performer.

The amount of complexity required of an instrument for improvisation is significant-especially if the player expects to play for more than five minutes without getting bored. The trouble with our finite electronic instruments is that they usually only go as far as the imaginations of the designers. Unlike an empty oil barrel, you can bang on a synthesizer any number of ways, the resulting sound will always be within the realm of its specified, "programmed" behavior, intended or not. The oil barrel delivers an infinite number of variations for each "whack". This brings us back to the above discussion on instrumental depth: If you are going to have to spend some time on a desert island with an instrument, or are planning to improvise, you better make sure your instrument is not easily exhausted. Designing instruments for arbitrary style-independent improvisation situations is not obvious. Even a saxophone would have a hard time living up to such an ideal. But let's say that even when the style is somewhat defined, such as the "out there" experimental electroacoustic free-jazz, like the kind of stuff you hear at festivals in Victoriaville, Quebec or in Vandoeuvre, France, a "suitable" instrument's potential range of expression is still enormous.

## 3.1 The instrument interface

The instruments described above, particularly the ones used for virtuoso passages in the Convolution Brothers' works, serve as good points of departure for improvisation instruments. Rather than going into endless descriptions of variations on the coupling/combining/piloting of unit generators, we will emphasize an additional aspect of the improvisation instrument: the interface. Unlike the instruments in "through composed" works, that resemble timelines more than anything else, the interface for an improvisation instrument needs to be intuitive and allow for on-the-fly enabling of the instrument's features (e.g. percussion sound triggering, reverse-gate reverb, triggered harmonization, sampler source material choices, etc.). Everything (all the DSP) must be able to run at the same time! The interface also needs to be modularized, so that subsets of it can be mixed and matched. Below is the interface that the authors have been developing over the past few years, and are currently using for improvised performance:

The interface provides two input meters, corresponding to DSP sends A and B. The DSP sends feed their respective unit generators. Many, but not all the types of unit generators in DSP A exist in DSP B. Using this "parallel" DSP implementation, the same input signal can be routed to both DSP A and DSP B, for extensive processing. Alternatively, two independent inputs can be routed to either DSP send, and thereby be processed independently of each other, as in the case of a duo. The functional parts of the interface are broken down into the following parts:

### *3.1.1 Recall via PRESET*

Contains a preset object, which allows the state of all the principal graphical parameter displays on the interface to be captured and recalled, snapshot fashion. This feature is invaluable, allowing for last-minute and/or in-concert modifications/additions to the instrument.

### *3.1.2 "Sub States"*

Consists of labeled switches (1A-16A) that represent enabled/disabled sub states (audio and control routing and DSP states, defined elsewhere, see fig 4), in which arbitrary control sources for the unit generators of DSP A are defined. These sources include parameter data, control algorithms, and event detection control-stream mappings to DSP parameters. For example, sub state "3A" activates a sampler in DSP A, and routes the triggers and amplitude envelope (from the event detector) to the sampler. Sub states are constructed add-hoc, as needed, addressing one or more unit generators; there is no one-to-one correspondence between sub states and particular unit generators. Thus, any combination of sub states can be active, however, certain enable/disable the same unit generators, so care must be taken when cooking up combinations of them. An input meter displaying the input level of the each of the two DSP sends is included.

### *3.1.3 Event Tracking Assign*

Routing of the computer's three audio input signals to the event detector, DSP A and DSP B is specified here. There is only one tracking block to which any or all of the audio inputs can be assigned.

### *3.1.4 Special Unit Generator Parameters*

Particular unit generator parameters, including sample names, are located here for quick access in performance, or convenience during pre-concert preparation.

### *3.1.5 Output Mixer*

Provides access to output gain levels for all the unit generators-a mixer of sorts. The "trim" slider controls the gain of the overall DSP output, and is quite indispensable since its value often varies from preset to preset in order to maintain a uniform global "mix" level across them.

### *3.1.6 DSP States*

Displays the state of the unit generators for both DSP A and B. For convenience and rapid access to parameters, external MIDI controllers can be used to control certain parameters of the interface such as feedback gain levels, trim, and master gain. Typically those parameters are related to mixing, not to the "instrument" per se.

The preset bank mentioned above provides instant recall of any combination of sub states. The fundamental behavior and functionality of the improvisation instrument(s) is defined in the sub states, and combinations of them. New sub states are added to the instrument from time to time, as necessity or inspiration has it. But for the most part, designing an instrument involves the combination of sub states rather than the creation of new ones. Below, a code example of a sub state defining the "gap-fill" reverb mentioned earlier. The performer's amplitude gates his send level to the reverb unit, and inversely gates the reverb proportion (dry/wet mix) in the unit's output.

# 4. USING THE INSTRUMENT

Normally, only a handful of presets are really used since each preset represents a complex instrumental configuration that can be used for some time before "timbral exhaustion" occurs. But minor changes to these basic presets are often made either before, or during, performance-mostly for mixing balance purposes. Thus, the majority of the many presets shown above are just modified copies of the basic handful. On stage, the instrument spends most of its time waiting for, responding to, and combining with the microphone input. Some preset changes are recalled from time to time when major musical "gear shifts" are required (such as going to a specific "bell triggering" effect for a moment).

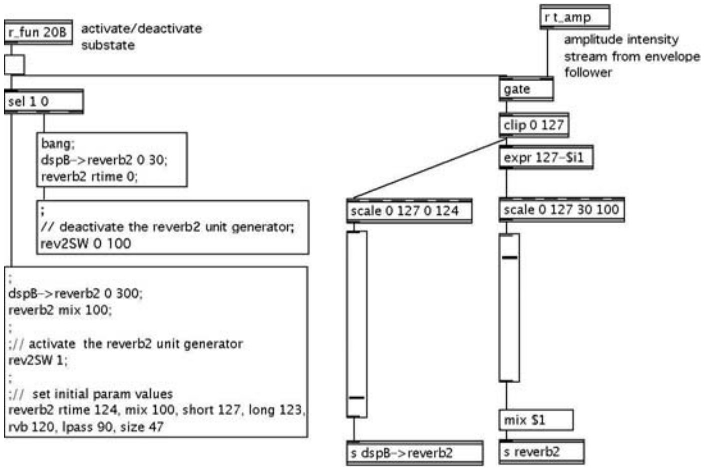



**Figure 3: Code example of a "sub state"**

However for the most part, it is the musician who evokes the timbral changes in the instrument simply through his modes de jeux, and not through major changes to the instrument's topology, signal routing or control mapping.

## 5. CONCLUSION

The strong emphasis on the "instrument" in this paper reflects our longtime preoccupation with composing with instruments for live performance. In our approach, the instrument is present in the composition process at its very inception-as a source of musical expression, wonderment, curiosity, discovery, and inspiration. The approach to electroacoustic instrument design presented in this text provides for the invention and reinvention of new instruments, which themselves can lead to new musical expressions, from which compositions eventually emerge-which, in turn, inspire the invention of still newer instruments. This iterative process embodies composition and performance in a continuous cycle of creation and expression, where the underlying techniques serve simply as a vehicle for the imagination and as a source for the musical spirit.

## 6. ACKNOWLEDGMENTS

Our thanks to Miller Puckette, Bennett Smith and David Zicarelli.

## 7. REFERENCES

[1] C. Lippe. "Getting the Acoustic Parameters from a Live Performance" . In *Proceedings of the 3rd International Conference for Music Perception and Cognition*, pp. 328-333, Liege, 1994.

[2] C. Lippe. "Music for piano and computer: A Description". In *Information Processing Society of Japan SIG Notes*, vol. 97, no. 122, pp. 33-38, 1997.

[3] R. Rowe. *Interactive music systems: machine listening and composing*. Cambridge, Mass.: MIT Press, 1993.

[4] D. Wessel. "Timbre Space as a Musical Control Structure" Computer Music Journal, vol. 3, no. 2 , 1979.

[5] D. Wessel, D. Bristow, and Z. Settel "Control of Phrasing and Articulation in Synthesis" In *Proceedings of the International Computer Music Conference*,1987.

[6] D. Zicarelli. "Music Technology As A Form of Parasite". In *Proceedings of the International Computer Music Conference*, pp. 69-72, 1992.

# A Component Model of Gestural Primitive Throughput

Insook Choi
Composer
is.choi@sympatico.org

## ABSTRACT
This paper suggests that there is a need for formalizing a component model of *gestural primitive throughput* in music instrument design. The purpose of this model is to construct a coherent and meaningful interaction between performer and instrument. Such a model has been implicit in previous research for interactive performance systems. The model presented here distinguishes gestural primitives from units of measure of gestures. The throughput model identifies symmetry between performance gestures and musical gestures, and indicates a role for gestural primitives when a performer navigates regions of stable oscillations in a musical instrument. The use of a high-dimensional interface tool is proposed for instrument design, for fine-tuning the mapping between movement sensor data and sound synthesis control data.

## Keywords
Performance gestures, musical gestures, instrument design, mapping, tuning, affordances, stability.

## 1. INTRODUCTION
This paper is theoretical and propositional to the extent that the explication of the proposed model is founded upon the theoretical basis of a perception-action oriented performance paradigm with respect to the long-standing tradition of musical performance practice. The proposition of *gestural primitive throughput* is much in line with the research concerns expressed by other researchers in so far as to multiparameter mapping strategies and feature extraction. The instrumental gestures defined in Cadoz [1] are more comprehensive and the research challenge is broader than mapping or feature extraction. A formalized approach for a component model can be considered in which the results from the two research areas can be brought together into the structure of a signal pathway. The purpose of such consideration is first to disambiguate the relationship between performance gesture and musical gesture. Second, once disambiguated the throughput model provides loci to host a designed configuration of the structure of the signal processing between them.

From the evaluation of our previous research we determined that the throughput model was implicit in the implementations of interactive signal pathways, but its role was not fully articulated. The throughput model is intended to provide a basis for robust tuning of gesture-based performance systems, calibrating gesture-sensitive devices and interactive displays to optimize the affordances for performers' movements. This calibration is analogous to the tuning of a musical instrument which enables a performer to sustain intonation during a performance, especially when the performer is required to manually override the instrument's native tuning limitations.

This paper first presents a model of symmetry in musical instrument throughput, then introduces gestural primitives, then discusses their relevance for mapping strategies in the design of music instruments and interactive presentation systems. System tuning using a model for gestural primitive throughput is intended to be extensible from musical instruments to multi-modal systems, for example the calibration of graphical displays as described by Goto [2].

## 2. SYMMETRY AND ASYMMETRY IN MUSICAL INSTRUMENT THROUGHPUT
Consider as a generalizable abstraction that musical instruments may be represented using a throughput model comprised of two gestural subsystems: a movement-sensing system and a sound-generating system. The movement-sensing system provides transducers that respond to *performance gestures*. The sound-generating system provides excitatory and resonating bodies to produce *musical gestures*. In traditional musical instruments these two systems are in immediate proximity to one another, with energy transfer from one to the other by direct contact of physical materials or by materials that may be one and the same. Figure 1 illustrates the symmetry between the two gestural subsystems; dotted lines indicate material regions and boundaries in various instruments. For example in a clarinet the keys and the reed are sensors; the reed is also an excitatory body; the clarinet body and resonator are the same tube. The functional components in Figure 1 are common to all musical instruments and are found in various arrangements according to differences in design.

One of the technological achievements of musical instruments is the coupling of two gestural subsystems into a single physical device. Disambiguating the relationship of two coupled subsystems can assist in the clarification of unresolved issues in the study of musical performance gestures. Two such issues may be referred to informally as the "handwriting recognition problem" and the "causality problem." Briefly, the handwriting problem undertakes a search for elements and features of gestures, their boundaries and generalizability. A question remains whether to measure such units and features at the performer, the instrument, or the sound. The causality problem attempts to identify gestural units on the basis of musical structures that are notated in a score or produced in performance, leaving the question of which comes first, the movement or the sound. The present lack of clear definition of "what is a gesture" in musical performance can be related to confounding aspects of these two problems. Cadoz and Wanderly provide a framework for disambiguating gestural relationships in [3].

Study of the transfer of movement in a signal pathway provides a re-orientation for these questions. Gestural primitives are an impetus; they can be detected in the transfer of movements; they are not proposed as units or segments of gestures. Movement transfer occurs at locations of transducers. The throughput model proposed in Figure 1 shows two sites of transducers, first from the performer to the physical instrument, second from the physical instrument to sound.

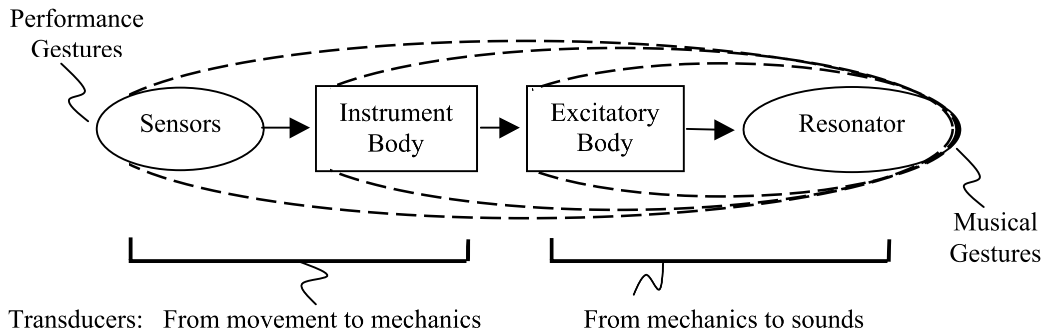



**Figure 1: Music Instrument Symmetry Model**

## 2.1 Local symmetry and global asymmetry in gesture throughput

The symmetry of the coupling between movement-sensing and sound-generating systems can be described as the association of performed movements to performed sounds. This association has been discussed as a mapping, i.e. the correspondence of sounds to movements as discussed for example by Hunt and Kirk [4] and Soto [2]. In further distinction, the symmetrical throughput model proposes a second-order correspondence, from one gesture transducer to another, a mapping of (a) movement gestures transduced as mechanical forces, with (b) mechanical forces transduced as sounds. Musical instruments are constructed to enable a limited range in a movement-sensing system to control a much larger range in a sound-generating system. This means the two gesture subsystems are radically asymmetrical; performers learn to master the asymmetry in order to stabilize the instrument into limited regions that are reliably symmetrical. Here "reliable" means a range of actions generates a predictable range of sounds. Instrument design and musical performance require mastery to create *local symmetry* within a range of gestures. This locally-symmetrical but globally-asymmetrical property in musical instruments creates performance conditions where a working definition of *gestural primitive throughput* can be applied. The definition will account for the role of gestural primitives in the actions of a performer to stabilize an instrument in a symmetrical region or to transition from one symmetrical region to another.

## 2.2 Symmetries as manifolds in musical instrument performance

Local symmetries between movement-sensing and sound-generation can be described as *manifolds*: a movement subspace and a sound control subspace that are continuously covariant. Learning to perform a musical instrument is a process of acquiring an ability to navigate these manifolds, a process in which the mode of navigation is defined by device-specific interaction. Performers execute transitions from one manifold to another with controlled degrees of discontinuity. The history of performance practice for a family of musical instruments includes a repertoire of gestures that reflect the characteristics of the manifolds native to that instrument family. Musical idioms common to an instrumental family reflect these characteristics. Performers learn idiomatic gestures as a repertoire of manifold stabilization and navigation techniques. These techniques are related to the musical idioms of an instrument's vocabulary. Gestural primitives provide a view of these idioms not as musical units rather as a performer's stance and movement orientation toward an instrument, undertaken to generate correspondences in sound.

# 3. A COMPONENT MODEL OF GESTURE THROUGHPUT FOR INSTRUMENT DESIGN

For the design of new musical instruments involving digital signal processing and movement sensors, a component model of gesture throughput includes an input control device, a manifold interface, a generative mechanism and a sound synthesis engine. Figure 2 illustrates these in a schematic model with a manifold mapping as an axis of symmetry. The manifold map is situated between two gesture transducing subsystems. In a traditional music instrument this mapping is a property of energy transmission through the instrument's body. In digital instrument design the manifold interface provides a dedicated mapping component to bring the first transducer into a symmetrical relationship with the second.

## 3.1 Gestural Primitives

Researchers have identified basic movement orientation as a structure contributing to gestures, for example the models of effort, after Laban, discussed in Camurri and Trocca [5]. Gestural primitives are fundamental human movements that relate the human subject to dynamic responses in an environment. With respect to musical instrument performance, a gestural primitive represents a chosen physical disposition to a movement sensor with intent to modify a dynamical process generating sound. The classification is based on the most fundamental human factors that are (1) the sense of movement, (2) the sense of weight distribution, and (3) the sense of organizing recurrent tasks. Concomitant gestural primitives are trajectory-based, force-based, and pattern-based primitives [6]. This classification is determined by the relationship between the performer's orientation and the mechanical or computational signal path where the gesture is transduced. The present task is to trace the relevance of gestural primitives along the signal path.

## 3.2 Gesture Transducers

In Figure 2 an input control device transduces movement as a function of two conditions. First the physical configuration of one or more sensors provides affordances for a performer's contact and movement. Second the movement at the sensor is encoded and quantized in range, resolution and in timing, then relayed along a signal path. In a computational system this encoding may involve pattern recognition or other data classification. Together the sensor configuration and the

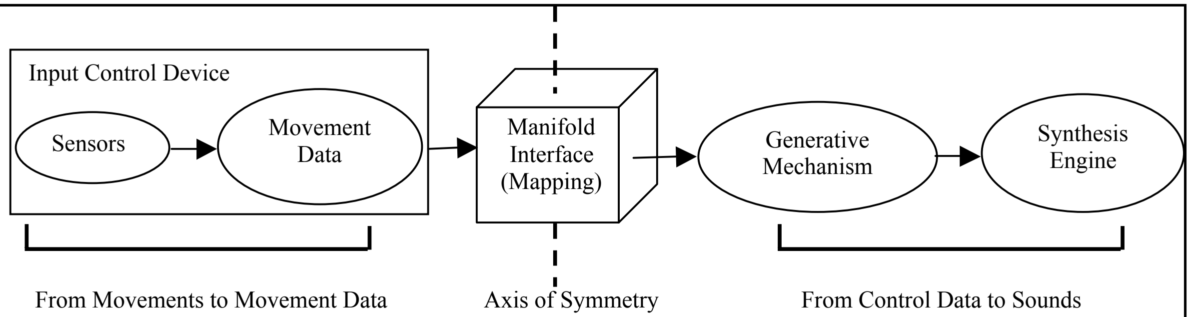



**Figure 2: Gestural Primitive Throughput component model**

encoding performed by the input control device determine the system affordances for gestural primitives.

In Figure 2 a generative mechanism is situated to receive data signals from an input control device and generate control signals for synthesis engines. The term "generative mechanism" implies that the control signals transmitted to the sound synthesis engine will be generated to exhibit coherence properties of some kind [6]. Coherence in control signals produces signatures in the sound in the form of covariance of multiple audible properties. For example, a generative mechanism might ensure that brightness in the sound co-varies with loudness, register and speed of onset (so-called "attack transients"). Coherence may be obtained through look-up tables, physically-based models or other methods for imparting structure.

Gestural primitives can affect the characteristics of control data transmitted to a generative mechanism, in turn affecting the coherence of a sound. These affects may be thought of as auditory signatures of the gestural primitive. A music instrument's capacity to transmit movement characteristics into auditory signatures is supported by the mapping that defines the symmetry in the instrument model.

## 3.3 Manifold interface properties

There are many methods for establishing the *m:n* mapping required to convert movement data into sound control data. In the method presented here, a manifold interface was previously developed for real-time control of a chaotic circuit as a musical tone generator [7]. Chaotic circuit oscillations present similarities to stability and instability of oscillation in traditional music instruments [8]. The manifold paradigm proved useful in identifying stable regions and generating control signals that required covariance of a large number of parameters (13 in the initial case). The algorithm supporting this interface provides an efficient way to establish *m*:*n* mappings that are continuous and differentiable within bounded parameter subregions of the total control space [9]. The interface supports the dual-transducer model in Figure 2 by establishing a "Window Space" – an abstract spatial model of the movement data from the input control device (in *m* dimensions). The interface is able to map continuously the Window Space into a "Phase Space" – an abstract spatial model of the sound synthesis control parameters (in *n* dimensions).

Using this interface a manifold subregion may be varied continuously within the global *m:n* asymmetry. This permits smooth transformation of the relationship between movement data (transducer 1) and multiparamter sound control (transducer 2). This smooth transformation capacity is important for real-time fine-tuning during a performance, and important also for tuning the sound-generating system when changing from one form of input control device to another, each supporting different gestural primitives

## 4. INSTRUMENT MODEL TUNING WITH GESTURAL PRIMITIVES

The capacity of an instrument model to produce musical gestures and with them musical idioms can be traced from the sound generator back through the physical controller to the performer's stance and disposition. Gestural primitives are generated with respect to this stance and contribute to the auditory signatures that are possible within the system. A system can be tuned for optimal symmetry, resulting in auditory signatures that reflect adequately both the properties of the sound generator and the details of a performer's movements. In practical application this means that a performer will be able to observe a variety of sound transformations corresponding to his or her disposition to the instrument, including but not limited to the individual actions imparted to the instrument. Sequences and relative changes in forces and nuances of actions are determined by the gestural primitives involved.

The task of musical performance includes real-time complex path planning in a control space, where the states of oscillation and the navigation strategies are informed and assisted by auditory feedback. Because music instruments present an entrainment scenario to a performer, the auditory feedback can be understood as an index of the performer's movement orientation with respect to a manifold space. Symmetry is important for instrument optimization because the gestural primitives provide a performer's primary sense for regulating upcoming actions and navigating the stability of the oscillatory system. A well-tuned manifold improves the transfer of the dynamics of gestural primitives into auditory signatures. For the performer these signatures become the idiomatic expression of the stability or transience of an instrument's oscillations with respect to performance actions.

The purpose is to develop a tuning process for new instrument designs. A tuning process can provide an overview of possible system states and a structure for selecting mappings. The intent is to improve upon the practice of generating *ad hoc* control movements and tweaking various parameter settings until "it sounds better." The representation of control space as a manifold lends a useful structure for identifying the subset of oscillatory states that are contiguous under a performer's sequence of actions. This representation can be used both to fine-tune the settings of an instrument and to assist performers in developing an understanding of the idioms of the instrument.

## 4.1 Proposed work

New research focuses on developing a system to determine the measurability of gestural primitives through auditory signatures. As stated above gestural primitives are an impetus; they are detectable in the transfer of movements in a signal path; they are not units or segments of gestures. In [10] Marin introduces a signal analysis approach to study covariance of movement and sound in the case of a conductor's movements. Further development of a measurement approach to evaluating movements with respect to sounds is needed, and is an appropriate topic for a PhD thesis in the field.

A real-time implementation of the component throughput model is underway in an interactive performance system and will be available to study the signatures of gestural primitives. The system combines previous work in manifold representations and mappings and in gesture-based performance. A computational implementation can be used to investigate the design of alternative high-dimensional maps situated between movement sensors and musical signal generators. Planned research includes developing a specification for a design interface at the axis of symmetry in instrument models. This interface would enable the optimization of stable manifolds using the signatures of various gestural primitives as a tuning method.

## 5. REFERENCES


[1] Cadoz, C. "Instrumental Gesture and Musical Composition." In *Proceedings of the 1988 International Computer Music Conference.* San Francisco, ICMA, pp. 1-12.

[2] Goto, S. "Virtual Musical Instruments: Technological Aspects and Interactive Performance Issues." In *Trends in Gestural Control of Music.* Battier and Wanderley, eds., IRCAM, Centre George Pompidou, Paris. 2000.

[3] Cadoz, C. and Wanderley, M. "Gesture – Music." In Battier and Wanderly, Op. Cit.

[4] Hunt, A. and Kirk, R. "Mapping Strategies for Musical Performance." In Battier and Wanderley, Op. Cit.

[5] Camurri, A., and Trocca, R. "Movement and Gesture in Intelligent Interactive Music Systems." In Battier and Wanderley, Op. Cit.

[6] Choi, I. "Gestural Primitives and the context for computational processing in an interactive performance system." Battier and Wanderley Op.Cit.

[7] Choi, I. "Interactive exploration of a chaotic oscillator for generating musical signals in real-time concert performance." *Journal of the Franklin Institute*, vol. 331B, no. 6, pp. 785-818

[8] Rodet, X. "Stability/instability of periodic solutions and chaos in physical models of musical instruments". *Proc. ICMC*, Aarhus, Denmark, pp. 386-393, 1994.

[9] Choi, I. "A Manifold Interface for Kinesthetic Notation in High-Dimensional Systems." In Battier and Wanderly, Op. Cit

[10] Marrin, T. "Searching for Meaning in Gestural Data." In Battier and Wanderley, Op. Cit.

# Demos

# The Hyper-Flute

Cléo Palacio-Quintin
Flutist-improviser-composer
3501 De Lorimier Avenue
Montréal, Québec, Canada H2K 3X5
(514) 525-3649
cleopq@sympatico.ca

## ABSTRACT

The Hyper-Flute is a standard Boehm flute (the model used is a Powell 2100, made in Boston) extended via electronic sensors that link it to a computer, enabling control of digital sound processing parameters while performing. The instrument's electronic extensions are described in some detail, and performance applications are briefly discussed.

## Keywords

Digital sound processing, flute, hyper-instrument, interactive music, live electronics, performance, sensors.

## 1. FROM INSTRUMENTAL GESTURE … … TO DIGITAL PROCESSING

When I decided to play flute with live electronics, the key issue for me, as a performer, was preserving the intimate relationship between my body, my instrument and the sound it produces. I wanted to keep intact the acoustic richness of the flute, and my way of playing it. The computer had to become a virtual extension of the acoustic instrument.

The richness of physical control required for performance with traditional acoustic instruments takes time to learn. I spent more than 15 years honing my instrumental skills on the flute. While playing an acoustic instrument, all performers receive mechanical feedback cues via a variety of physiological and perceptual mechanisms. These haptic sensations include tactile sensation (touch sensitivity of the skin) and proprioceptive or kinaesthetic perception (awareness of one's body state, including position, velocity and forces supplied by the muscles). Of course, aural feedback is also very important, but the tactile sensation of how one is playing tends to prevail over the sonic outcome.

In extending my flute's tonal field with digital sound processing, I wanted to retain the same subtle control over the sound. It was obvious that I would be better off using my already refined instrumental skills to control the sound processing parameters along with the acoustic flute. The key to achieving this was to use kinetic sensors to capture performance information on the flute and then send these data to the computer. Sensors can convert physical energy into electricity; they make the link between the human world and the machine world.

The next section describes in detail the types of sensors used on the Hyper-Flute, how they react, and exactly where they are installed on the instrument. The physical gestures made while playing the flute have direct consequences on all sensor information sent to the computer. While the mapping of these data into meaningful controls for sound processing is one of the most important issues in working with such an interface, this short article focuses more on the physical description of the instrument.

## 2. SENSORS

Though there is not much space available to add such hardware to a flute because of its complex mechanism of small keys, it was possible to install several sensors in specific, strategic places:

- magnetic field sensors, which detect the position of the G# key and low C# key (controlled by the two little fingers);
- an ultrasound transducer, which monitors the distance between the flute and the computer;
- mercury tilt switches, activated by the tilting and rotation of the flute;
- pressure sensors under the main points of contact between my hands and the flute (i.e. the left hand and both thumbs);
- a light sensor, which reacts to ambient light on the flute; and
- button switches (discrete values: on/off), which can be reached with the thumbs while playing).

These analog sensors send continuous voltage variation data to a *Microlab* interface, which converts them into MIDI (Musical Instrument Digital Interface) data. These data are then redirected to the computer via a standard MIDI port.

The Microlab is an electronic interface that analyzes the voltage variations from various analog sensors (between 0 and 5 volts) and converts this information into standard MIDI data, which can be sent to a computer, synthesizer, sampler or any other MIDI-compatible device. The interface was originally designed and developed by J. Scherpenisse and A.J. van den Broek, working at the Department of Sonology at the Royal Conservatory in The Hague, Netherlands.

Proprioceptive sensors, describing movements or position, continually send data as MIDI Continuous Control Messages.

Two *magnetic field sensors* transmit the exact positions of two keys (G# and low C#, both controlled by the little fingers). The very short key action distance is precisely measured in 95 steps. I can play with the keys to generate different curves for the output, with quite accurate control. This of course affects the acoustic properties of the flute.

The *ultrasound transducer* is used to measure the distance between the flute and the computer. An ultrasonic signal is sent from the computer and received on the flute. By calculating the delay time, the Microlab provides two different scalings of the flute-to-computer distance.

Some movements also provide discrete values. Two *mercury tilt switches* are triggered by the movement of the instrument. Tilting the flute (moving the footjoint up) activates a Note Number message (on-off), and rotating the flute (turning the headjoint outward) activates another Note Number message.

*Pressure sensors* are considered as isometric, because there is no movement involved, only muscle tension. Three of them are installed on my flute at the main contact points; each sends Continuous Control Messages. A larger one is installed under my left hand, which holds the flute. There is constant contact and pressure variation as I play. A smaller pressure sensor is found at the B key, under the left thumb. While playing, this key moves often, and is sometimes released completely. The third sensor is under the right thumb, which also supports the flute. There is constant variation depending on which fingerings are played and on the instrument's balance. For these three sensors, maximum values are reachable only with extreme pressure, which does not occur in normal playing, but can be used expressively.

A *light sensor* also sends Continuous Control Messages. This photoresistor, which detects variations in ambient light, is positioned at the headjoint, and is designed to be used in conjunction with stage lighting effects.

Other controllers used on the Hyper-Flute are small *button switches,* which send discrete values (Note Number on/off). Two of them (blue) are placed close to the headjoint and are not easily reachable while playing. I mostly use them to change settings between sections or at the beginning of a piece. Four others are placed close to the thumbs, and can be reached while playing.

## 3. MAPPING & PERFORMANCE

Using the Max-MSP programming environment, different programs, or "patches," are developed and integrated into a complex software interface, which performs the flute's sound processing in real time. This software is entirely controlled by the Hyper-Flute. For each programmed patch, all the MIDI data can be processed and used to control different sound processing parameters. The mapping of the MIDI daya to different parameters can be modified before each performance, or even during a performance. This mapping is a crucial step in the interface between instrumental gesture and digital sound processing.

As explained earlier, each sensor occupies a specific position on the instrument. The way the flute is played and held creates multiple interactive relationships between some of the sensors. For example, diminishing the pressure on one or both thumb sensors immediately increases the pressure under the left hand, and pushing one of the button switches causes a corresponding thumb pressure sensor to lift. This interaction between the various sensors needs to be considered while programming the mapping of the data to the sound processing parameters.

Of course, different sound processing patches require different ways of controlling them. The mapping must be adapted to each specific situation, and a lot of fine-tuning is necessary. Since my sound processing software is in continual development, no definite mapping scheme is in use yet. I am constantly experimenting with different combinations of direct, convergent and divergent mapping, some being more suitable than others for controlling specific sound processing patches. Comprehensive analyses of the data generated by the sensors while playing the instrument would be necessary to find more precise relationships and develop a very good, multiparametric interface (see Hunt and Kirk, 2000, for more details on mapping strategies.)

Musical applications of the Hyper-Flute are infinite. The immediate link between the physicality of playing the flute and the data sent to the computer makes possible a variety of interactions between performer and computer. Besides processing the acoustics of the flute in real time, the patches can also be used to control other types of electronic structures—for example, to trigger sound files or independent sound synthesis algorithms. The design of the patches thus becomes part of the compositional process. The whole concept is very different when a composed piece is being played, as opposed to improvisational performance. Musical examples will be performed during the demonstration of the Hyper-Flute.

## 4. ACKNOWLEDGMENTS

I extend especial thanks to Bert Bongers and Lex van den Broek, who built parts of my instrument and patiently taught me how to solder tiny cables, sensors, resistors, switches, and so on; to Jonathan Impett, whose Meta-Trumpet and musicianship have been and continue to be a huge source of inspiration; to Paul Berg, for making me discover and plunge into the world of computer music; and to Anne LaBerge, for her fantastic flute tips, contagious energy and musicality.

## 5. REFERENCES

[1] Bongers, B. Physical Interfaces in the Electronic Arts. In *Trends in Gestural Control of Music* [CD-ROM], M.M. Wanderley and M. Battier, eds., Paris: IRCAM-Centre Pompidou, 2000.

[2] Bongers, B. The Use of Active Tactile and Force Feedback in Timbre Controlling Electronic Instruments. *Proceedings of the 1994 International Computer Music Conference,* Arhus, Denmark, 1994.

[3] Hunt, A. and Kirk, R. Mapping Strategies for Musical Performance. In *Trends in Gestural Control of Music,* op. cit.

[4] Impett, J. A Meta-Trumpet(er). *Proceedings of the 1994 International Computer Music Conference,* op. cit.

[5] Machover, T. and Chung, J. Hyperinstruments: Musically Intelligent and Interactive Performance and Creativity systems. *Proceedings of the 1989 International Computer Music Conference,* San Francisco, CA, 1989.

# SensorBox: Practical Audio Interface for Gestural Performance

Jesse Allison
University of Missouri – Kansas City
Impact Center
4949 Cherry
Kansas City, MO 64110 USA
jta@allisonic.com

Timothy Place
University of Missouri – Kansas City
Impact Center
4949 Cherry
Kansas City, MO 64110 USA
tap@sp-intermedia.com

## ABSTRACT
SensorBox is a low cost, low latency, high-resolution interface for obtaining gestural data from sensors for use in realtime with a computer-based interactive system. We discuss its implementation, benefits, current limitations, and compare it with several popular interfaces for gestural data acquisition.

## Keywords
Sensors, gestural acquisition, audio interface, interactive music, SensorBox.

## 1. INTRODUCTION
Practitioners of interactive music frequently require sensing devices to obtain gestural control of various signal-processing parameters. While much attention is given to new and novel sensing devices, and to the interactive system itself, the interface that connects the two is often overlooked. This "middle man" which negotiates between the analog sensors and the digital computer is frequently a source of bottlenecks, latency variations, and system instability, not to mention considerable expense. These attributes combine to make such systems impractical for many artists [1].

SensorBox is an interface developed in an attempt to resolve these difficulties. Our initial experiences with the SensorBox have shown it to perform admirably when compared to most commercially available systems.

## 2. DEVELOPMENT
The SensorBox is the third generation in a series of solutions utilizing standard audio hardware to digitize sensor data. The initial solution, the TeaBox (so named because it was housed in a tea box), converted continuous voltages from two sensors into square waves, using a 555 timer, and transmitted them to the computer over two audio lines [Figure 1]. The data was represented as the frequency of the square waves. The computer decoded the sensor information by counting the zero-crossings of each signal. This solution worked well, however it uses bandwidth very inefficiently, and could quickly fill all available audio inputs on a computer.

The TeaBox2 improved on this design by replacing the square wave oscillators with sine wave oscillators. Still built in a tea box housing, the TeaBox2 collects data from three sensor sources and mixes the sine wave signals they generate onto a single audio line to send to the computer. The computer then separates the signals using band-pass filters and analyzes each signal individually. The data here is represented by the amplitude envelope of the sine waves, which is converted into a control signal in the software.


**Figure 1. The TeaBox sensor-audio interface.**

## 3. CONSTRUCTION
### 3.1 Hardware
The SensorBox is a single rack-space unit that connects sensors to a computer through any available audio input(s) on a computer. The front panel hosts 2 XLR inputs and 8 Neutrik combo (XLR or 1/4" TRS) inputs, while on the rear, two audio outputs feed the computer.

The premise of the SensorBox is that frequencies above 18,000 Hz in the audio input will not be needed by the interactive system. The SensorBox filters the audio input through a low-pass filter with a cutoff frequency of about 17 KHz. Meanwhile, the connected sensors drive oscillators in the range between 18 and 20 KHz, which are mixed back into the audio signal that goes to the computer. Software on the computer removes these high frequencies from the input stream and performs a computationally inexpensive analysis to acquire the data from the signal.

### 3.2 Software
Due to the simple methodology used to transmit the data, a variety of software may be used to interpret the acquired sensor data. This includes MaxMSP [Figure 2], PD, Jmax, and SuperCollider. Additionally we have created a VST/MAS/RTAS plugin for the Mac which will automate other plugins with the sensor data in realtime [Figure 3].

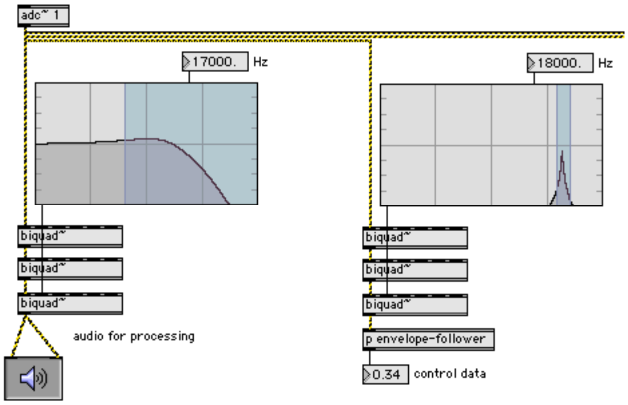



**Figure 2. A portion of a MaxMSP patch for separating and analyzing the frequency regions.**

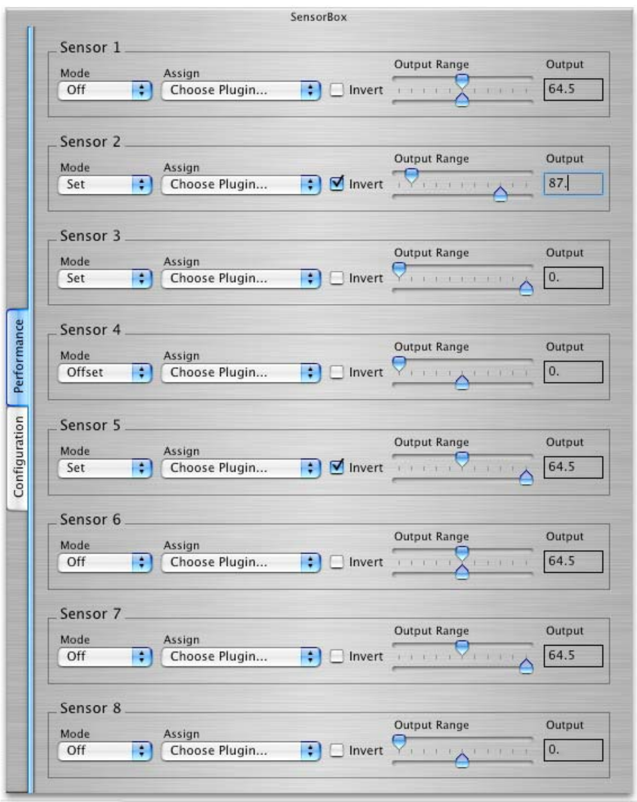



**Figure 3. The SensorBox mapping plugin for VST/MAS/RTAS.**

## 4. BENEFITS

### 4.1 Connectivity

One of the significant benefits of this approach is that it connects to the computer through the audio inputs. It is beneficial because most musicians already have an audio interface or audio input device of some sort, thereby eliminating platform dependencies and the need for custom drivers or extensions.

It also avoids the use of USB either directly or through a serial-to-USB converter. The authors have experienced reliability problems with USB, and we are happy to avoid it altogether by using these techniques. By using the audio stream we also avoid the use of MIDI.

### 4.2 Latency

One problem with realtime sensor acquisition systems is latency, and latency jitter, in the data from the sensor interface. By using the audio stream we are able to eliminate the jitter present in both MIDI-based systems [2] and some serial systems because the signals are digitized at audio sampling rates. This also reduces the latency vastly. Additionally, because all of the data is carried together on the same signal, and digitized by a common system, the gestural data is very tightly synchronized with the audio data entering the system, making overall system design a much less arduous task [3].

### 4.3 Resolution

While MIDI-based solutions typically have 7-bit (occasionally 14-bit) resolution, the SensorBox is able to leverage the dynamic range of the audio interface used on the computer (typically 16-24 bits). While we have not tested that the data the sensors produce can fully use this resolution, at least we can be assured that the interface is not a troublesome bottleneck for our data as is frequently the case with MIDI. Further empirical testing of this is currently in progress.

### 4.4 Cost

Most commercially available systems are expensive. The popular Icube system, for example, costs nearly $650 – not including the MIDI interface you need, nor any sensors [4]. The new Le Toaster for The Kitchen sells for $1200. Even do-it-yourself methods can be expensive, such as using the Basic Stamp. Systems like the Basic Stamp also require a significant amount of time and experience to get running smoothly. The Board of Education Kit for the Basic Stamp costs over $100, and doesn't include either the serial-to-USB adapter most Macintosh users would need, or a MIDI interface [5].

The SensorBox can be built for under $100 in parts, although factoring soldering time in will raise the price a bit. Variations, such as the TeaBox, can be built for as little as $5 or less.

## 5. FUTURE DIRECTIONS

As the SensorBox continues to develop, we are able to fine-tune its construction and improve its performance. Areas of particular interest are in making it a more easily scalable system, making the sensor acquisition wireless with the appropriate technology inside the box, and powering the box with phantom power from a microphone preamp or mixing board.

## 6. ACKNOWLEDGMENTS

Our special thanks to Dr. Paul Rudy who has encouraged this work and provided essential feedback.

## 7. REFERENCES

[1] Allison, Jesse and Place, Timothy. Practical Interfaces for Gestural Sensing Devices. Presented at the 4th annual Electronic Music Midwest Festival, Romeoville, Ill. December 7, 2002.

[2] Loy, Gareth. Musicians Make A Standard: The MIDI Phenomenon. *Computer Music Journal* Vol. 9 no. 4 (Fall 1985). Cambridge, MA: The MIT Press.

[3] Wessel, David, and Wright, Matt. Problems and Prospects for Intimate Musical Control of Computers. *Computer Music Journal* Vol. 26 no.3. Ethics and computer use. Commun. ACM 38, 12 (entire issue).

[4] Infusion Systems http://www.infusionsystems.com/

[5] Parallax Inc. – Basic Stamp MicroControllers http://www.parallax.com/

# Multi-Conductor: An Onscreen Polymetrical Conducting and Notation Display System

Kevin C. Baird
University at Buffalo
Baird Hall
Amherst, NY 14221
01-716-564-1172
kcbaird@world.oberlin.edu

## ABSTRACT
This software tool, developed in Max/MSP, presents performers with image files consisting of traditional notation as well as conducting in the form of video playback. The impetus for this work was the desire to allow the musical material for each performer of a given piece to differ with regard to content and tempo.

## Keywords
Open form, notation, polymeter, polytempi, Max/MSP.

## 1. INTRODUCTION
While the system could easily be used for closed-form polymetrical pieces, it is intended for the performance of compositions with open forms, in which some or all decisions about the temporal relationships of pre-existing material are made in real-time during the performance. When implemented, each performer would have a client CPU running either Max/MSP or the free runtime version, with notation and video conducting presented on the display. Each client is designed to accept MIDI data from either a master CPU or a human, which determines tempo, start time, and notated content. Prior to the actual downbeat, each performer is given a visual warning (shown as a red "X"), followed by a two beat pickup at the appropriate tempo.

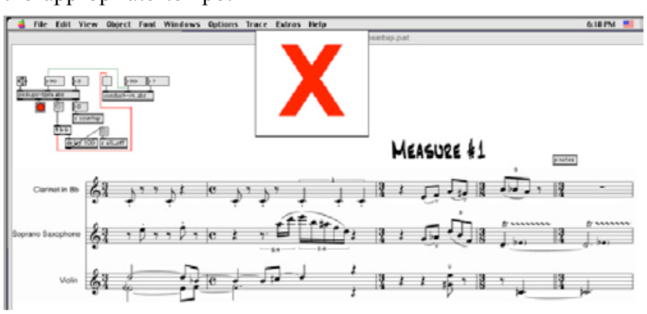



**Figure 1: The system alerts the performer that pickup beats are imminent**

The patch uses two main abstractions with which the user has any interaction: pickups-bpm.abs and conduct-[inst].abs, where [inst] is replaced by vn, cl, or sax. These suffices indicate the instruments arbitrarily used by the musical example: violin, clarinet, and soprano saxophone - this allows for easy changes of instrumentation.

When used, the patch has pre-defined musical content stored as external text files, which store metrical data for the examples of notated content called "iterations." The conduct-[inst].abs abstraction takes arguments of tempo and an integer identifying the "iteration" to be used, while the pickups-bpm.abs abstraction takes arguments of tempo and metrical numerator.

## 2. PICKUPS
In order to ensure both that the multiple players could start together when necessary and also have appropriate pickup beat indications, a delay is required between the signal for a simultaneous start and the downbeat. An arbitrary time of 2000 ms was chosen, to allow for two pickup beats at a tempo of 60 bpm. This "lead time" is always present, but individual users could easily alter this value as suits their own needs. When a signal for a start time is received, the player's screen shows the red "X" indicator, meaning "get ready to play" (see Figure 1). Parallel client-side versions of the pickups-bpm.abs abstraction then calculate beat timings and beat number for the appropriate player, whose screen then shows numerals (as appropriate for the current meter) indicating the last two beats of the pickup measure (see Figure 2).


**Figure 2: The pickup corresponding to beat #2 within whatever meter is currently in use (3/4, in this case)**

## 3. DOWNBEATS
At the downbeat of measure 1, the X and numeral indicator box shows video frames of a conductor. Playback speed is controlled by the tempo argument to the conduct-[inst].abs abstraction. The system uses a combination of beat patterns (2, 3, 4, 5, and 7) and externally-loaded text files to allow for metrical and tempo changes from measure to measure within a given phrase (e.g. the change from 3/4 to 3/8 back to 3/4 in Figure 2). It also displays the current measure within each phrase using a numerical indicator as shown in Figure 3.

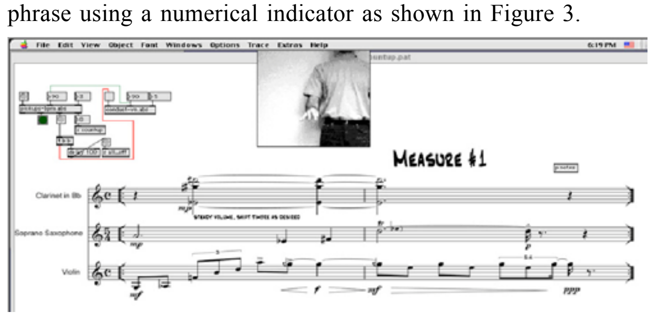



**Figure 3: The downbeat of mm1**

Video playback continues, looping until new information indicating iteration is received. New tempo information takes effect at the next downbeat of measure 1.


**Figure 4: Mid-beat in mm5**

## 4. OVERHEAD/PERFORMER FEEDBACK

Rather than true video files (.mov, .avi, etc.), the patch uses the PICS object, which stores successive stills in the PICT image format. This allows for a simple dividing operation to determine the bang rate from a metro object used to step through the PICS array. Each image has been downsampled to 3-bit grayscale, which seems to be the smallest file size which still maintains enough image quality for reasonable visibility. The patch has a memory overhead of approximately 76 Megabytes to load the PICS arrays into RAM. As systems become more powerful and especially as RAM becomes cheaper, this requirement does not seem terribly onerous.

The placement and type of notational information was explicitly chosen to accommodate performers' existing training and expectations. Early feedback from performers suggests that players should have minimal difficulty in adjusting to the conducting and notation method used in the patch.

## 5. ACKNOWLEDGMENTS

Thanks to Erik Oña of the University of Birmingham, England, Cort Lippe, Director of the LeJaren Hiller Computer Music Studios at the University at Buffalo, and Susan Fancher of the Amherst Saxophone Quartet for their helpful feedback and suggestions.

## 6. ACQUISITION

The newest version of this software is available at http://www.kevinbaird.net/Multi-Conductor/index.html.

## 7. REFERENCES

[1] Brown, E. Available Forms I, Associated Music Publishers, Inc., New York NY, 1962.

[2] Lawter, J., and Moon, B. Score Following in Open Form Compositions. in Proceedings of ICMC '98 (Ann Arbor MI, Oct 1998), International Computer Music Association, 21-24.

[3] Pennycook, B., and Johnstone, E. MIDI Time Clip: A Remote Messaging Device for Interactive Performance. in Proceedings of ICMC '92 (San Jose CA, 1992), International Computer Music Association, 459-460.

[4] Rowe, R. Interactive Performance Systems. MIT Press, Cambridge MA, 1993.

# Dsp.rack: Laptop-based Modular, Programmable Digital Signal Processing and Mixing for Live Performance

William Kleinsasser
Towson University
Department of Music
8000 York Road
Baltimore, MD 21252 USA
wkleinsasser@towson.edu

## ABSTRACT

This document describes modular software supporting live signal processing and sound file playback within the Max/MSP environment. Dsp.rack integrates signal processing, memory buffer recording, and pre-recorded multi-channel file playback using an interconnected, programmable signal flow matrix, and an eight-channel i/o format.

## KEYWORDS

Digital signal processing, Max/MSP, computer music performance, matrix routing, live performance processing.

## 1. INTRODUCTION

Dsp.rack is a suite of Max/MSP modules that run on a Macintosh Powerbook, iBook, or desktop computer with a G3 500 mHz or faster CPU. Dsp.rack uses the familiar paradigm of combined mixer, patch bay, and signal processors for integrating electronic music with live performance. Dsp.rack was developed to take advantage of the familiarity of this paradigm and the decades of performance practice related to it. Building on the flexibility offered by software-based systems, Dsp.rack integrates the functions of programmable mixing, routing, and audio processing along with the ability to play overlaid, pre-recorded sound files. Dsp.rack was designed to offer a familiar, flexible, and open-ended entry point to composers, performers, students, and teachers.

## 2. THE DESIGN

Dsp.rack is available in two versions which offer beginning and more advanced environments for live performance. Dsp.rack version 1 uses a menu-driven crossbar method for routing signals. This version offers a flexible and simple approach to integrating signal input, routing, processing, mixing, and output. Version 2 uses the matrix~ object for routing that supports programmable, complex, signal flow combinations. Having been developed in Max/MSP, Dsp.rack also benefits from the open sharing of resources that comes with that environment.

A basic set of processing modules is included with the distribution of Dsp.rack and a mini-tutorial on integrating

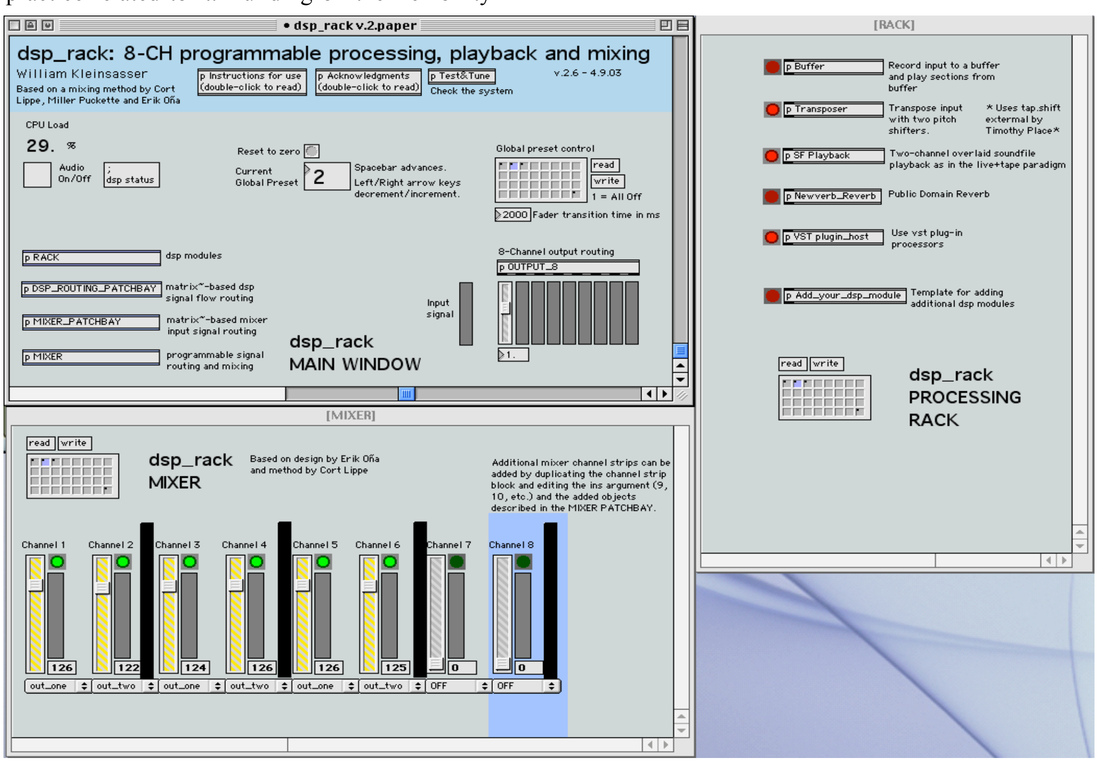



**Figure 1: dsp.rack screens**

additional user-designed modules is provided. The mixer and patch bay are extendable and limited only by screen saturation and processing speed of the computer.

Running on a Powerbook with an eight-channel i/o converter like the RME Hammerfall or MUTO 828, Dsp.rack can support eight independent input and output channels for processing. With other i/o hardware, like the MOTU 2408, it can support up to 24 channels. This makes Dsp.rack capable of instrumental and vocal ensemble processing with multi-channel output.

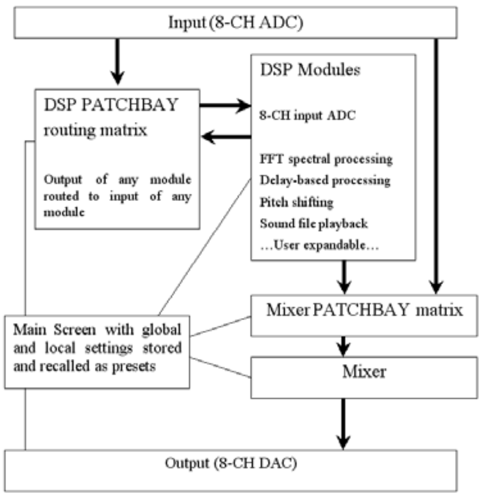



**Figure 2: dsp.rack block diagram**

## 3. INTEGRATED SIGNAL PROCESSING AND PRE-RECORDED SOUND FILE PLAYBACK

The integrated performer+tape paradigm that flourished after 1960 offers a model of musical expression that expands the capabilities of acoustic music through integration with electronic studio environment. Composers have produced a repertoire that presents acoustic performance in the context of technologically transformed music on tape but the synchronization issues involved in performer+tape music remain a concern in these works. Dsp.rack is designed to support live interactive signal processing as well as performer+tape repertoire.

This is done by offering the ability to present pre-recorded, overlapping sound files using a method for mixed overlaying that enables performance timing flexibility. The sound file player module loads and plays sound files using four independent multi-channel players. Sound files can either be routed directly out to the sound system or, using the flexible signal flow matrix, they can be routed to the inputs of the other processing modules. Dsp.rack can layer sound files with as many channels as the i/o supports depending on sufficient drive speed, i/o buffering, and CPU loading.

## 4. PERFORMANCE AND CPU LOAD

CPU load is directly related to the processing intensity and number of simultaneous modules used as well as the i/o vector sizes. Running several simultaneous dsp modules, an 8-channel mixer, and 8-channel i/o, Dsp.rack uses about 35% of the CPU on a 1G G4 Powerbook. The same setup uses about 75% of a 500 mHz G3 Powerbook. Dsp.rack uses the mute object for enabling and disabling each individual dsp processor which is useful for handling collections of processor-intensive modules.

Dsp.rack provides a path to familiar, personally expandable tools for integrating computer music with live performance and it is hoped that it will prove attractive to composers, performers, students, and those who teach others entering the field of live electro-acoustic music.

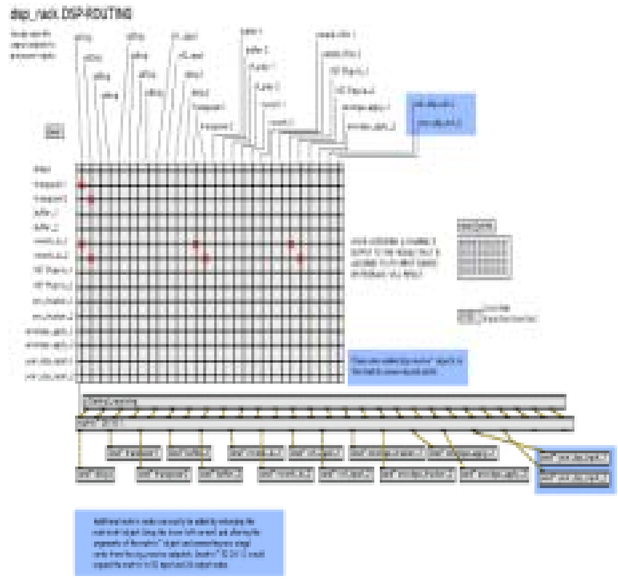


**Figure 3: Version 2 matrix-driven routing**

## 5. ACKNOWLEDGMENTS

The audio processing in Dsp.rack is based on standard-issue Max/MSP objects with the exception of the tap.shift pitch shifting object which is distributed with Dsp.rack by permission from its programmer, Timothy Place.

Dsp.rack owes to the following Max/MSP developers who have offered models and suggestions during development: Cort Lippe, Miller Puckette, and Erik Ona who developed models for crossbar mixing and routing methods using menu-driven send/receive signal flow. The approach of modular dsp functions in an integrated software environment relates to work by Cort Lippe (*compositions*) and Zack Settel (*multi effects processor, Jimmies*) Christopher Dobrian and Cort Lippe offered help on the buffer writing method and other audio handling. Daniel Koppelman provided the preset advancing method. The sound file playback and delay methods were developed in order to help, and deriving help from, my students Brian Comotto, Daniel Hope, Ljiljana Jovanovic, Scott Leake, and Nicholas Schoeb. Thanks to Miller Puckette and David Zicarelli for developing Max and Max/MSP and to the Max/MSP developers who share their solutions and ideas.

## 6. REFERENCES

[1] Dobrian, C. Programming New real-time DSP Possibilities with MSP, *Proceedings of the 2nd COST G-6 Workshop on Digital Audio Effects (DAFx99)*, NTNU, Trondheim, December, 1999

[2] Lippe, C. A Look at Performer/Machine Interaction Using Real-time Systems, *Proceedings of the International Computer Music Conference*, Hong Kong, 1996

[3] Lippe, C. A Composition for Clarinet and real-time Signal Processing: Using Max on the IRCAM Signal Processing Workstation, *Proceedings of the 10th Italian Colloquium on Computer Music*, Milan, 1993

[4] Lippe, C. Music for Piano and Computer: A Description, *Information Processing Society of Japan SIG Notes*, Volume 97, Number 122, 1997

[5] Place, T. tap.tools, Silicon Prairie Intermedia, http://www.sp-intermedia.com/downloads/index.html

[6] Puckette, M. New Public-Domain Realizations of Standard Pieces for Instruments and Live Electronics, *Proceedings, International Computer MusicConference*, 2001

[7] Rowe, R. *Interactive Music Systems: Machine Listening and Composing*, The MIT Press, Cambridge, 1993

[8] Settel, Zack, Jimmies, lecture/demonstration at the Max/MSP workshop at the University at Buffalo, June, 1999, Buffalo, NY

[9] Zicarelli, D. An Extensible Real-Time Signal Processing Environment for Max, *Proceedings of the International Computer Music Conference*, Ann Arbor, 1998[i]

---

[i] More about Dsp.rack can be found at: http://concert.towson.edu/WK/dsp.rack

# BASIS: A Genesis in Musical Interfaces

Mat Laibowitz
MIT Media Lab
Responsive Environments Group
1 Cambridge Center, 5FL
Cambridge, MA 02142
(617) 452-5614
mat@media.mit.edu

## ABSTRACT
This paper is a demo proposal for a new musical interface based on a DNA-like double-helix and concepts in character generation. It contains a description of the interface, motivations behind developing such an interface, various mappings of the interface to musical applications, and the requirements to demo the interface.

## Keywords
Performance, Design, Experimentation, DNA, Big Five.

## 1. INTRODUCTION
There has been much work in the use of signals and systems such as traffic patterns, viral mutations, and images to metaphorically compose and generate music [1,2]. However, the use of metaphor in this way to design new interfaces for realtime performance has been largely untapped. An interface designed in this fashion has the potential to give the musician feelings of creating more than music as they are now part of the metaphor. It also provides the audience with another level of interest and makes for a visually engaging performance.

## 2. DESIGN MOTIVATION
The original idea behind this interface was to design a system in which the musician was creating and modifying the music as if he/she was creating and modifying a personality in a story. Essentially, the musician would mix various personality traits and learned behaviors to design a character that would be musically mapped. The character can be continually modified and expanded. As the design process continued, the idea of incorporating genetic code into the character was adopted. This gave the musician the ability to create a DNA sequence for the character and modify this sequence in realtime. It was also a requirement for the interface to provide visual and tactile feedback to the user as well as be visually exciting for the audience. The interface consists of two physical parts, the DNA editor and the Personality editor.

## 3. DESIGN AND IMPLEMENTATION
### 3.1 DNA editor physical design
The DNA editor is built based on the Watson and Crick double-helix model of DNA[3]. The DNA editor is a 7 foot 6 inch tall vertical oriented double helix made out of aluminum, steel, and plastic. (see figure 1). The rungs on the double-helix represent the base nucleotide pairs. In DNA, a base pair can be one of four combinations, A-T, T-A, G-C, C-G. The order of these pairs makes up the genetic code. In the DNA editor, the rungs are made of a square tube attached to a rotary encoder allowing the user to select one of four positions. There are 80 rungs on the DNA editor grouped in 16 groups of 5. Each group of 5 is bounded by two plastic stationary rungs that can light one of three colors. The DNA editor can report to the mapping software the position of all the rungs and changes of these positions, as well as the speed with which the rung was rotated.

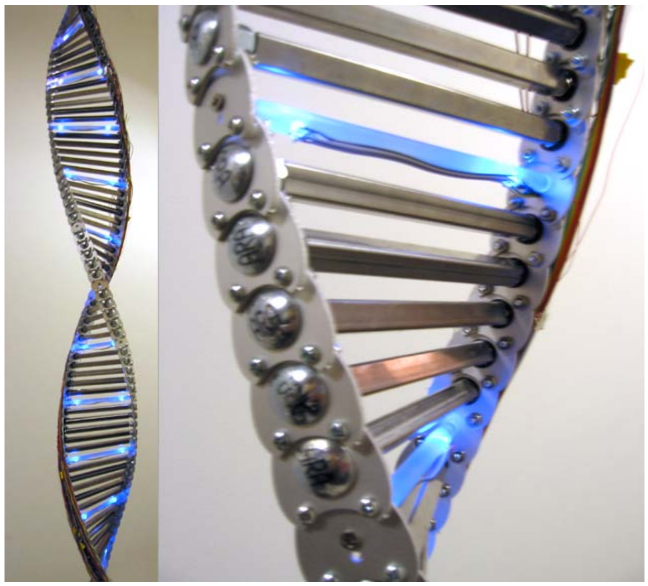


**Figure 1. DNA editor physical design, aluminum scales, rotating steel rungs, and lucite light rods.**

### 3.2 DNA editor mapping strategies
The DNA physical interface lends itself to many mapping strategies. Currently, 4 mapping strategies have been implemented and are available to demo. The first 3 strategies stick to the idea that DNA is a sequence of coded instructions and map this to a sequence of musical information. The 4$^{th}$ mapping strategy strays from the metaphor slightly and uses the harp-like nature of the DNA editor's physical implementation in its mapping.

#### *3.2.1 Mapping Strategy 1*
The first mapping strategy that can be shown uses the genetic code from the DNA as a sequence of notes as if it were an analog style sequencer. This is similar to past work done using DNA code to generate music [4], except that with this interface, it is modifiable in realtime as an instrument. The 16 groups light in tempo, each signifying a step in the sequence, with the rungs of each group setting the parameters for that step, pitch, glissando, and accent.

#### *3.2.2 Mapping Strategy 2*
Similar to strategy 1, but the DNA sequence becomes mapped to a 5-track drum machine. Each of the rungs in a particular step control the accent of a drum sound for that step.

### *3.2.3 Mapping Strategy 3*

The DNA sequence is mapped to a waveform of a particular instrument that is being played by traditional means. The DNA editor becomes a timbre control allowing the realtime dynamic effect of the sonic waveform of an instrument.

### *3.2.4 Mapping Strategy 4*

The double-helix is played like a harp, where each rung plays a note when spun. The speed at which you spin the rung controls the volume of the note.

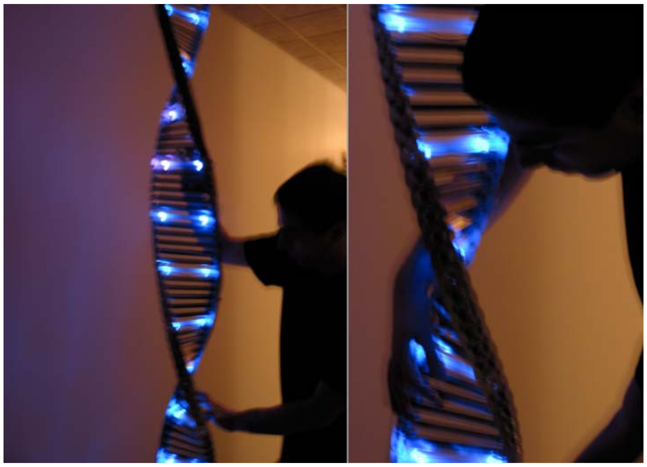


**Figure 2. The DNA editor being played**

## 3.3 Personality editor physical design

The personality editor is essentially a panel with five motorized faders, a single line lcd at each end of each fader, and a four line lcd with navigation buttons for control of the personalities (patches). Using this familiar interface a user is able to create and edit in real-time the personality of the character. A personality consists of seven sets of five parameters each. The sets are labeled as INBORN, GROWTH1 thru GROWTH5, and OVERALL GROWTH. Paging through the sets moves the faders to the stored values for that set and shows the names of the parameters on the lcds. On the INBORN page the parameters are based on the Big Five personality metric [5]. Each of the GROWTH# pages is used to design a modification to one of the Big Five traits. The five parameters are used to describe an event or a progression in the growth of the character that has an effect on one of the parameters in the Big Five metric. The OVERALL GROWTH page is used to design a modification on the entire character following the more specific growth elements.

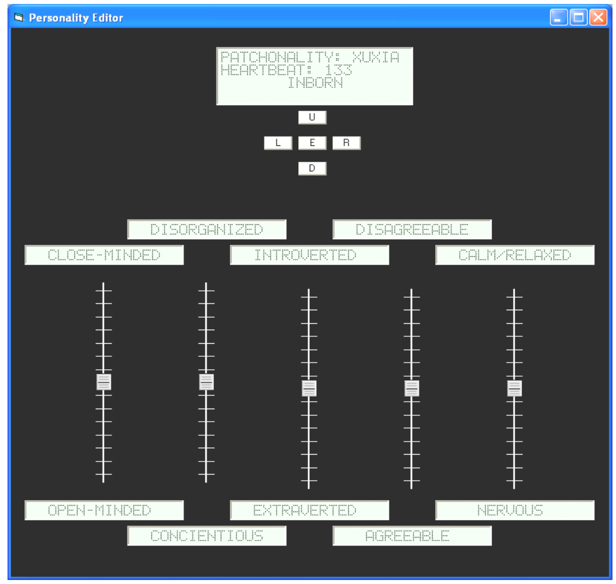



**Figure 3. Software Simulation of Personality editor**

## 3.4 Personality editor mapping strategy

The current mapping of the personality is to synthesis parameters. The inborn personality traits are mapped to sound generators such as oscillators, which are mixed in according to the mix of the Big Five personality traits. Growth modifiers are mapped to effects and filters on these generators. The synthesis parameters that the personality editor is mapped to give voice to the musical instructions that the DNA editor is mapped to. This completes the metaphor by combining the inborn genetic code with some personality traits leading to a complete character which is mapped to a complete musical section comprised of note information and timbre information all of which is "playable" as an instrument.

## 4. DEMO REQUIREMENTS

The main requirement for this demo is space. The DNA editor needs to be hung from the ceiling. It also requires room to walk around it. The demo also needs a table for the personality editor, computer, synth modules, mixing board, and speakers.

## 5. ACKNOWLEDGMENTS

Thanks to Joe Paradiso, Glorianna Davenport, and Steve Benton.

## 6. REFERENCES

[1] Cope, D. 2000 *The Algorithmic Composer*. Madison: A-R Editions.

[2] Biles, J.A. 2002 "GenJam: Evolution of a jazz improviser," *Creative Evolutionary Systems*, P.J. Bently and D. W. Corne, eds. San Francisco: Morgan Kaufmann.

[3] "A Primer in DNA Structure and Function." http://seqcore.brcf.med.umich.edu/doc/educ/dnapr/index.html

[4] Clark, M. A. "Protein Music." http://whozoo.org/mac/Music

[5] Goldberg, L. R. 1992 "An alternative 'Description of personality'': The Big-Five factor structure" *Journal of Personality and Social Psychology*

# Head-Tracking for Gestural and Continuous Control of Parameterized Audio Effects

David Merrill
MIT Media Lab
20 Ames St. E15-313
Cambridge, MA 02139-4307
dmerrill@media.mit.edu

## ABSTRACT
This paper describes a system which uses the output from head-tracking and gesture recognition software to drive a parameterized guitar effects synthesizer in real-time.

## Keywords
Head-tracking, gestural control, continuous control, parameterized effects processor.

## 1. INTRODUCTION
Researchers and performers working with gestural control of music have a history of training digital video cameras on themselves and inventing interesting mappings for the output. A notable early example is David Rokeby's Very Nervous System [5]. This paper presents a working system that uses a modified real-time head-tracker to drive a parameterized guitar effects processor. A conceptual model for a similar system was proposed by Lyons [7], and he built a system that used a visual mouth-tracker to drive parameterized audio effects [8].

## 2. SYSTEM OVERVIEW
The system consists of a modified real-time head-tracker which communicates via a TCP/IP socket connection to a custom server program. The server is responsible for managing the mapping of sensed gesture onto appropriate control messages, which it sends to a guitar effects processor via MIDI messages.

The FaceSense program [1] runs under Linux, and uses an IR-sensitive camera based on the BlueEyes camera from IBM [4]. Pupil positions and sizes are tracked using a difference images, and eyes/eyebrows are tracked using templates. At a higher-level, detection of head nods, head shakes, and eye blinks is also implemented. The system runs at 29-30 frames-per-second on our IBM Netvista 1.5GHZ machine.

The mapping server is written in Java, and is responsible for handling incoming data from the FaceSense program and managing the state of the guitar effects processor. Modules implemented include running-average filters to smooth the incoming data (implemented efficiently with a ring buffer), an amplifier-selector-manager to handle the state machine used in the gestural interaction, a midi device manager to provide easy midi message transmission, and a general-purpose sensor-value-to-midi-message mapping class.

The Line6 Pod 2.0 [6] proved to be a useful guitar effects processor for this project, since all settings are externally-controllable via MIDI messages (see figure 1).

**Figure 1: The author using the system. The IR camera is just beneath the monitor, and the effects processor is on the left.**

## 3. MOTIVATIONS AND MAPPING
The dialogue between humans and machines is fundamentally social in nature [2], and it has been argued that the man-machine interface can be improved by leveraging human expectations of natural human social cues when designing technologies that interact with people. Such a cognitive "scaffolding" can engage the user's existing behaviors and expectations about interaction to enhance interaction with a computer system [3]. This work explores ways in which a camera-to-audio-mapping interface can respond to the performer socially by reacting in the following ways:

Reaction to personal space: People are acutely aware of how close the face of their conversational partner is to their own, and an excessively close partner can cause anxiety or excitement. This system monitors maps distance between the performer's head and the camera onto a continuously-varying parameter (wah and volume were tried), tracking the continuously-varying level of comfort associated with personal space intrusions.

Reaction to face orientation: Cognitive psychology has shown that faces and other shapes become less recognizable as they are rotated off-axes. This system maps the tilt of the performer's face onto a continuously-varying parameter (volume and distortion level were tried).

Gestural communication: Humans frequently communicate semantic content by gesture, especially musicians in an environment too loud for spoken language. Musicians often cue each other with a head nod, signaling a musical change. This system engages the performer in a gestural dialogue to switch between amplifier presets. The head nod/shake gestures which the performer uses to drive the interaction borrow the scaffolding of everyday human-human yes/no interactions (see figure 2).

Finally, the interface helps solve the "laptop musician problem" in which the performer-computer interaction inhibits performer-audience interaction. Even if the performer chooses to look at the computer screen during the interaction, the computer is also "looking back", and the camera's eye view of the performer can be projected for the audience to enjoy as well. Furthermore, a shoulder-mounted display would allow the performer to walk the stage and continue to interact directly with the audience.

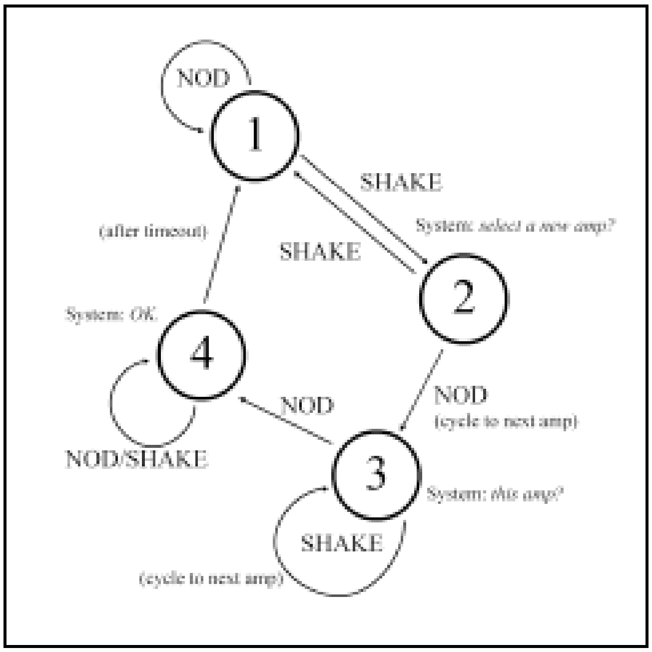



**Figure 2: State-machine diagram for gestural interaction**

## 4. EVALUATION

The system is usable and entertaining. It is also relatively user-independent, although the FaceSense head-tracker works best in a dark or dim environment free of specular light sources. The mapping of head-distance from the camera to a wah effect has been received particularly well. A problem with the head-tilt interaction is that it can be difficult for the performer to keep his face within the camera's field-of-view. The visual feedback of the performer's face on the computer screen makes this coordination possible with some practice, but a head or shoulder-mounted device [8] could improve on the usability of the system.

In addition, the gestural "yes/no" interaction, which currently requires the performer to be looking at the screen, could be improved by moving this feedback to another channel, such as audio or tactile.

## 5. CONCLUSIONS AND FUTURE WORK

This project represents an early step in using head-tracking software for both continuous and gestural mappings to musical output. It would be interesting to continue the current work by using a true face-tracker which could extract socially-meaningful representations of facial expressions (anger, fear, surprise, etc..). However, even with more meaningful feature-extraction, finding compelling mappings for the output of such a system will continue to be a challenge. As parameterized audio effects processors swell in sophistication, and the human input and acoustic output of these music-creation tools become decoupled to a greater extent, the problem of mapping becomes increasingly complicated. In time, machine-learning-based tools could be developed to supplement or perhaps even replace the human in the difficult mapping task. A system which collects either implicit or explicit feedback from the user could learn optimally pleasing mappings of the detected features.

## 6. ACKNOWLEDGMENTS

Thanks to Ashish Kapoor for his generous help with the FaceSense tracking software.

## 7. REFERENCES

[1] A. Kapoor. Real-Time, Fully Automatic Upper Facial Feature Tracking. Proceedings of The 5th International Conference on Automatic Face and Gesture Recognition 2002, Washington D.C., May 20-21, 2002.

[2] B. Reeves, C. Nass. The Media Equation: How People Treat Computers, Television, and New Media like Real People and Places. Cambridge University Press/CSLI, New York, 1996.

[3] C. Breazeal. Designing Sociable Robots. MIT Press, 2000.

[4] C. Morimoto, D. Koons, A. Amir, and M. Flickner. Pupil detection and tracking using multiple light sources. Technical report, IBM Almaden Research Center, 1998.

[5] D. Rokeby. Personal website. http://www.interlog.com/~drokeby/

[6] Line 6 Pod v2.0, http://www.line6.com/

[7] M. Lyons and N Tetsutani. Facing the Music: A Facial Action Controlled Musical Interface. Proceedings, CHI 2001, Conference on Human Factors in Computing Systems, March 31 - April 5, Seattle, pp. 309-310.

[8] M. Lyons, M. Haehnel & N. Tetsutani. The Mouthesizer: A Facial Gesture Musical Interface. Conference Abstracts, Siggraph 2001, Los Angeles, p. 230.

# Sonic Banana: A Novel Bend-Sensor-Based MIDI Controller

Eric Singer
LEMUR
New York, NY
eric@ericsinger.com

## ABSTRACT
This paper describes the Sonic Banana, a bend-sensor based alternative MIDI controller.

## Keywords
Interactive, controller, bend, sensors, performance, MIDI.

## 1. DESCRIPTION
The Sonic Banana (Figure 1) is a MIDI controller and alternative musical instrument in the form of a 2 foot long flexible rubber tube. Four bend sensors, each 5" long, are mounted in a row along the inside of the tube on a flexible metal bar running the length of the tube. A single pushbutton switch is mounted at the top end of the tube.

The pushbutton and sensors connect to a BX24 microcontroller which converts data from these inputs into MIDI messages. The four bend sensors send MIDI continuous control messages 1-4, respectively. Bending the tube in different locations changes the control value corresponding to the sensor in that location.

The pushbutton switch sends a MIDI note-on with pitch 60 and velocity 127 when depressed and 0 when released.

Data from the Sonic Banana can be sent into Cycling 74 Max and processed to create a variety of algorithmic improvisational instruments. Examples of performance algorithms are described below.

The Sonic Banana may also be used to generate MIDI data for directly controlling another MIDI instrument. This can be done by mapping the pushbutton and controllers as presented, inside a synthesizer, or by reprogramming the firmware on the BX24 to generate different controller values more appropriate to the particular application. Alternatively, of course, control data may be remapped in Max before being sent to a synthesizer.

## 2. BACKGROUND
Bend sensing has been incorporated into a number of alternative MIDI controllers, including body suits such as the MidiDancer [1] and MIBURI [2], gloves such as the Lady Glove [3] and Mattel PowerGlove-based instruments, and the CyberShoe [4]. The approach with the Sonic Banana differs from these instruments in that bend sensing is not used to sense joint positions or muscular flex, but rather to provide a direct gestural user interface.

The Sonic Banana is an attempt to create an expressive instrument with an unusual performance interface using a "less is more" approach – to create a rich musical output from a limited set of control streams. After the mapping and generation algorithm has been set, the interface encourages experimentation to explore what sort of bending and twisting gestures produce interesting musical effects.

## 3. PERFORMANCE PATCHES
Examples of improvisational performance algorithms developed in Max include an Arpeggiator and a Harmonica Simulator.

In the Arpeggiator patch, the pushbutton is used to start and stop arpeggiation. The bend sensors control various parameters of the algorithm such as speed, note duration, velocity, chord type, root pitch and pitch bend, with their ranges being scaled appropriately to the parameters.

The example Max patch in Figure 2 shows one particular mapping of the sensors. Sensor 1 selects one of six chord interval sets stored in the coll object (maj, dom7, min, min-maj7, min7, dim). Sensor 2 controls both note velocity and duration of the makenote object. Sensor 3 is mapped to pitch bend. Sensor 4 controls the speed of the metro object.

In the Harmonica Simulator, data from all four sensors is first summed to obtain a single overall bend value. This value is then sampled and differenced to derive a bend velocity value. The bend velocity value is scaled and used as force in a simplified mass-momentum algorithm to simulate pushing a mass along a surface, with adjustable coefficients of friction and air resistance. On each sample, the algorithm uses the following formula

$$v_{t+1} = v_t + F_t / m - \mu * m - d * v_t$$

where F is the force (scaled bend velocity), $\mu$ is the coefficient of friction and d is the drag coefficent due to air resistance.

The velocity of the mass maps to MIDI volume, simulating the air flow into a harmonica. A chord is played and held when absolute velocity goes above a small threshold, with positive velocity selecting one chord and negative selecting another. The pushbutton changes the pair of chords selected by velocity.

Values for mass, coefficients and scaling were experimentally determined. A reasonable set of values which gave a good "feel" to the simulation were as follows: sample period of 50 mS, bend velocity scaled by 0.5, m = 0.4, $\mu$ = 10, d = 0.001.

Example video of these algorithms can be found at http://ericsinger.com/SonicBanana.

## 4. FUTURE DEVELOPMENT
In addition to ongoing performance algorithm development, two improvements to the instrument firmware are planned. One will enable scaling and mapping of the sensors and pushbutton via MIDI input; another will put several useful performance algorithms on-board so the instrument may directly drive a synthesizer or sound module. Algorithms may then be selected from the MIDI input using patch changes.

## 5. REFERENCES

[1] Coniglio, M., MidiDancer, http://www.troikaranch.org/mididancer.html.

[2] IEEE Spectrum, 1997, The Yamaha Miburi System, http://www.spectrum.ieee.org/select/1297/miburi.html.

[3] Sonami, L., Lady's Glove, http://www.sonami.net/lady_glove2.htm.

[4] Paradiso, J., Hu, E. and Hsiao, K.-Y. The CyberShoe: A Wireless Multisensor Interface for a Dancer's Feet. Proceedings of International Dance and Technology 99,, Tempe, AZ; FullHouse Publishing, Columbus, OH (2000), pp. 57-60.

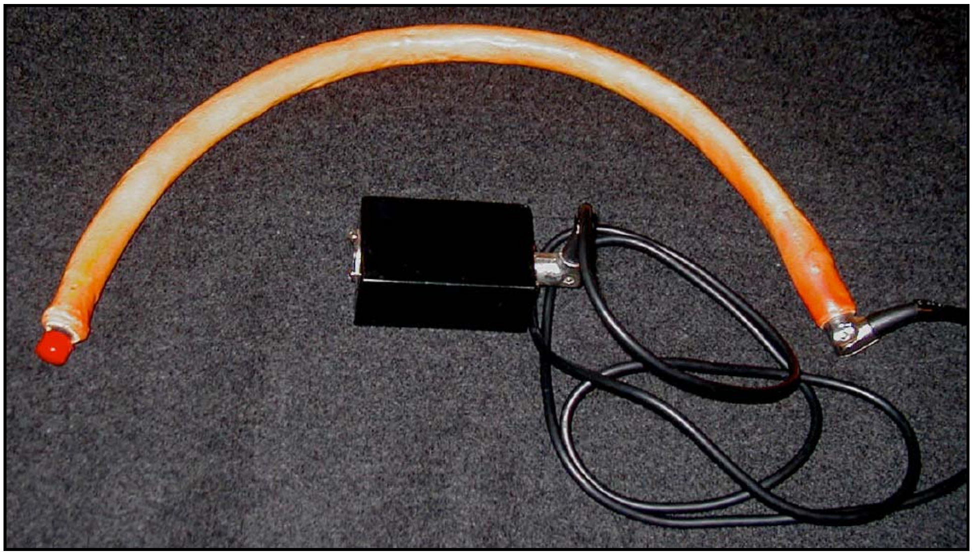


**Figure 1: Sonic Banana**

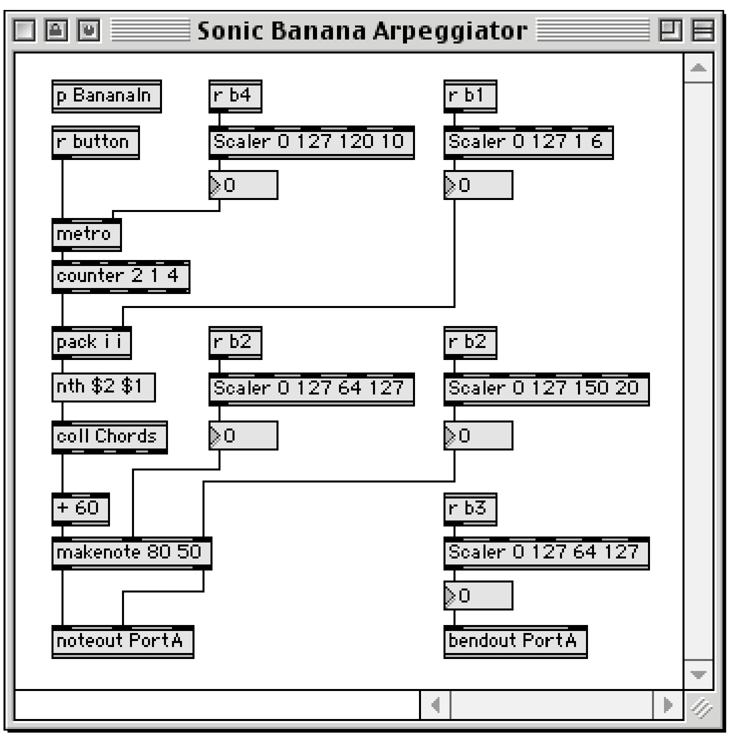



**Figure 2: Arpeggiator Max Patch**

# Sodaconductor

David Muth
Ravensbourne College of
Design and Communication
54 Lower Clapton Rd
London E5 ORN
+44 7956 696955
david@test.at

Ed Burton
Soda Creative Ltd
17-25 Cremer Street
London E2 8HD
+44 20 77396217
ed@soda.co.uk

## ABSTRACT
Sodaconductor is a musical interface for generating OSC control data based on the dynamic physical simulation tool Sodaconstructor as it can be seen and heard on http://www.sodaplay.com.

## Keywords
Sodaconstrucor, Soda, Open Sound Control, Networked Performance, Physical Simulation, Generative Composition, Java Application, Non-Linear Sequencing.

## 1. INTRODUCTION
Sodaconductor consists of a dynamic physical simulation generating control data that is fed into a local network, and which can be interpreted by custom built audio patches authored in packages such as MAX/MSP or Supercollider.

This simulation is a modified version of the Sodaconstructor online construction kit as it can be seen and heard on http://www.sodaplay.com. the software adds certain functionality which turns a Sodaconsturctor model into a 'musical instrument' - or being more precise into an interface to a musical instrument - by transmitting dynamic properties using the OSC (Open Sound Control) protocol that can be utilized by several instances of sound generating software.

## 2. MODEL SIMULATION
Sodaconstructor allows the user to create models ranging from life-like creatures to abstract animations using digital masses, springs and muscles.

Sodaconductor uses information of simulated Sodaconstructor models such as node positions, spring/muscle tensions or node - edge collisions, brings it into an for music software understandable format using the OSC protocol and transmits it to other computers. It hereby gives the user complete control on which data is being used.

In contrast to the physical modeling approach to synthesis as described by Castagne and Cadoz (Physical Modeling Synthesis: Balance Between Realism and Computing Speed), the flexibility of the framework *Sodaconductor / OSC / interpreting audio software* (figure 2) allows Sodaconductor also to be used for purposes such as non-linear sequencing.

As with Sodaconstructor, simulated models (and as a result the generated control data) can be interacted with either by grabbing the model itself or by changing parameters of the model's environment such as gravity or friction.

## 3. THE USER INTERFACE
An additional interface (figure 1) allows the choice of how many springs, nodes or collisions to transmit per frame of running simulation and certain parameters can be randomized on the fly.

Limiting the amount of information to be sent each frame will result in a generally more stable system.

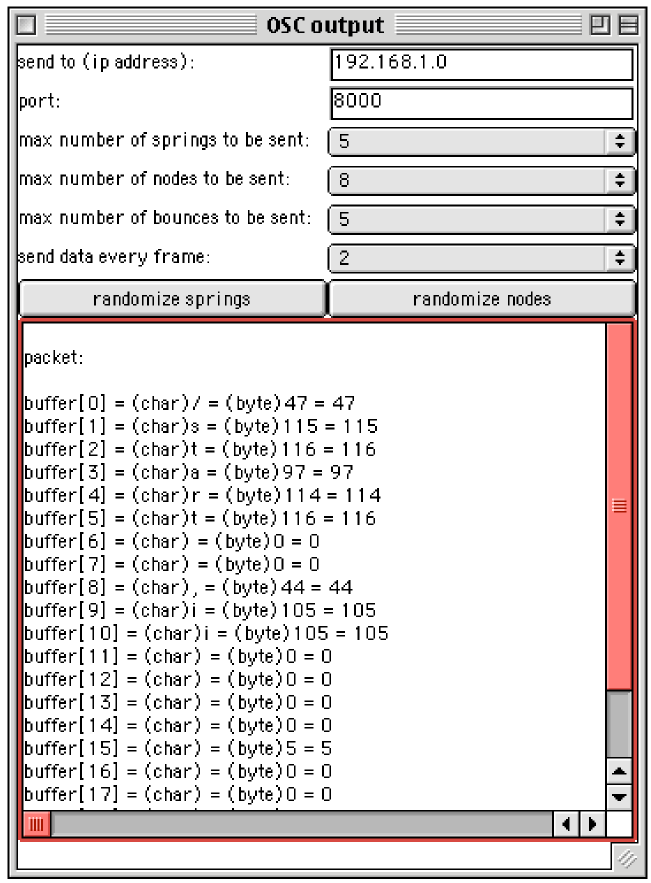



**Figure 1**

Which springs and/or nodes are being sent via OSC is being visualized in the main model simulator window (figure 4). These Springs and Nodes are being displayed in an orange color instead of a white one. If a specific node or spring is desired to be transmitted, it can be added as a 'performer' by clicking on it.

## 4. A SHORT INTRODUCTION TO OPEN SOUND CONTROL
OSC is a protocol for communication among computers, sound synthesizers, and other devices developed by Matt Wright.

In a view instances OSC has been described as a protocol that will at some point take the place of the nowadays aging MIDI.

At present the OSC protocol implementations are available for software packages such as MAX/MSP, PD, SuperCollider and Reaktor.[1]

## 5. SODACONDUCTOR AS A NON-LINEAR TOOL FOR PROVIDING CONTROL DATA

In addition to the possibilities of using Sodaconductor for providing real time physical simulation control data for influencing sound synthesis, as stated above, Sodaconductor also can be viewed as a tool for non-linear sequencing purposes.

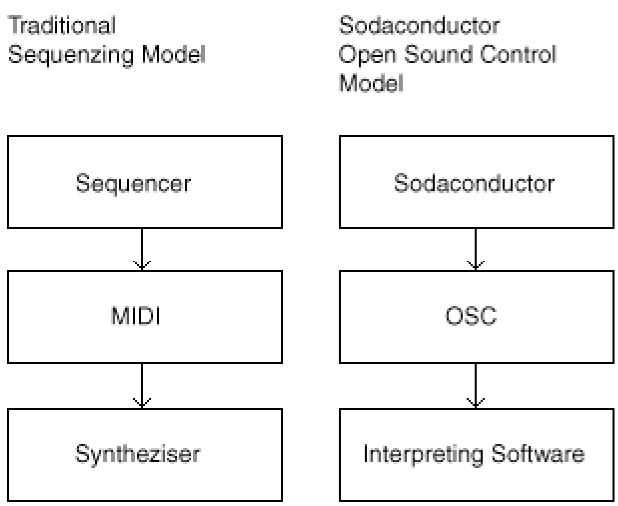



**Figure 2**

## 6. IN PERFORMANCE CONTEXT

Using this application within a networked performance context (as OSC is transmitted via a Local Area Network) enables several performers to improvise with each other whilst referring to a common thread: the control data that is being transmitted (Please see figure 3).

How this control data dynamically influences the sound generated results from the implementation of each interpreting audio patch and by choices made during the performance by the individual performers.

For example, in recent performances the movement of Sodaconductor models have been used for the spatialization of various sounds within a souround sound set-up.

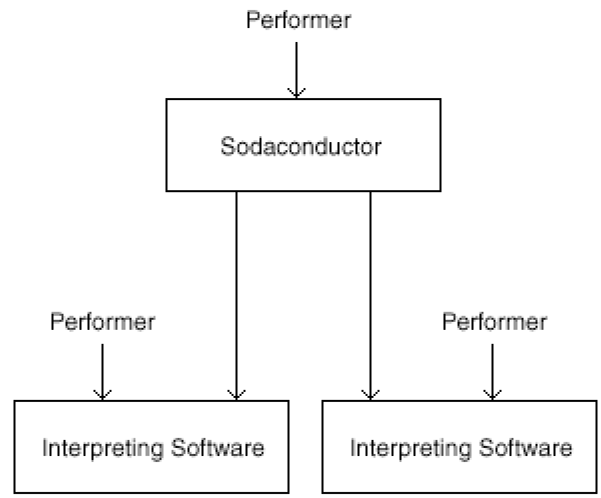



**Figure 3**

## 7. ABOUT THE SOFTWARE

In contrast to Sodaconstructor, a tool which is accessible online, Sodaconductor is an application which has to be installed locally due to certain Java (the programming language the software has been authored in) security restrictions.

At the moment Sodaconductor is still in beta – state, with a release for the public planned in the near future.

## 8. ADDITIONAL INFORMATION

Sodaconstructor is software originally developed by Ed Burton, Research and Development Director of Soda Creative Ltd, and recently won the Interactive Arts award in the 2001 BAFTA (British Academy for Film and Television Arts) Interactive Entertainment ceremony in London.[2]

The Open Sound Contol add on which turns this software into a tool that can be used within a networked audio visual environment has been implemented by David Muth.[3]

The first public performance based on this networked framework using Sodacoductor took place at the Montreal based FCMM festival 2001 under the title 'Sollbruchstelle' / Gcttcatt + Soda, in collaboration with Austrian artist Mathias Gmachl.[4]

Technical requirements for the demo:

Large Monitor or Projector, 2 Macintosh Computers running Mac Os 9 (one provided by Soda), Speakers (preferably a HIFI System)

[1] For more information please visit the Open Sound Control website: http://cnmat.cnmat.berkeley.edu/OpenSoundControl/

[2] http://www.sodaplay.com/
http://www.soda.co.uk/play/
http://www.soda.co.uk/works/sodaconstructor.htm
http://www.soda.co.uk/members/ed.htm

[3] http://www.soda.co.uk/explore/osc.htm
http://www.soda.co.uk/members/david.htm

[4] http://www.fcmm.com/2001/html/prog_e_perf.php
http://web.fm/

## 9. ACKNOWLEDGMENTS

Thanks to Mathias Gmachl for Inspiration and Julian Saunderson and Thomas Willomitzer for help and advice on the Sodaconstructor OSC add on.


## 10. REFERENCES


[1] Wright, M., Open Sound Control http://cnmat.cnmat.berkeley.edu/OpenSoundControl/

[2] Castagne, N., and Cadoz, C.,Physical Modeling Synthesis: Balance between Realism AND Computing Speed, Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00), Verona, Italy, December 7-9, 2000 http://profs.sci.univr.it/~dafx/Final-Papers/pdf/Castagne_DAFx2000.pdf

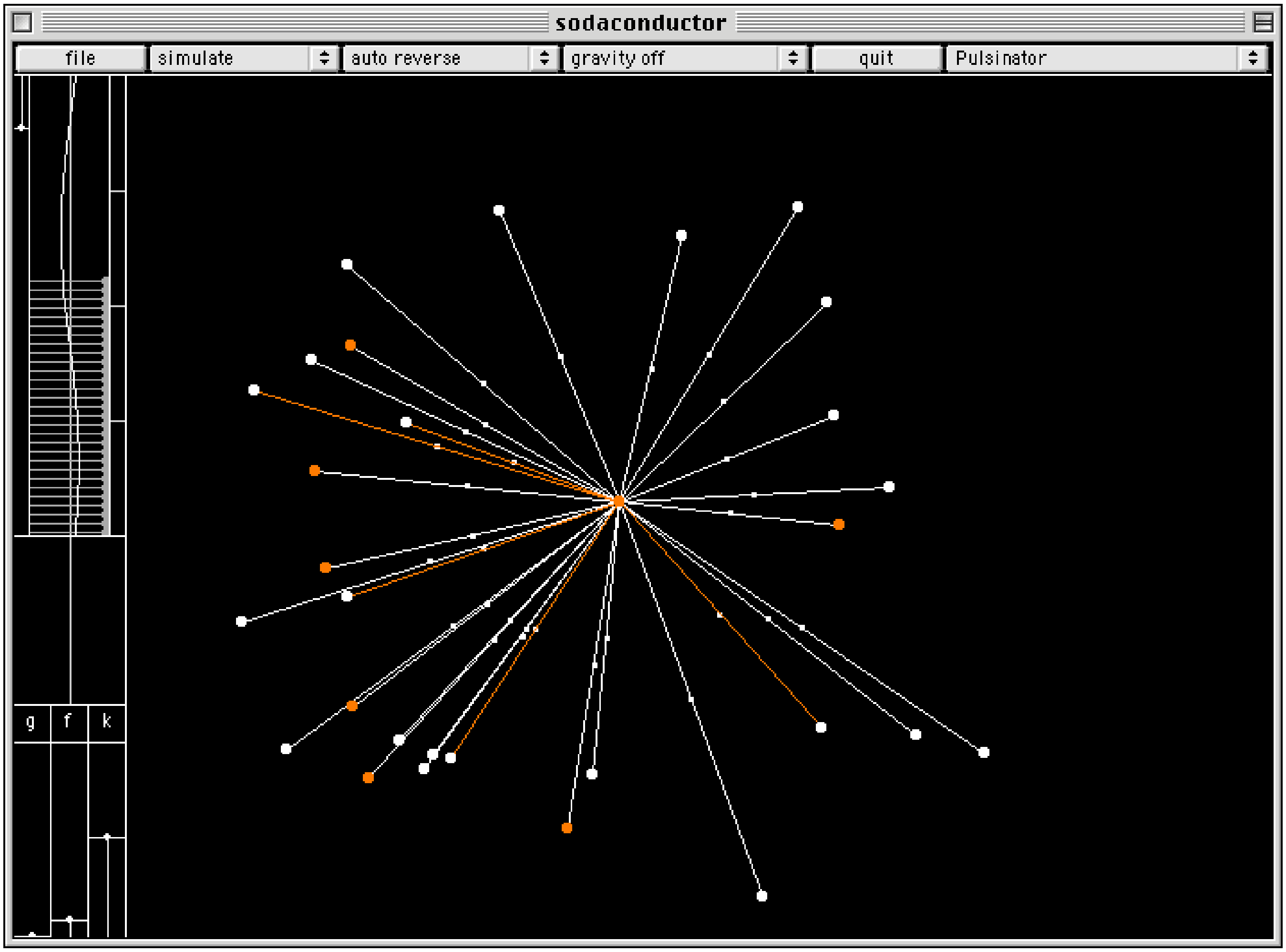




**Figure 4**

# EoBody : a Follow-up to AtoMIC Pro's Technology

Emmanuel Fléty
Engineer - Creation Department
Ircam - Centre Pompidou – 1 pl. Igor Stravinsky
75004 Paris – France
+33/1 44 78 48 43
emmanuel.flety@ircam.fr

Marc Sirguy
Director
EoWave CEO - 7, rue Rouvet
75019 Paris - France
+33/1 40 05 17 18
mesi@mesi.fr

## ABSTRACT

Ircam has been deeply involved into gesture analysis and sensing for about four years now, as several artistic projects demonstrate. Ircam has often been solicited for sharing software and hardware tools for gesture sensing, especially devices for the acquisition and conversion of sensor data, such as the *AtoMIC Pro* [1][2]. This demo-paper describes the recent design of a new sensor to MIDI interface called *EoBody*[1]

## Keywords

Gestural controller, Sensor, MIDI, Computer Music.

## 1. INTRODUCTION

The design of our previous sensor digitizing system aimed to be as versatile as possible [2]. This attempt was quite successful, thus satisfying : it showed that despite of the numerous contexts in which the device was used, the multiple functionalities solved the problem. The multi-configuration feature was also a real asset, especially for universities and art schools where several students shared the same unit. Nevertheless, *AtoMIC Pro* also appeared to be over-dimensioned for some applications and also quite expensive to build (and so to sell). So, our next wish was to design another analog to MIDI interface more adapted to electronic music and to end-users who would like to be able to experiment with sensors in their home-studio at a reasonable price (half the price of high-ends existing interfaces).

## 2. NEW DESIGN, OTHER COMPROMISE

The goal of this design was not to replace *AtoMIC Pro*. In order to reduce size and cost, we had to choose what to keep and to suppress from our original design.

We really meant to keep attractive features from *AtoMIC Pro* but the cost aspects lead us to remove the LCD and so the onboard configuration system[2]. Of course we kept the facility of a stand alone mode : once configured, the setup is stored in a non-volatile memory. Only one configuration setup can be stored in the unit at this time, but libraries can be stored on the host computer that runs the configuration editor.

The number of inputs has been decreased to 16, but we added 3 onboard potentiometers (knobs) and 4 buttons (switches) : thus, the user does not need to waste any analog inputs for implementing some "classic" sensors. The buttons and potentiometers, like the analog inputs, can be mapped to any MIDI message.


**Figure 1 : The EoBody unit**

All these significant modifications and restrictions might lead us to think this interface was getting closer and closer to other commercial products such as the I-Cube system. Nevertheless, what motivated the design of such an interface was the latency and sample rate aspect. As a matter of fact, existing systems are still clocked at a sampling period varying between 1 and 4 ms (or more) and we really wanted to keep the converter running as fast as possible : the fewer sensors you use, the faster it goes. The quality of the sampled gesture depends on this [3].

While considering sampling performances we also got interested in the portability aspect. Thus, we decided to port into the C language some software modules from AtoMIC Pro. In order to keep the system fast, critical sections (sampling and interrupts, among others) were written in assembler, but most of the code is easily portable and upgradeable.

The consequence of this port is a slight increase of the scan latency which now reaches 150 µs per active input[3] which leads us to a total scanning latency of 2.5 ms with all the 16 inputs activated.

**Table 1 : Comparison of quoted Analog to MIDI Converters (adapted from [3])**

| *Interface* | *AtoMIC Pro* | *EoBody* | *Digitizer* |
|---|---|---|---|
| Manufacturer | Ircam | Ircam – EoWave | Infusion Systems |
| Platform | Any | Any | Any |
| Max SR [Hz] | 1000 / Active inputs | 900 / Active inputs | 244 / 240 (12/7 bits) |
| Analog IN | 32 | 16 + 3 pot. | 32 |
| Digital IN | 8 | 4 switches | - |
| Input Res. | 10/7 bits | 10/7 bits | 12/7 bits |
| Outputs | switches + 4 MIDI | MIDI + merger | 8 switches + MIDI |
| Size (HWD) [mm] | 38x165x225 | 30x160x115 | 34x121x94 |

[1] http://www.forumnet.ircam.fr/
http://www.eowave.com/human/eobody.html

[2] The 4 line LCD represents about one fourth of the manufacturing cost.

[3] See the comparison chart (Table 1)

# 2. TECHNIQUES & IMPROVEMENTS

## 2.1 Sensor bandwidth

Designing a "light" version of a device does not mean removing all interesting aspects of the regular version. Since MIDI is (still) deadly slow compared to the amount of data we want to export, we kept our implementation of the noise gate algorithm and the sub-sampling process [2]. Thus, priority can be given to sensor, and gesture (or sensor) noise can be removed, with significant gain on the dataflow bandwidth. This feature also allows the user to adjust a pertinence criteria on each sensor, reducing the post-processing of the digitized sensor.

## 2.2 Running status

Strongly concerned by the temporal resolution of gestural acquisition, we have implemented this very well known technique, introduced into the MIDI standard a few years after the birth of the protocol[4] : it consists in suppressing the status byte of a MIDI message[5] if it hasn't changed from the previous message. Since *EoBody* is mainly designed to export continuous values thought *Control Change* MIDI message, we can easily assume that most sensors might be configured to talk on the same MIDI channel[6]. The status byte remains the same for the scan of a whole array of sensors and can thus be omitted.

Our *running status* algorithm actually sends/repeats the status byte every 32 sent messages to avoid a major drop when the status does not change. Improvement can easily be calculated, on 32 messages, as the following table shows :

**Table 2 : Running status improvement**

| | No Running Status | Running Status |
|---|---|---|
| 1st message | 3 bytes | 3 bytes |
| messages 2 to 32 | 3 bytes | 2 bytes |
| | Total = 96 | Total = 67 |

The bandwidth thus increases of :

$$100 \times (1 - \frac{67}{96}) = 31\%$$

## 2.3 High resolution vs. standard MIDI

Again, considering temporal resolution aspects, it was obviously not a good idea to send high resolution data (10 bit wide in our case) through System Exclusive MIDI messages, the involved number of bytes being a disaster for the transmission time of the digitized info. So, we rather preferred single or combination of standard MIDI messages like *control change* (7 LSBs + 3 MSBs) or *pitch bend* (the 10 bits being mapped on the 10 MSBs of the message).

## 2.4 Sensors' range

To minimize the difference of sensors range, the voltage reference of the A/D converter is still accessible, via a trimmer. Despite the fact that signal windowing is unavailable, signal zooming can be achieved using the 10 bit resolution and then scaling the value elsewhere.

## 2.5 Connectors and plugs

A known problem with *AtoMIC Pro* was the "complex" connector used to connect sensors (i.e. a *Sub-D* plug). We still think that wiring is extremely important, and that's why *EoBody* still uses *Sub-D* connectors, with locks. However, to keep plugging sensors easy, we include in the *EoBody* package 2 splitter cables that distribute the 15 pin *Sub-D* male plug to 8 regular *jack* plugs. Thus, the user can choose between easy (but quite bad quality, we all know how good a *jack* plug is) connection for experimenting, and very secure wiring for a performance or an installation. We also underline that exporting the *jack* plugs through a splitter cable allowed us to reduce the size of the housing box[7]. Moreover, it is easier to repair a splitter cable with regular connectors[8], than having to open the whole box for repairs.

The danger of using jack connections would be sensors hot-plugging. People who have experimented expression with pedals know the problem well : inserting the male jack plug with the power on creates a little short-circuit that sometimes makes the fuse burn. Most systems actually have a current- limiting circuit to avoid this, but it is impossible to simply implement in our device, since we do not know how much current the connected "sensor" will sink. So, we have chosen to use a *Polyswitch*™ fuse : it allows transitory peaks of current, but cuts off the power supply if too much current is drawn[9] : there is no risk of flashing fuse after fuse just by connecting a sensor.

# 3. REFERENCES

[1] Fléty, E., "Interactive devices for gestural acquisition in the musical live performance context." *Proceeding of SCI,* SCI 2001 – Orlando, Florida – USA, pp 535-540.

[2] Fléty, E., "AtoMIC Pro : a Multiple Sensor Acquisition Device." *Proceedings of NIME,* NIME 2002 – Dublin, MIT Europe – Ireland, pp. 96-101.

[3] Wanderley, M., "Performer-Instrument interaction : applications to gestural control of sound synthesis". *Doctor's Degree Thesis*, Université Paris 6 – France, pp 41-42.

4 `www.midi.org`

5 Most MIDI messages are composed of three bytes, one being the status, i.e. the command to be executed (note generation, parameter modification etc.), and the two others being the data (7 bit long) bytes of the message.

6 Even if the MIDI channel can be individually set, input per input.

7 Made of steel for a better hardness.

8 *Sub-D* and *jack* plug can be found anywhere and are really cheap.

9 The resistance of the *Polyswitch* increases with temperature. It rearms itself when the temperature decreases (thermal fuse).

# Invited Papers

# Dual-Use Technologies for Electronic Music Controllers: A Personal Perspective

Joseph A. Paradiso
Responsive Environments Group
MIT Media Laboratory
1 Cambridge Center, 5FL
Cambridge, MA 02142
(617) 253-8988
joep@media.mit.edu

## ABSTRACT

Several well-known alternative musical controllers were inspired by sensor systems developed in other fields, often coming to their musical application via surprising routes. Correspondingly, work on electronic music controllers has relevance to other applications and broader research themes. In this article, I give a tour though several controller systems that I have been involved with over the past decade and outline their connections with other areas of inquiry.

## 1. FROM PHYSICS TO INSTRUMENTS

People devote an amazing amount of energy into developing new modes of musical expression. There's nothing quite like the satisfaction that one gleans after building and playing a new instrument, feeling its response, and hearing sounds that have never been produced before. Although most of the NIME audience is quite familiar with the technical literature in computer music (e.g., *Computer Music Journal*, *Journal of New Music Research*, *Leonardo Music Journal*, *Organized Sound*, etc.), the periodical *Experimental Musical Instruments*, and Bart Hopkins' books [1,2] give an excellent survey of a wider grassroots movement where artisans of all sorts bend their abilities into crafting new ways to create and shape sound. Indeed, people heralding from many daytime callings cross-fertilize all sorts of ideas and approaches from many fields into musical instruments.

When growing up, I was hardly immune to this muse; like many of my colleagues coming of age in the generation of consumer electronics, I essentially learned circuits by building various devices that made sounds, often buying old gear left over from the Boston area's extensive high-tech, military-industrial R&D at local surplus houses and hacking it to make music. Perhaps in my case, the expression got a little extreme in the 140-module homebrew patchable synthesizer that evolved in my basement during the 70's and early 80's [3]. In addition to being a source of unusual sounds, it is very much an intimate musical controller. Despite the drawbacks of being too closely wedded to the world of atoms, with a knob and patchcord on every signal, modular synthesizers provide a highly fine-grained, tangible, and parallel interface into sonic structure. Although it's a little rusty now, I'm not ready to surrender that axe to pasture…

**Figure 1: The homebuilt modular in my former basement**

Since electronic music systems by definition rely on a fresh supply of ideas and technology to keep things current, instrument inventors and developers often tap the accessible edge of Moore's Law. Some fascinating stories can be found where this trend is pushed to its extremes, e.g., the initiative by North American Rockwell to push large-scale integrated circuit technology directly from the space program into musical instruments, resulting in the Allen Digital Organ, the world's first real-time digital wavetable synthesizer, which appeared on the market way back in 1971 [4,5]. This example illustrates how a mixture of different perspectives can lead to a disruption in an established field. Innovation seldom comes out of comfort; it often arises from a cultural clash [6], which frequently manifested testy circumstances as Allen engaged with Rockwell [5].

One would think that experimental high-energy physics would have little effect on electronic music controllers, but indeed it has, through several avenues. Since Bob Bowie had worked with Veljko Radeka's Instrumentation Group at Brookhaven National Laboratory, he was well aware of capacitive pickup electronics for cathode-strip drift chambers (standard charged-particle detectors) [7]. This proved to be the inspiration for the sensor system that he designed with Max Mathew's for the Radio Baton [8], one of today's best-known alternative controllers. As both Bob and Max knew Neil Gershenfeld through Bell Labs, these ideas propagated further into the Media Lab's cello bow controller [9] used in Tod Machover's Hyperstring performances [10]. At that time, I was also using capacitive sensing technology, but in high-energy physics applications at Draper Laboratory, this time using a stretched-wire to sense the precision alignment of drift chamber packages for the muon system of the proposed GEM detector at the Superconducting Supercollider (SSC) [11].

Upon joining the Media Lab in 1993, I pushed these technologies into a wireless violin bow tracker and a free-gesture controller for our *Sensor Chair* [9]. When designing the sensor suites for the *Brain Opera* performance interfaces [12], I again adapted technologies that we had developed earlier for aligning high-energy physics detectors. In particular, the *Digital Baton* [12,13] used an optical tracker based around a position-sensitive photodiode (PSD) that we had evaluated at Draper for GEM's optical straighness monitoring [14], and the laser rangefinder design that I turned into a hand tracker and musical interface for large projection walls [15,16] was inspired by a rangefinder that we had intended to use for dynamic detector surveying [17].

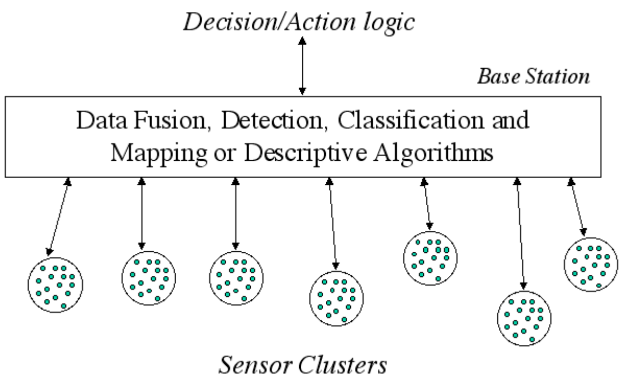



**Figure 2: Wireless sensors in a star topology**

## 2. HIGH-DENSITY WIRELESS SENSING

Although interfaces for electronic music face some very interesting research challenges on their own turf [18], this section will provide a few examples that illustrate how particular controller designs that we've pursued address broader research issues - essentially taking an opposite tack to the previous discussion. In particular, the goals of Ubiquitous Computing [19], which envisions sensors, processing, and communication moving into everyday objects and environments, form a good match to technical research in many avenues of musical controller design.

The sensor topology described in the next two sections is the centralized "star" with a heavy basestation, as portrayed in Figure 2. This topology is well suited, for example, to a wearable sensor array used in a dance performance, where one needs to rapidly acquire all information from every sensor cluster on the stage without the latency that would be incurred in a peer-peer network as shown in Figure 6.

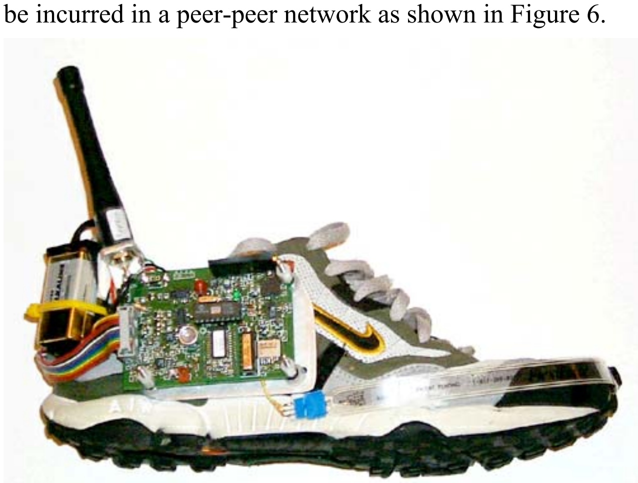


**Figure 3: The final version of the *Expressive Footwear* shoe**

When I first conceived of the *Expressive Footwear* project [20] in 1997, I wanted it to be a wireless sensor tour-de-force. Knowing nothing about dance, I threw every sensor that would fit and seemed even vaguely useful onto a dance sneaker, with a wireless datalink coming directly off the shoes. In the end, we put 16 diverse sensors on each shoe to measure many parameters of contact and free gesture together with position. We developed a series of such shoes between 1997 and 2000 [21].

As the first devices were deployed before compact sensor packages, such as the Motes [22,] became established, it was somewhat of a radical statement, an early case of what I call "sensing as commodity", partly inspired by the various dexterous glove interfaces developed at STEIM [23]. Traditional sensing applications have been based on measuring only parameters of direct relevance. When designing an artifact needing measurement, sensors are traditionally placed exactly where they're needed to provide primarily the information required. Now that sensors are becoming so inexpensive and small, however, we can look at pursuing another, less stringent strategy that involves packing as many sensor measurements as possible into the object's form factor. If there's any suspicion that a measurement can be at all relevant, and if it can fit into the package constraints, just include it as a member of a large embedded sensor suite. This way, a host of multimodal sensor readings catch many features of activity and expression – instead of "sharpshooting" particular parameters of interest with explicit sensors, this approach catches a wide range of phenomena with multisensorary "buckshot", allowing one to reconstruct a variety of features and states by fusing the data in software. This allows an instrument designer or player to be more open to serendipity – the rich sensory stream produced by such a heavily instrumented controller captures many types of gesture, enabling a user to map an effective response to many types of activity and usage modalities that weren't anticipated when the device was designed. In the case of the *Expressive Footwear*, this was indeed the case – after perfecting a compact circuit card to do such dense wireless sensing and survive on the foot of a dancer (a major challenge in itself [24]), the data stream was sufficiently rich to map expressive response onto many different styles of dance across the wide range of dancers that we worked with in this project.

We have since miniaturized this instrument package further, producing a device that we call the sensor "*Stack*" [25] that is composed of circuit cards roughly an inch and a quarter on a side. Mating at small connectors on their perimeter, different such cards can be vertically layered, allowing a designer to stack up a suite of sensing devices into a compact form factor, roughly the area of a large wrist watch. One card contains a 22 MIPs processor and RF transceiver; subsequent cards encapsulate different sensing modalities. At the moment, we have developed two sensor cards (a 3-axis inertial measurement unit [IMU], and a tactile input device that interfaces to pressure [FSR and piezo], bend, and capacitive sensors), and a sonar card is under development. Our current application of this platform is in medical biomotion diagnosis and therapy, where we're trying to use a heavily instrumented shoe to enable some of the function of a high-infrastructure gait laboratory at a hospital to be accommodated in a small doctor's office or home environment [26]. We are also planning to use this platform as a research tool to investigate state-driven processing and resource allocation in sensor nodes. As energy, computation, and communications bandwidth tend to be quite limited in battery-powered systems, sensor nodes have to take careful account of what sensors are used, what features are extracted, and what data is transmit [27]. Accordingly, appropriate processing at each node can extract a limited amount of local context in order to dynamically adjust this resource balance. Instead of blindly and wastefully dumping all measured bits all of the time, a more efficient sensor node will send only relevant features at appropriate times.

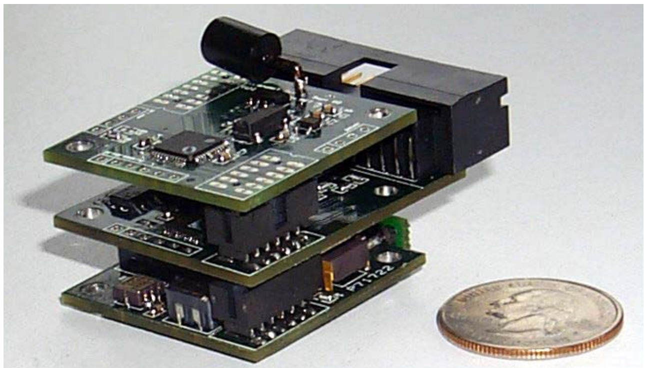


**Figure 4: The currently working version of the *Sensor Stack***

In the near future, we intend to explore the application of our *Stack* in ensemble dance, where we instrument the hands and feet of a small troupe of dancers. By upgrading the 115 kbit/second RF transceiver that we're using to a 1-2 Mbit/second capacity and running a simple TDMA protocol, we anticipate being able to maintain a 100 Hz full state update from each node of this system for 4-5 dancers, effectively capturing many features of real-time dance performance. In addition to just building an architecture to acquire the data, this system will confront significant technical challenges in real-time data fusion in order to produce a prompt and relevant media response to the 300-500 parameters streaming in with each measurement update. There are likewise issues involved in content mapping here – we can no longer map our data directly at the sensor level, as is now conventional in MIDI mapping packages like MAX, since there's just too much dissimilar data streaming in to deal with by hand. Metavariables defined at a higher level, reflecting information relevant to the performer (perhaps inferred affect [28], synchronicity and deviation, energy, learned or entrained parameters, etc.), will need to be defined in order to effectively author content on top of these systems.

**Figure 5: Low cost "jerk" sensor to instrument large crowd**

### 3. FEATHERWEIGHT SENSORS

We have also been pushing another dimension in high density wireless sensing. Instead of making heavy nodes that each host many degrees of sensing freedom, we have developed a system that supports huge numbers of extremely lightweight nodes that each measure only one coarse parameter. This system has been targeted at interactive entertainment for large groups. Whereas Loren Carpenter's camera-driven Cinematrix [29] effectively and economically enables a large group to be instrumented with passive optical targets, kinetic musical expression, such as interactive dance, can have difficulty with the line-of-sight and lighting constraints that video-based approaches require. Accordingly, we have developed [30] an extremely compact wireless sensor that sends a narrow RF pulse out when it's jerked. As the active duty-cycle is so brief and since the circuit needs no complex components, a small, onboard watch battery lasts years of regular use. The device, manufactured in large quantity, is so inexpensive that it can be given out at sports games or dance raves as a party favor with the ticket, enabling participants to contribute some level of group control over interactive media. We have derived a set of real-time statistics from the data stream that indicate the level of activity, mean tempo, and significant events with many coincident hits, and have used these features to define parameters exploited by an interactive music system for MIT dance parties [31]. Although the results were intriguing, the area of interactive entertainment for large groups is still quite open – maintaining some degree of collective consonance and causal engagement with scores of participants is a difficult, if not impossible challenge [32].

These minimal wireless "featherweight" sensors have many applications in other areas. We will soon deploy them in "smart home" environments that monitor overall patterns of activity for elder care – a significant and growing problem, since so many seniors are living alone and unattended. Much more noninvasive than a camera or microphone, and potentially more reliable, these minimal sensor packages can be affixed to doors, furniture, cabinets, etc., where they will produce a wireless response to associated activity. By monitoring patterns evident in the wireless signals, deviations in habits can be detected, potentially indicating an evolving medical problem.

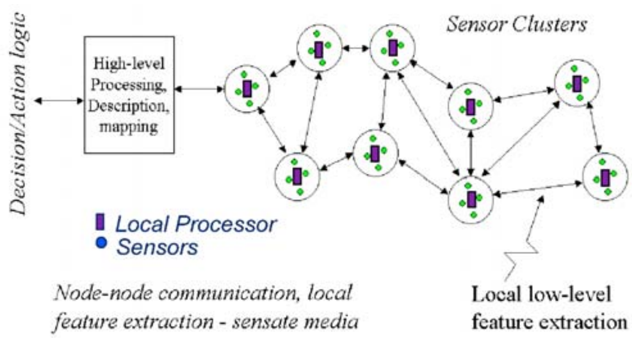



**Figure 6: Dense peer-peer sensor network**

## 4. ELECTRONIC SKINS

Another interesting frontier in dense, multimodal sensing is posed by the concept of sensate electronic skins. Applications abound in areas like robotics, telepresence, medical diagnostics, and prosthetics for very dense tactile arrays that approach the sensory capabilities of biological skin. Similarly, significant technical challenges are posed here in fabrication, microelectronics, and signal processing [33]. Today's tactile arrays (e.g., FSR [34] and fiber optic matrices [35], "smart skin" for aircraft wings [36], etc.) are all heavily multiplexed; a dedicated processor essentially scans all sensor cells and looks at each piece of data. Accordingly, these centralized systems have difficulty scaling up to large arrays because of the mass of wiring and data involved. In order to feasibly build such systems, processing must be blended smoothly into the sensing substrate. A rough inspiration can be taken from biology, where signals from tactile and other sensor receptors are combined and preprocessed in the nervous system, often before reaching the brain [37]. Hence, a possible manifestation of electronic skin involves a peer-peer, ad-hoc sensor network, much as has been proposed for battlefields, cities, and buildings, but shrunk down to a mm node spacing. In this scenario (Figure 6), a processor manages a group of local sensors (a mix of different types can be included to enable multimodal sensing – e.g., pressure, temperature, proximity, etc.), collecting and processing the resultant data, and communicating with its neighbors. When a stimulus occurs, the processors will cluster, characterize, and isolate it, thereupon routing the resultant high-level features out node-node to an external portal, suppressing the granular detail.

Such electronic skins could provide a very promising technology for advanced musical interfaces, as they possess both a high-resolution, multimodal, tactile sensing capability together with the possibility of local optical, tactile, and possibly acoustic display via actuators connected to each processor that are driven via a distributed control scheme. Musical performance or installation applications place tight requirements on the latency of response (depending on the instrument or interface, roughly 1-100 ms of delay can be tolerated), hence routing and internode communications protocols and topologies must be appropriately constrained.

**Figure 7: 100 Pushpin nodes pushed into their substrate**

Since the challenges here are considerable, we have developed a few hardware testbeds with which we can conduct experiments in dense sensor networks and begin to explore applications of such electronic skins. The first, "*Pushpin Computing*," [38] is composed of a large, sandwiched conductor/insulator power plane and an array of small processors with configurable communication and sensing/actuation capabilities (via a set of layered boards, as in the *Stack* described above). As the bottom layer of the Pushpin sports a pair of unequal-length insulated pins connected to the local power lines, Pushpins can be pushed into the power plane at any position, where they pull power from the conductors and establish communication (currently via IR) with their neighbors. Accordingly, the Pushpin system is highly configurable and has been used to test dynamic routing in sensor nets [39].

Another testbed now nearing completion is called the "*Trible*" ("Tactile Reactive Interface Based on Linked Elements") [40]. Shown in Figure 8, it is essentially a soccer ball tiled with 32 Circuit card "patches", each hosting a 22 MIPs processor and an array of up to 18 sensors, including pressure transducers, piezoelectric cantilevers bonded to fibrous "whiskers" that protrude from holes in the surface, microphones, temperature monitors, and light sensors. As each card also supports a small audio speaker, a vibrator, and an RGB LED, all nodes are capable of providing a direct, multimodal response. There is no central control in this system – the patches only talk to their neighbors, hence, as in Figure 6, they collectively process the sensor information and coordinate their local responses and/or route the processed features out to an external connection. Although we have yet to exploit its musical potential, with 516 channels of multimodal sensing and local actuation, the *Trible* promises to open up some interesting avenues of music control and distributed sound generation.

**Figure 8: The *Trible*, before installation of its whiskers**

**Figure 9: A few assembled *Z-Tiles* under test**

The last device in this category is a collaboration between the *Interaction Design Group* at the University of Limerick and the Media Lab's Responsive Environments Group called the "*Z-Tiles*" [41]. Partially shown in Figure 9, it is an array of interlocking, puzzle-shaped floor tiles, each of which hosts an array of five processors and a set of force-sensitive resistors, each roughly 3 cm in diameter. When the tiles are interlocked, a mating connector routes both power and digital data tile-tile, hence a sensor network is built up as the floor is assembled. Contrary to the previous sensate floors on which it was based (e.g., our *Magic Carpet* [16] and Limerick's *LiteFoot* [42]), which involved heavy cabling infrastructure that limited their span, the *Z-Tiles* are intrinsically scalable. Upon detecting pressure, neighboring tiles will communicate to isolate and characterize footsteps, then route the resulting features node-node to an attached computer that can provide an appropriate response. As the *Z-Tiles* were designed for interactive dance, the routing and processing routines need to be sufficiently prompt to avoid introducing excessive delay when passing messages across the maximum span of tiles in a given installation (and with a given amount of foot traffic). Although prototype tests of a half-dozen linked tiles have been completed, this system is currently under development. The resultant floor is planned to be used not just in entertainment, but also in "smart home" applications, where gait can be characterized and occupants tracked [43] throughout a responsive space.

**Figure 10: The *Musical Trinkets* engaging a Crowd in Milan**

## 5. OTHER EXAMPLES

Many other technologies that have made their way into the world at large have started from or been inspired by musical controllers. Force-sensitive resistors (FSR's), common components used for moderate-resolution pressure sensing in many applications, were perfected by a founder of Interlink [44] for sensing aftertouch on keyboard interfaces. The first conceptual implementation of spread-spectrum communication, posed by actress Hedi Lamarr and the composer of *Ballet Mechanique*, George Antheil, was based on the sequencing principles of a player piano [45].

I've been able to participate in pushing a few other musical controller designs into a range of applications. The swept-frequency tag reader that I designed for the *Musical Trinkets* installation [46] was inspired by Electronic Article Surveillance (anti-shoplifting) systems [47]. The *Trinkets* hardware is now evolving further into a 3D volumetric tracker for passive tags [48]. Although this has many potential applications in augmented and virtual reality (e.g., various control points on objects, fingers, etc. can be wirelessly tagged and tracked), this incarnation was inspired by the need to tag and precisely track the position of a tumor on a patient undergoing radiation therapy [49].

The *Sensor Chair* [9] is another controller that has had particular success outside of the musical realm. It began its life in 1994 as a transmit-mode capacitive sensing system to track free gesture at the arms and legs of a seated occupant, in this case, the magicians Penn and Teller, who used it to perform a mini-opera by Tod Machover together with a comedic séance [50]. Two attendees took special notice of this device in its performance debut at MIT's Kresge Auditorium that summer. One was the current agent for the Artist formerly known as Prince. After a convoluted series of events that is difficult to summarize, this connection culminated in one of the strangest musical interfaces that I've ever built, the *Sensor Mannequin* [9], an electric-field-sensing monstrosity probably stored somewhere deep in Paisley Park now. The other interested attendee at this event was from the North American division of NEC Automotive. He saw the *Sensor Chair* as a potential solution to a persistent problem in automotive safety, namely a sensor system that could determine whether or not to fire a car's airbag during a collision based on the status of the facing seat's occupant (several infants had recently been killed by airbag deployments when their car seat was not properly oriented). After adapting some of the innards of the *Sensor Chair* system, then prototyping and testing many layouts for sensate seats, they have moved to product with the Elesys *Seat Sentry* [51], now a feature on several cars in current production. Closing the circle, Motorola has recently released a 9-channel capacitive-sensing chip for this system, the MC33794 [52]. Originally inspired by the electronics in our chair, this device is a useful building block for musical controller builders wanting to work multichannel capacitive proximity sensing into their interaction portfolio.

**Figure 11: Bono enjoying the Sensor Chair at MLE, Dublin**

## 6. CONCLUSIONS

Electronic Music Controllers have absorbed technology, ideas, and innovators from many fields of inquiry and practice. Conversely, developments in musical interfaces have also contributed concepts, inspiration, and products to entirely different areas of application. The field is very much a melting pot, where artists and technologists hailing from many different backgrounds come together to exchange perspectives. Such environments are fertile incubators for new and disruptive concepts. Musical controllers also can provide excellent testbeds and challenges though which to explore and demonstrate ideas in areas like Ubiquitous Computing. Yes, at the end of the day in this field, the show is what counts the most. But along the way, interesting tributaries lead to territories that could never have been imagined beforehand. It's been a wild ride, and there's plenty of water still out there, so hold onto the hull and keep exploring!

## 7. ACKNOWLEDGEMENTS

I've touched on several projects in this paper that many people have contributed to, hence they all deserve acknowledgement. My past and present students in the Media Lab's *Responsive Environments Group* have been instrumental in most of these efforts; in particular, Kai-Yuh Hsiao, Ari Benbasat, Josh Lifton, Mike Broxton, Eric Hu, and Stacy Morris have been principal in many of the projects mentioned here. My early work on violin bows, sensor chairs, and the Brain Opera is what got me started in this area, thanks very much to a longtime collaboration with Tod Machover.

Similarly, Neil Gershenfeld was the first to extend me the invitation to switch from my world of physics detectors and sonar systems at Draper Lab to the universe of bows and gesture controllers at the Media Lab (as this article entails, it's hard to keep both feet in any one place!). Several alumni were vital to some of the early projects noted here, particularly Ed Hammond, Pete Rice, and Josh Smith, who hung on through the *Sensor Chair* and *Brain Opera*. And, of course, many other students, faculty, and researchers at the Media Lab contributed to the efforts described here and are thanked for making the place the cutting edge that it is. I've delighted in exchanging ideas with my many colleagues doing musical interfaces at other institutes. In particular, Mikael Fernstrom from the University of Limerick has been a resource and friend in many projects beyond just the Z-Tiles. And, of course I give special thanks to Max Mathews for not only being a pioneer and inspiration to us all, but for also being perhaps the field's greatest boundary-breaker and cross-fertilizer as he brought disparate talent and strong ideas together to push music deep into the future.

## 8. REFERENCES


[1] Hopkin, B., *Gravikords, Whirlies & Pyrophones: Experimental Musical Instruments,* Ellipsis Arts, NY, 1996.

[2] Hopkin, B., *Orbitones, Spoon Harps & Bellowphones*, Ellipsis Arts, New York 1998.

[3] Paradiso, J.A., "Diary of a Teenage Synth Hacker," Forward to Kettlewell, B., *Electronic Music Pioneers*, ProMusic Press, Vallejo, CA, pp. 4-9, 2002. See also: http://www.media.mit.edu/~joep/synth.html

[4] Carson, B., "The World's First Digital Sample-Playback Synthesizers," *Keyboard*, March 1995, pp. 38-46.

[5] Markowitz, J., *Triumphs and Trials of an Organ Builder*, Allen Organ Publishing, Macungie, PA, 1989.

[6] Paradiso, J.A., "From Tangibles to Toolkits and Chaos to Convection - Management and Innovation at Leading Design Organizations and Idea Labs," Presented at the Workshop on *Managing as Designing: Creating a vocabulary for management education and research*, Weatherhead School of Management, Case Western Reserve University, Cleveland OH, June 14-15, 2002. To be Published, Stanford University Press.

[7] Fernow, R., *Introduction to Experimental Particle Physics,* Cambridge University Press, 1986.

[8] Mathews, M.V., "Three Dimensional Baton and Gesture Sensor", US Patent No. 4980519, Dec. 25, 1990.

[9] Paradiso, J., Gershenfeld, N. "Musical Applications of Electric Field Sensing", *Computer Music Journal*, 21(3), 1997, pp. 69-89.

[10] Machover, T., *Hyperstring Trilogy*, Oxingale Records, April, 2003.

[11] Paradiso, J., "New Technologies for Monitoring the Precision Alignment of Large Detector Systems," *Nuclear Instruments and Methods in Physics Research* A386, 1997, pp. 409-420.

[12] Paradiso, J., "The Brain Opera Technology: New Instruments and Gestural Sensors for Musical Interaction and Performance," *Journal of New Music Research*, 28(2), 1999, pp. 130-149.

[13] Marrin, T. and Paradiso, J., ``The Digital Baton: A Versatile Performance Instrument," *in Proc. Int. Computer Music Conf. (ICMC'97)*, pp. 313-316.

[14] Paradiso, J., "Testing and Development of Extended Range Straightness Monitor Systems", GEM Collaboration Report (SSCL), GEM-TN-93-331, May 1994.

[15] Paradiso, J., *et al.*, "Sensor Systems for Interactive Surfaces," *IBM Systems Journal*, Volume 39, Nos. 3 & 4, October 2000, pp. 892-914.

[16] Paradiso, J., *et al.*, "New Sensor and Music Systems for Large Interactive Surfaces," *Proceedings of the International Computer Music Conference (ICMC 2000),* pp. 277-280.

[17] Hashemi, K.S., Hurst, P.T., and Oliver, J.N., "Sources of Error in a Laser Rangefinder," *Review of Scientific Instruments* 65, No. 10, 3165-3171 (1994).

[18] Paradiso, J.A., and O'Modhrain, S., "Current Trends in Electronic Music Interfaces," *Journal of New Music Research*, to be published, July 2003.

[19] Weiser, M., "The Computer for the Twenty-First Century," *Scientific American*, pp. 94-10, September 1991.

[20] Paradiso, J. and Hu, E., "Expressive Footwear for Computer-Augmented Dance Performance," in *Proc. of the First International Symposium on Wearable Computers*, Cambridge, MA, IEEE Computer Society Press, Oct. 13-14, 1997, pp. 165-166.

[21] Paradiso, J., *et al.*, "Design and Implementation of Expressive Footwear," *IBM Systems Journal*, Volume 39, Nos. 3 & 4, October 2000, pp. 511-529.

[22] See: http://webs.cs.berkeley.edu/tos/

[23] Blaine, T. (Bean), "Tech: A Soft Touch," *Electronic Musician*, June 1998, pp. 106-109.

[24] Paradiso, J., "FootNotes: Personal Reflections on the Development of Instrumented Dance Shoes and their Musical Applications," in Quinz, E., ed., Digital Performance, *Anomalie, digital_arts* Vol. 2, Anomos, Paris, 2002, pp. 34-49.

[25] Benbasat, A., Morris, S., Lovell, D., Paradiso, J., "A Wireless Modular Sensor Architecture and Application in On-Shoe Gait Analysis," Submitted to the *IEEE Sensors Conference,* 2003, Toronto, CA.

[26] Morris, S.J. and Paradiso, J.A., "A Compact Wearable Sensor Package for Clinical Gait Monitoring," *Offspring* Vol. 1, No. 1, pp. 7-15, January 31, 2003.

[27] Raghunathan, V., *et al.*, "Energy Aware Wireless Microsensor Systems," IEEE Signal Processing Magazine, March 2002, pp. 40-50.

[28] Camurri, A., Ricchetti, M., Trocca, R., "EyesWeb - Toward Gesture and Affect Recognition in Dance/Music Interactive Systems," *ICMCS*, Vol. 1, 1999, pp. 643-648.

[29] Carpenter, L., "Cinematrix, Video Imaging Method and Apparatus for Audience Participation," US Patents 5210604 (1993) and 5365266 (1994).

[30] Feldmeier, M. and Paradiso, J., "Ultra-Low-Cost Wireless Motion Sensors for Musical Interaction with Very Large Groups," Presented at the *UBICOMP 2001 Workshop on Designing Ubiquitous Computing Games*, ACM UBICOMP Conference Proceedings, Atlanta GA, Sept. 2001.

[31] Feldmeier, M., Malinowski, M., Paradiso, J., "Large Group Musical Interaction using Disposable Wireless Motion Sensors," in the *Proceedings of the ICMC 2002 Conference,* International Computer Music Association, San Francisco CA, pp. 83-87, September 2002.

[32] Ulyate, R. and Bianciardi, D., "The Interactive Dance Club: Avoiding Chaos in a Multi-Participant Environment," *Computer Music Journal*, Vol. 26, No. 3, Fall 2002, pp. 40-50.

[33] Lumelsky, V., Shur, M.S., Wagner, S., "Sensitive Skin," *IEEE Sensors Journal*, Vol. 1, No. 1, pp. 41-51 (2001).

[34] Papakostas, T., Lima, J., and Lowe, M., "A Large Area Force Sensor for Smart Skin Applications," *in Proc. of the IEEE Sensors 2002 Conference*, Vol. 2, pp. 1614-1619.

[35] See: http://www.tactex.com/

[36] Udd, E. (ed.), *Fiber Optic Smart Structures,* Wiley, 1995.

[37] Kandel, E.R., Schwartz, J.H., and Jessell, T.M., *Principles of Neuroscience*, 4th Ed., McGraw-Hill, 2000.

[38] Lifton, J., Seetharam, D., Broxton, M., Paradiso, J., "Pushpin Computing System Overview: a Platform for Distributed, Embedded, Ubiquitous Sensor Networks," in F. Mattern and M. Naghshineh (eds): *Pervasive 2002*, Proceedings of the Pervasive Computing Conference, Zurich Switzerland, 26-28 August 2002, Springer Verlag, Berlin Heidelberg, pp. 139-151.

[39] Lifton, J., *et. al.*, "A Sensor Network Connectedness Determination Algorithm and Implementation," submitted to the ACM Sensor Network Conference (*SenSys*), 2003.

[40] Lifton, J., Broxton, B., and Paradiso, J., "Distributed Sensor Networks as Sensate Skin," Submitted to the *IEEE 2003 Sensors Conference*, Toronto CA.

[41] McElligott, L., *et al*., "'ForSe FIElds' – Force Sensors For Interactive Environments," in Borriello, G. and Holmquist, L.E. (Eds.): *UbiComp 2002*, Springer-Verlag Berlin Heidelberg, LNCS 2498, pp. 168-175, 2002.

[42] Griffith, N. and Fernström, M., "LiteFoot: A Floor Space for Recording Dance and Controlling Media," *Proceedings of the 1998 International Computer Music Conference*, International Computer Music Association, San , CA (October 1998), pp. 475-481.

[43] Orr, R.J. and Abowd, G.D., "The Smart Floor: A Mechanism for Natural User Identification and Tracking," in *Proceedings of the CHI 2000 Conference on Human Factors in Computing Systems: Extended Abstracts*, ACM Press, New York (2000), pp. 275-276.

[44] See: http://www.interlinkelec.com/

[45] Braum, Hans-Joachim, "Advanced Weaponry of the Stars," *American Heritage of Invention and Technology*, Vol. 12, No. 4, Spring 1997, pp. 10-17.

[46] Paradiso, J.A., Pardue, L.S., Hsiao, K., and Benbasat, A., "Electromagnetic Tagging for Electronic Music Interfaces," To appear in the *Journal of New Music Research*, July 2003.

[47] Lichtblau, G.J. Resonant Tag and Deactivator for use in Electronic Security System, US Patent No. 4,498,076, Feb. 5, 1985.

[48] Paradiso, J.A., Hsiao, K., "Multiple-axis tracking of passive resonant structures," US Patent No. 6,404,340, June 11, 2002.

[49] Dr. Paul G. Seiler, Paul Scherr Institute (PSI), Villigen, Switzerland, personal communication, 1999.

[50] Machover, T, *Media Medium* (musical score), Ricordi, Milan/Paris, 1994. Debut performance at the *Digital Expression Symposium*, MIT Kresge Auditorium, October 20, 1994.

[51] http://www.elesys.co.jp/

[52] Ohr, S., "Non-contact sensor discerns passenger size for airbags," EE Times, February 6, 2003.

# ACROE - ICA
# Artistic Creation and Computer Interactive Multisensory Simulation Force Feedback Gesture Transducers

Claude Cadoz, Annie Luciani, Jean-Loup Florens, Nicolas Castagné
ACROE - ICA
INPG, 46 Av. Félix Viallet - F-38000 Grenoble
Ministère de la Culture et de la Communication
Claude.Cadoz@imag.fr, Annie.Luciani@imag.fr, Jean-Loup.Florens@imag.fr, Nicolas.Castagne@imag.fr
www-acroe.imag.fr

## Introduction

The ACROE and the ICA laboratory are two associated institutions running a program of research, development, creation and pedagogy in the field of computer for music and animated image synthesis. ACROE was founded in 1976 by Claude Cadoz, Annie Luciani and Jean-Loup Florens, in the *Institut National Polytechnique* of Grenoble (INP-G) with the support of French Ministry of Culture. The ICA (Artistic Creation and Computer) was initially a team, belonging successively to the LCP (*Laboratoire de la Communication Parlée* - INP-G), from 1976 to 1985, to the LIFIA (*Informatique Fondamentale et Intelligence Artificielle* - IMAG), from 1985 to 1995, and to the CLIPS (*Communication Langagière et Interaction Personne-Système* - IMAG). Since 1999, ICA is itself a laboratory of INP-G. ICA is in charge of the scientific research part of the global program.

The team built up from 1976 with fundamental aims motivated by the computer entering in several fields of artistic creation. The deep mutation represented by the computer, regarding the technology history, required a new and fundamental analysis of the role of the material tools in artistic creation as well as the role of the computer itself as tool. New concepts and theories were necessary that cannot be simply deduced from the previous.

The following few points, from this time, determined the basic positioning of the group:

- The computer being essentially a universal tool for *representation*, founded on the concepts of information, symbolic operations, language, etc. cannot be envisaged as a simple extension of the previous tools and instruments founded on physical, mechanical, energetic processes.
- The creation process, in "instrumental" arts is supported by a hierarchy of tools and systems, from a physical level (the musical instrument for example), to a conceptual level (the musical theories for example). But even if the theoretical evolutions of esthetics can be at the origin of technical changing in the physical tools, the technology, at a moment of the history, determines fundamentally the range and the limits of the possibilities. So, the first level at which we must start, while there is a so deep change in technology, is the most elementary: the physical instrumental level.
- The physical instrumental interaction is *sensory-motor* and *multisensory*. Sensory-motor: every perception needs always a kind of action, and every action is always more or less accompanied by a perception. Multisensory: every perception, even through a specific channel, is always more or less correlated with perceptions on other channels.
- The computer, as a general representation system, must be used at a first stage for the level of the instrumental interaction. This level, then, will be the base for the development of the higher levels.

The computer was then envisaged to introduce explicitly a new mediation level in the creation process, through the paradigm of the interactive and multisensory simulation (IMS) of physical objects. This principle and its correlated techniques are the core of the computer creation tool envisaged. The higher level functionality being built from this base.

These points determined the general program axes:

- Research and development on the technical conditions for multi-sensory-motor interaction with computer: a substantial point here, is the concept and technology of force-feedback gesture transducers (TGR®, for *Transducteurs Gestuels Rétroactifs*, in French).
- Definition of a modeling and simulation language of physical objects - The CORDIS-ANIMA language.
- Designing and implementation of hardware and software architectures for real-time simulation - The TELLURIS platform.
- Study and research on the human interaction modalities, of the instrumental interaction, the instrumental gesture, the haptic perception and the multiple and multisensory action-perception loops.
- Research and development on the user interfaces for artistic (musical, animated images, multisensory) creation - The GENESIS and MIMESIS environments.
- Exploration, in scientific and artistic ways, of the universe of multisensory models of physical objects and phenomenon.
- Research and development on application of IMS and TGR in scientific and industrial fields.
- Application in artistic and pedagogic activities.

## 1. The force-feedback gesture transducers (TRG®)

To allow an instrumental gesture interaction with the computer, we have to take into account the fact that this interaction is bi-directional: from the human being to the computer, and conversely. The transducers, whatever their type (acoustic-electric, electro-acoustic, mechanic-electric or electro-mechanic, etc.) work, by their technologic principle, in a unidirectional way. So, the first characteristic of a gesture transducer able to support an instrumental interaction is that it must be, by principle, a pair of symmetrical unidirectional transducers: a force or a displacement sensor paired with an actuator (motor).

The physical phenomenon within the gesture action and perception (tactilo-proprio-kinesthesic perception) are complex and accurate. There are several important criterions that we must consider carefully in the design of retroactive gesture transducers:

- the number of degrees of freedom (DOF), (our hand has itself 23 degrees of freedom),
- the bandwidth of gestural signals (at least 1KHz),
- the resolution of displacements (up to 1µm),
- the range of forces (up to several hundred N),
- the range of displacements (at least ~ 1m).

One of the greatest difficulties in the design of such devices is to respect all these criterions at the same time in the same device assuming that under a sufficient level, the gesture interaction has no relevance at all.
And then, above all, a particular difficulty is to provide a well-adapted geometry (or *morphology*: shape and displacement trajectories,…) to the physical component we manipulate. Due to the physical contact and interaction, on the contrary to the electro- acoustical or visual transducers, it is not possible to build a universal mechanical architecture available for every application.

We worked on this set of problems since 1978. The first TGR was built by Jean-Loup Florens in 1978 [FLORENS (JL), 1978].

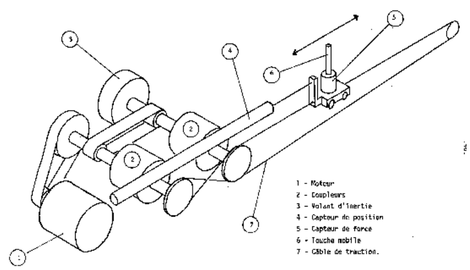



**Fig.1a - On the left : First TGR® from ACROE - J-L. Florens (1978)**

This device, sensing forces and displacements at its manipulation stick was able to produce a force-feedback of several tens of N with a time response of about 1ms, and with a displacement range of about 1m. It allowed for the first time to evaluate the importance of the force-feedback in the manipulation of simple virtual objects. It allowed also to highlight, from decisive experiences, the inter-sensory phenomenon and its importance (for example, the influence of the visual perception on a correlated tactile perception, and conversely).

A second device was built in 1981 by Claude Cadoz and Jean-Loup Florens [CADOZ (C), LUCIANI (A), FLORENS (JL), 1984, 1989], (fig.1).

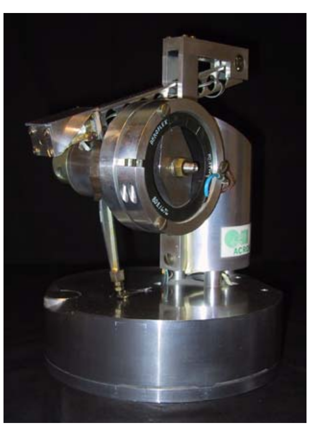


**Fig. 1b : "la Touche", C. Cadoz, J-L. Florens (1981)**

This piano key device, more compact and with better performances that the previous was the first table device. But it had always only one degree of freedom.

It allowed carrying out the first actual real-time multisensory interaction.

The next step was the invention of the concept and technology of *Modular Retroactive Keyboard* (CRM®) and of its associated principle of *Slice Motor*® [CADOZ (C), LISOWSKI (L), FLORENS (JL), 1987, 1989, 1990, NOUIRI (J), 1994, 1995], (fig.2).

This concept solved the two crucial problems within the TGR: modularity in terms of number of degrees of freedom, and modularity in terms of manipulation morphology.

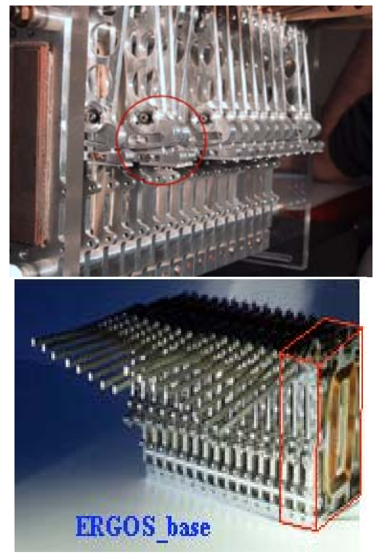
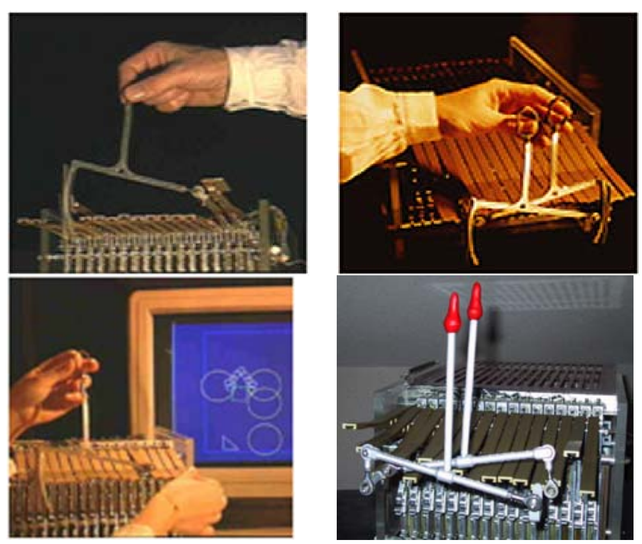



**Fig.2 - The Slice Motor® - C. Cadoz, L. Lisowski, J-L. Florens, 1987**

A unique rectilinear magnetic field is provided by a set of magnetic "slices" (one per key) inserted between flat coils (one per key) driving each key. This principle allows getting a strong magnetic field (by adding the one of each magnet) within a very compact volume. It is possible to add any number of keys in a complete modular way, according to the number of degrees of freedom needed.



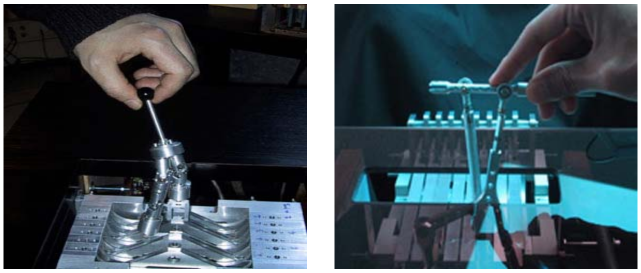


**Fig.3 - Morphological modularity: several "*habillages*".**

The "*habillages*" are added mechanical components, designed according to the specific applications, that convert the rectilinear and parallel sensor-actuator axes provided by the bloc to the manipulated object trajectories.
The first version (1987), with 16 DOF, 13,75mm width per slice, 3µm of minimum displacement detection, 80N of maximum force and 1ms of response time, was the most performant in its category. Thanks to this device, J-L. Florens [FLORENS (JL), 1988] achieved the first real-time simulation of a bowed string played with a real gesture, showing the fundamental role of the mechanical interaction between the player and the object in terms of quality and expressiveness of the sound.

Numerous other experiments achieved thanks to this equipment were decisive for the global understanding of the gesture in instrumental interaction and to demonstrate the relevance of physical modeling in this context.

After numerous analyses, experiments and improvements carried out from this period, a second family of TGR based on the same technology has been built. The maximum force provided on each DOF can reach today 200N, and the main categories of "*habillages*" have been designed and implemented: classical keyboard, 2D, 3D sticks, pliers, 6D stylet, 6D platform, 6D sphere,…
The size, variety and performances of the ACROE's TGR family give them the place of the best candidate for the future desk haptic devices.

These devices are today used for various applications: basic experiences on haptic modality in experimental psychology, telemanipulation, nanomanipulation, etc. and, of course for artistic musical, visual and multisensory creation.

## 2. The CORDIS-ANIMA language

The second important concept in the Interactive Multisensory Simulation is of course the simulation itself. So, the physical modeling, which is the background of simulation, is not simply a judicious paradigm to link with TGR, but the TGR and the physical modeling (or simulation of physical objects) are the two non-separable components of a single approach. Each of them can give the best when conceived and implemented within the context of the other. We considered these two concepts as non-separable from the beginning of our work in 1976.

The simulation paradigm holds the idea of a reference, for the mental and material representations, to the physical real objects and phenomenon. But it is only a reference, assuming that our active and cognitive system have been elaborated through permanent interaction with the real world. The laws, the regular properties of this last have equivalent and resonance in the structure and the economy of our active-cognitive system. This "ecological" point of view, related to the one of Gibson [GIBSON (J), 1979], is crucial in this approach. But it plays only as a reference since it is no longer question (and, by the way feasible) to duplicate the real world, but to provide a stable basis for invention of free imaginative and expressive representations.

Since the interaction between instrumentalist and instrument is supported, through the TGR, by two signals representing dual physical variables (intensive as input, extensive as output, or conversely), and in respect to the scale of the considered phenomenon, the conceptual framework will be the Newtonian physics. Then, the basic notions represented in the language will be inertia, spatial positions, displacements, velocities, and physical interaction laws, referenced to the classic mechanic mathematics.

Designing the simulation process, then, must be supported by a high level language avoiding the user to write the basic code for each case. The definition of the primitives is then determinant for the consistency, the generality and the relevance of the language. So, the CORDIS-ANIMA formalism [CADOZ (C), LUCIANI (A), FLORENS (JL), 1983, 1984, 1985, 1989, 1991, 1993, 1994] was completely (and exclusively) deduced from a main principle: the generalization of the interaction concept from the macroscopic scale (interaction between instrumentalist and instrument) to the components of the "object" themselves. This principle led to the general pattern of the CORDIS-ANIMA models: a network where the nodes (the <MAT> elements) are the smallest modules representing inertia, and where the links (the <LIA> elements) represent physical interactions between them. Under <MAT> is an algorithm calculating at each sampling time a position (in one, two or three-dimensional space) from a force input and according to an inertia parameter (M). Under <LIA> are several kinds of algorithms calculating the interaction forces between the two <MAT> it links, from their positions and according to basic interaction laws like elasticity, viscosity, etc.

These algorithms are designed in absolute optimized forms in regard of the number of basic numeric operations and of the calculation delays. So, the language is general and consistent.

This systemic pattern assumes that all the relations between its entities, whatever the scale (including the one of the instrumentalist itself) are bi-directional interactions. There is no *a priori* hierarchy, but any hierarchy can be decided in the modeling approach. The properties of the models are "emerging": they come exclusively from the local properties of its components and from the way they are combined in the network.

This is a strong choice having important consequences in the modeling approach and, more deeply, in the creation process itself, since we must think everything in terms of interaction.

## 3. Real-time simulators

The CORDIS-ANIMA system can be implemented on any general-purpose computer but the real-time interaction implies two major conditions: of course a maximum of computing power to simulate models of sufficient complexity, but also a specific control of the input and output protocols. A real-time simulation loop is indeed complex: it is constituted of three nested loops at three frequencies: the sound sampling frequency (the higher), the image sampling frequency and the gesture sampling frequency. The two last being a sous-multiple of the first. The synchronism must be absolute and driven by an external clock. For each input (gesture) data, the computer has to run a complete simulation loop including the nested ones and to produce the output data (for sound, image and force-feedback control).

In the early stages of development in the laboratory, the very first simulators were in fact real-time in their principle: an analogic computer has been used for the first force-feedback experiences, and a DEC LSI11 just after. The LSI11 processor was controlled by an external clock assuming a rigorous synchronism of the input and output. Of course, the simulated

models were very simple, and sometimes simulated in very low frequency, but they were real-time.
After that, we adopted two consecutive solutions for real-time simulation:

In 1982, Talin Berberyan [DARS-BERBERYAN (T), 1982, 1983] built a dedicated processor, the CTR (CORDIS Temps-réel), with a hardware implementation of the CORDIS-ANIMA algorithms, that was probably the first real-time processor for physical modeling. It allowed to simulate models of strings (or others) with about 20 to 30 masses in real-time and with gestural control. It has been replaced during the 80's by an Array Processor (AP120 from Floating-Point System inc.) that reach approximately the same performances, but with more generality.

In both of the previous cases, the simulator was a specific machine exclusively dedicated to simulation process and driven by a host computer. The next step was centered on Silicon Graphics workstations (1993) offering at this time the best compromise between the computing power, the graphical resources and the general software environment (under Unix). In this last case, as in any case today focused on any non dedicated platform, it is not possible to overcome the real-time constraints without intervention at the basic hardware (including the processor in a specific architecture allowing a complete control of the input/output data-flows) and at the basic software level (of the operating system itself in order to eliminate all operations not strictly involved in the simulation during the simulation process).

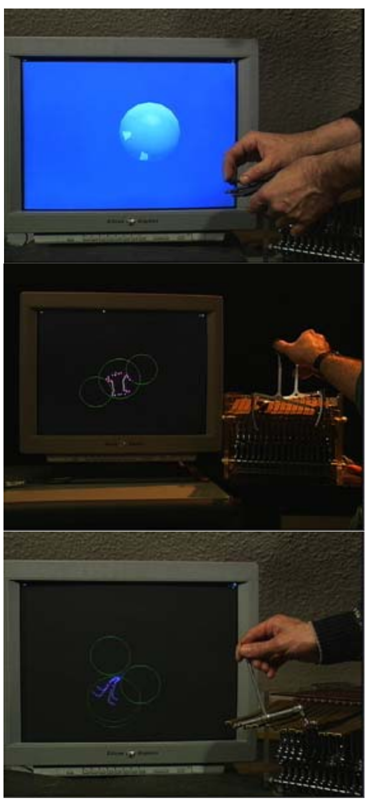


**Fig.4 - Real-time manipulation with TELLURIS**

The TELLURIS project, dedicated to real-time multisensory simulations with usual platforms, is at this time a major axe of development in the laboratory.

# 4. Study and research on the human interaction modalities

[CADOZ (C), LUCIANI (A), FLORENS (JL), 1984, 1988, 1992, 1994, 1996,1997]

The gesture, in instrumental interaction, is more than a simple "control". From a physical point of view, one can consider that during the interaction, the instrumentalist and the instrument are together a global system which properties depends intrinsically of the ones of each of them and which is more than the sum of their parts. Moreover, in the natural instrumental interactions, the instrumentalist is the source of any energy produced. From this point of view, the study of instrumental interaction concerns the physical phenomenon involved: the dynamic properties of the mechanical systems (of the instrument as of the instrumentalist itself), the energy transfers, energy transformations (from mechanical to acoustical vibrations or to visible movements).

From a communication point of view, the instrumental interaction is a combination of three nested loops, from the gesture action to the haptic perception, from the g.a. to the auditive perception, and from the g.a. to the visual perception. The study, then, concerns the human motricity, the auditive perception, the visual perception and the complex haptic perception (tactilo-proprio-kinesthetic perception). But even if these fields can be studied separately, the more relevant and needed study is about their combination in a global multi-sensori-motor process.

The study of the instrumental interaction, thanks to experimental devices or in order to develop them, suppose an investigation in each of these fields.

## 4.1 Instrumental gesture typology

[CADOZ (C), 1994, 1995, 1999]

The gestures applied on a same instrument may be of different types. But we can generally observe three main categories whatever the instrument and its function. They are determined by the principles of action, modification and selection.

In the action or *excitation* part of the gesture (for example what a violinist does with his right hand), we produce an energy, which is communicated to the instrument and converted by it into energy in the final phenomenon (the sound).

In *modification* or *modulation* part of the gesture (the left hand of violinist), we modify certain properties of the instrument, and, consequently the way in which it works in the previous function. We may spend energy in this gesture, but it is not transmit to the final phenomenon.

In the *selection* part of the gesture (ex: selection of keys in piano performance), we spend energy to move our fingers, hands, arms… in order to act on, or modify something but there is no actual effect on the instrument.

These three categories are generally intimately correlated. They are components of the gesture as a whole, but from a physical point of view, they correspond to different process and then, they are not supported by the same rules. From the communicational point of view, they don't play the same role. The *excitation* gesture has a quantitative consequence while the *modification* gesture is more qualitative. To the *selection* gesture corresponds an exploration of separate categorized elements.

One of the direct applications of this typology is that it offers criterions for gesture device design: the force feedback (which has a technological cost) is actually necessary only for the *excitation* gesture.

## 4.2 Coding, analyzing, composing the instrumental gesture

The gesture transducers introduced the *gesture signal*. So, in the same way than the *sound signal* gave rise to the Schaefferian "*Objet Sonore*", the gesture can become an *object* that can be recorded, observed, analyzed, transformed, edited, composed.

*Unfortunately it is not so simple!* The gesture interaction doesn't produce one, but two correlated (input and output) signals. Both of them can be recorded and candidate to editing. But what are the meaning and the consequences of the modifications we can apply on them? The output signal is completely determined by the input signal and the global transfer function of the virtual physical object. Conversely, the input signal is completely determined by the output (which is the input of instrumentalist) and the "instrumentalist transfer function", that is not a transfer function since the instrumentalist (hope so!) is an active component. So, too fast manners with this can be of catastrophic consequence: an arbitrary modification of one of the signals (*a fortiori* of the two) may be inconsistent within the integrity and permanence of the mechanical systems involved (instrument and instrumentalist).

This point represents the most important (and fascinating) question in this problem of understanding and treatment of instrumental gesture signal.

We laid down the basis and the theoretical solution of this problem. Assuming that the notion of gesture signal is, strictly speaking, non-definable, we replaced it by two concepts: the *gesture interaction signal*, which is solely measurable (through TGR) but that cannot express exactly the intention of the instrumentalist, and an hypothetical *internal gesture signal*, unreachable for measurement, that represents purely the gesture intention.

To define the second, Sylvie Gibet and Jean-Loup Florens [GIBET (S), FLORENS (JL), 1987, 1988] considered the instrumentalist itself as a two parts mechanical system: a purely passive one (that can be, incidentally, modeled thanks to CORDIS-ANIMA), and an ideal generator of force or displacement. Using the two interactive signals, it is possible (theoretically) to infer this model and to get the two separate parts. Editing the *internal gesture signal* so obtained, it becomes possible to create new gestures in respect of the consistence and integrity of the (virtual) instrumentalist.

This defines the rigorous theoretical context of coding, representing and editing of gesture. But it is quite complex to implement.

Some simplifications are possible when restricting to gestures with low interaction (for example in the percussive actions). Claude Cadoz and Christophe Ramstein [CADOZ (C), RAMSTEIN (C), 1990, 1991] developed a first gesture editor working solely on the input signal that allowed to create and edit *gestural scores*.

We are today working on further developments from these bases and for the definition of a standard format for the gesture representation, editing and transmission.

Through the instrument modeling, the gesture performance and the gesture editing, we have at our disposal the complete panoply to propose a new approach for music and animated images creation with computer, replacing totally the focus on the phenomenon by a focus on their causes. Of course, this approach is completely compatible with a continuation of signal modeling and editing and with composition of music and/or animated images scenarii by traditional methods.

## 4.3 Study of haptic modality and multisensory interaction

The technical equipment for gesture interaction (TGR) and interactive multisensory simulation (IMS) is, apart its application in music and image creation, ideal as experimental environment to study the human active-cognitive features.

Numerous experiments achieved randomly during the development and adjustment of the systems has soon focussed our attention on the peculiar properties of the cognitive system in interactive and multisensory situation: the role of the action during perception, the mutual influence of the sensory channels between them, or the robustness of our action-cognition system in certain cases, in spite of strong anamorphosis in the experimental conditions.

Some of these experiences have been carried out systematically in several specific works.

### *Sensory-motor anamorphosis - the "presence" notion*

During the 80's, an important evolution occurred in the Man-Machine-Interaction domain (MMI), with concepts like direct manipulation of windows, icon, and menu through pointing devices (WIMP). This evolution from a technocentric to an anthropocentric approach of MMI borne the general idea of "metaphorisation" of interaction: make the manipulation of numerical entities and processes analogous to the manipulation of simple current objects. The full outcome of this concept is probably in the Virtual Reality trend, strongly emphasized from the end of 80's [RHEINGOLG (H), 1991, KALAWSKY (R-S), 1993, QUEAU (P), 1993, CADOZ (C), 1994, KUNII (TL), LUCIANI (A), 1998] in which the laboratory has been implicated.

A specific difficulty in this domain is to get a good projection of the real or referenced scene in the virtual ones. Two obstacles arise: a reduction of the properties through the numerical representation, and several kinds of anamorphoses between the real and the virtual scenes. We dedicated some works to the second [BOUZOUITA(A), CADOZ(C), UHL (C), LUCIANI (A), 1995, 1996, 1997].

An example of first order anamorphose (purely geometric) is, for example, when we control the 2D displacements of a point on a surface by two buttons driving separately the horizontal and the vertical dimensions. Such situations may be imposed by unavoidable technologic constraints, and sometimes they can be overcome by training. But in certain cases, they may bring advantages by transposing complex manipulation in simpler ones, better adapted to our possibilities or relieved of unnecessary and perturbing features. The question is then: on what criterion can we determine if an unavoidable anamorphose is a drawback, or how can we define an anamorphose that improve the interaction? This problem cannot be studied without considering the whole context of the interaction, and, particularly in terms of multisensoriality.

So, we brought to the fore, against the general expectation that gave favor to the immersive realism, that under certain conditions, we are able to tolerate quite strong geometrical and spatial anamorphoses, for example the separation and the delocalisation of the sources of the acoustical, visual and haptic phenomenon for a same virtual object, or the transformation of the range and trajectories of its manipulated parts. More precisely, when the force-feedback devices performances are sufficient and can respect the dynamic and energetic consistence, this last is more important than the pure geometric and visual one in the sophisticated (and costly) 3D immersive systems.

Now, this is crucial, because it is quite impossible to get relevant performances in force-feedback within immersive approach. On the other hand, these performances are possible in the desk and "*vis-à-vis*" systems.

In fact, the relevant and crucial concept here, is not this notion of purely visual and superficial realism, but the "feeling of presence": what makes us convinced that the phenomenon with which or by which we act and perceive are manifestation of reality? What makes present the things? Feeling of presence is stronger than "realism": it is more efficient, as well in learning tasks as in dexterous manipulation ones. Our hypothesis [LUCIANI (A)] is that the gesture channel, by its double function (linking closely action and perception) has the major role in this question: everything being equal elsewhere, just adding the gesture channel with sufficient performance brings a gap in the presence feeling.

### *The haptic and multisensory modality*

In collaboration with the Experimental Psychology Laboratory (LPE) of the Université Pierre Mendès-France, in Grenoble, we worked to better understand the haptic modality and the gestural action-perception loop, and also to get a new understanding of the eye-hand inter-sensory coupling.

Some relevant results have been obtained recently: it seams possible to experimentally determine if a person acts preferably with reference to visual or to tactilo-propri-kinesthetic information,the person uses the forces he produces to get information on the external environment, certain forces perturb this estimation; so, there are useful and non-useful forces and certain persons are more sensible to this perturbation.

This results in new analysis methods allowing to better distinguishing between categories of people ("geometric people" and "dynamical people").

## 5. Environments and interfaces for artistic creation

The Gesture Interfaces (TGR), the simulation language (CORDIS-ANIMA), the simulator (TELLURIS) are the core of the creation tool. To complete this tool, a global environment to support the creation process must be designed.

The creation process is the series of operations we do in order to achieve an artistic work: a sound or/and a video record, a mixed numeric piece where gesture intervenes during the performance, a mixed piece where fixed numeric data, real-time numeric processes, real people, instruments, objects can perform together.

A creation environment must support the different tasks involved to achieve such works. It must be a production tool allowing to create the various objects: models, set of parameters, descriptions or recordings of real or virtual performances, sound and/or visual signals, scores, etc. But it must be also a *creation* tool, providing tutorials, supporting experimentation, suggesting heuristics, accompanying empiric explorations, allowing scribing and mnemonic marking. He must also provide tools to analyze and understand the phenomenon.

In 1985, Annie Luciani and Aimé Razafindrakoto [LUCIANI (A), 1985, RAZAFINDRAKOTO (A), 1996] designed the first modeler allowing to create graphically CORDIS-ANIMA models of 2D and 3D objects. The graphical interface was implemented in an Evans § Sutherland graphic workstation, and the models were simulated, for some of them in real-time, in the AP120 array-processor.

From the beginning of 90's, this area become an important part of our work. Because the real-time simulation techniques were at this time difficult to implement in ordinary platforms, we decided to develop a special effort in specialized interfaces dedicated to our physical modeling method. Moreover, we decided to approach independently the musical and the visual domains, which communities and needs were quite different. De facto, we are working actually and yet today on three environments: GENESIS for musical creation, MIMESIS for animated images creation (which are both non-real-time environments) and TELLURIS, which, connected with the previous, allows experiments and creation solely in the laboratory.

GENESIS and MIMESIS are both interfaces to built physical models within the CORDIS-ANIMA formalism. They work mainly in the same way, with a two modes process: creating and editing the models / running the models to produce the output phenomenon (sound signals / visual signal). Nevertheless, an important difference is that MIMESIS requires two steps in the final result production (see after).

### *GENESIS*[1]

The creation of a model in GENESIS is made by direct manipulation of icons corresponding to the basic components (<MAT>, <LIA>) of the CORDIS-ANIMA language. Among them, GENESIS uses only the unidimensional components that are simpler to manipulate and to calculate, and that are sufficient. Positioning each <MAT> and linking then by <LIA>, on a workbench, we define the network (structure) of the objet. Then, we can give parameters and initial conditions thanks to suitable windows. A first simulation with a visualization of the movements of the objects at low rate allows verifying and understanding how the model works. Putting "microphones" on certain <MAT> elements, the movements of these <MAT> are recorded in a file that becomes the output sound files.

With these very simple bases, any user (even without any previous experience or specific knowledge) can get start with the concepts and functions, obtain models that works and understand the philosophy of the approach.

Beyond this elementary layer, numerous elaborated functions supported by very studied ergonomics allow to work at different levels: analysis of the properties of the structures, grouping of elements in functional entities, define macro and metaparameters allowing to control the "emergent" properties of the models, define the energy and action time of "pilot events", etc. Within these resources, GENESIS can be used not only to create sounds, but also to compose music in a new way (see below).

A wide and structured library of generic objects and processes is provided, accompanied with several kinds of didactic interactive supports.

The first version of GENESIS was distributed under partnership collaboration in 1996. From this date, numerous improvements made by Nicolas Castagné [CASTAGNE (N), CADOZ (C), 2002] and under a close collaboration with the users allowed to realize a very powerful tool. It is, with MIMESIS, the core of a user network created in 1998 in the Rhône-Alpes Region (France), and with several European institutions, called R_APM (Pedagogic Mobile Workshop Network).

### *MIMESIS*

Apart the final object, which is an animated sequence instead of a sound file, MIMESIS, presents three other important differences:

1 [See on www-acroe.imag.fr]

- the objects are 3D
- a textual language is used to create and edit the models
- a specific visualization phase is defined after the mechanical simulation.

### *Textuel langage*

Due to the great number of particles and the complexity of the topology for the objects to be animated and visualized, a direct manipulation system is not possible. So it was necessary to define a language for the conception of such physical networks using logic operations, loops and symbolic assemblage under graphico-textual modalities.

### *Visualisation*

As a strong feature of the simulation approach, the animated image creation works in the opposite way of the classical approaches. In the last, the visual properties of the objects (shape, texture, color, etc.) are first defined. Then, they are animated. In the MIMESIS (simulation) way, the movement (animation) is first, and then, the objects (a set of consistently moving particles) must be "clothed" with visual attributes. This visualization can be done through several different ways that are, in so speaking, a modeling of "how photons meet matter".

Annie Luciani and Arash Habibi [HABIBI (A), LUCIANI (A), 1993, 1997] developed a physical modeling based concept for visualization, the "engraved screen" inspired from the technique invented, in the traditional animated images domain, by Alexeieff.

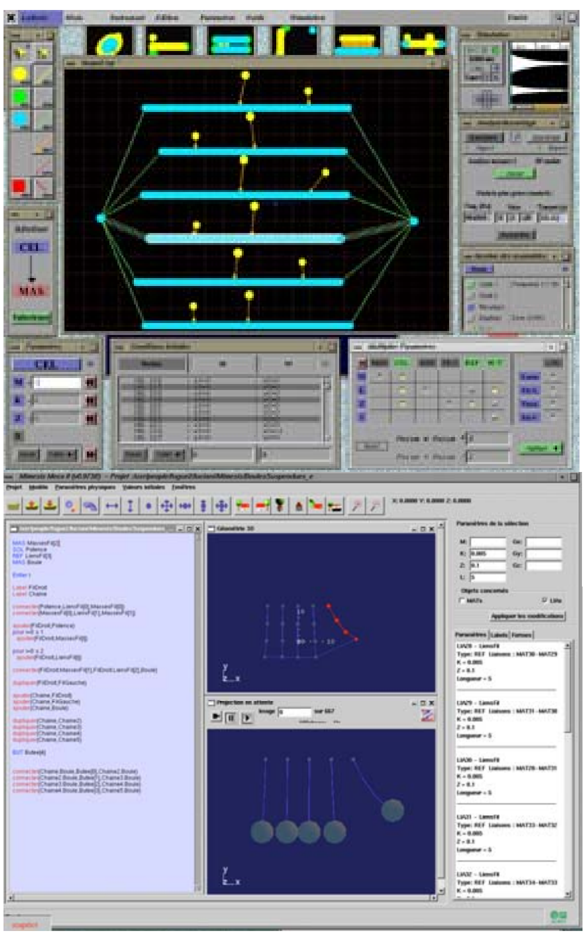


**Fig.5 - The GENESIS (top) and MIMESIS workbenchs (bottom)**

GENESIS and MIMESIS are used today in several cultural institutions  and have been used also for numerous creations in music and image (see below). Claude Cadoz created is musical piece "*pico..TERA*" (2001) which uses exclusively GENESIS, and Annie Luciani her film "*Mémoires Vives*" (2001) that uses also exclusively the simulation technique.

## 6. The future platform

Sight and hearing are two complementary ways of apprehension of the objects and phenomenon in our environment. They inform us on different features from a same object or event. Some objects are better perceived by sight, other by hearing. So, it is not surprising that the creation of acoustic and the creation of visual phenomenon refer to different approaches. So, it was legitimate to dedicate tools for each domain. But the simulation paradigm is, precisely, above the perception modalities. Focusing on the causes rather than on their effects open the way for a deep synthesis allowing not only to superficially "paste" together the audio and the visual media, but to integrate a multisensory consistence in the "model" itself.

In an other hand, several years of development and practice within GENESIS and MIMESIS reveal two facts: numerous of their functionality are de facto formally identical (even if they are sometimes implemented with some superficial differences), and it is no rare that, using one or the other, the artists from one category develop obviously implicit thinking of the other. This incites strongly to actualize this synthesis in a unique tool.

The computing power needed by real-time multisensory simulation is today reachable, at least for models of reasonable complexity, and the CORDIS-ANIMA simulation process are now stable enough to be implemented in any platform.

If we add the fact that gesture devices (TGR) are now compact enough, with high-performance, to be integrate nearby the mouse on usual desk workstations, we can clearly conceive the future platform: a real-time simulator connected to a general physical modeler and to panoply of TGR.

Assuming (through a deep and carefully studied definition of the standards) the complete transparency and compatibility with general environments for computer artistic creation, and thanks to a relevant analyze of what can be transmit and what must be resident on a site, such platforms can be the nodes of a wide user network.

## 7. Artistic, scientific, research, creation and applications

In this last part, we expose and summarize, in a global survey, several significant steps in the research and application of these concepts and techniques, as well in the internal activities during or in parallel of the designing of the tools, as in collaborative activities in several fields: artistic creation, education and pedagogy, scientific and industrial applications.

<u>1979:</u>

- First TGR (J-L. Florens) and first experiments of gestural interaction with virtual objects (rigid obstacles), with an analogic calculator, and then with DEC-LSI11.
- First simulation of vibrating structures (C. Cadoz).

<u>1980-83:</u>

- First CORDIS-ANIMA simulator (C. Cadoz) on DEC-LSI11, first non real-time simulation of strings, and also the "falling ball", a simple model of a ball falling on the ground and producing very realistic sounds.

- First simulation of simple deformable objects manipulated in real-time and visualized on oscilloscope screen (A. Luciani).
- The "Touche" (C. Cadoz and J-L. Florens), simulation of various matter tactile rendering.
- The CTR (Cordis Temps-Réel), first physical modeling real-time procesor (T. Berberyan).
- Thanks to the CTR (specialized hardware for CORDIS-ANIMA simulation in real-time), simulation of plucked strings manipulated with the "Touche" and modulated (in elasticity) with a position sensor.
- Multisensory (eye-hand) experiences with simple visualization on oscilloscope screen.

1983-87:

- Implementation of real-time simulator on Floating-Point System AP120 (J-L. Florens).
- Implementation of the first modeler (A. Luciani, A. Razafindrakoto).
- Design of a vectorial real-time visualization screen (B. Merlier).
- Model of a marionette simulated and manipulated (by its feet) in real-time.
- First complete multisensory real-time simulation: a tennis ball (deformable object), played with a racket moving horizontally and manipulated with the "Touche", striking walls with acoustical vibrating properties.
- First models of bow-string interaction within the CORDIS-ANIMA modeling (C. Cadoz).
- Design of the "Clavier Rétroactif Modulaire" (CRM®) and its slice-motor technology (C. Cadoz, L. Lisowski, J-L. Florens)

1987-93:

- Thanks to the CRM and the AP120, first real-time simulation of a violin bowed string (J-L. Florens): the bow, handily manipulated through a joystick mounted on the CRM could be controlled on pressure and velocity. The bow-string interaction included the pressure parameter. A third key from the CRM allowed modifying the elasticity of the string, and then its pitch.
- With the same equipment, model of maracas with a set of particles in a box, manipulated in real-time with the CRM (C. Cadoz).
- Series of models of structured deformable objects for image animation (S. Jimenez, A. Luciani):
  - second model of marionette with force-feedback,
  - flag, football ball, trampoline, basketball basket, bicycle, complex vehicle, etc.
- First models (animated images) of non-structured objects: large set of particles in elastic interaction, producing waterfall effect, study of plasticity, etc. (S. Jimenez).
- Second model of complete real-time multisensory simulation, the "Granule" (J-L. Florens). This model was decisive in regard of the "presence" sensation: several small objects are enclosed in a circular 2D box manipulated by CRM. Thanks to a very precise modeling of the collisions between the objects and the container, even with very simple sounds and visual representations, but thanks to the deep consistence of the physical phenomenon and their relation to manipulation, we obtain a genuine effect of real existence ("presence").
- Development of the first gesture editor (C. Cadoz, Ch. Ramstein).
- Premises of GENESIS.

1993-2000:

- Implementation of simulator and all environments on Silicon Graphics Workstations, replacing the AP120.
- Models of complex structured objects and complex interactions: complex vehicle interacting with a soil (B. Chanclou). In the context of contracts with CEA for the studies of a VAP (Autonomous Planetary Vehicle), a complete 6 wheel articulated vehicle has been modeled, taking account the deformations of the wheels, their interaction with the soil, the deformation of the soil. This vehicle included also a physical model of motor generating its movements.
- In the same context, models of motricity have been studied and implemented in simple models of frog, snakes, etc.
- At the same time, B. Chanclou implemented the first physical models to realize non-physical operations: task planning for robotic application.
- Experiments on psycho-physics: using of multisensory simulation and gesture control in complex manipulation with several collaborating people on a same task (A. Bouzouita, C. Uhl).
- Models of collective behavior within large ensemble of particles:
  - models of sand, pastes, smoke, fluid (A. Luciani, Z. Junedi, A. Vapillon),
  - models of turbulence and chaotic phenomenon, autosimilarity,
  - application to the human scale of crowd behavior (A. Luciani, N. Tixier),

From these steps, the modeling activity in animated image made a breakthrough where the artistic (or esthetic) finality become significantly a guideline for modeling.

- Starting of the general study of the form/movement relation as a central paradigm in animated images with physical modeling (A. Luciani).
- Development of the "engraved screen" principle for visualization of physical model by physical models (A. Habibi, A. Luciani).
- Models of large scale atmospheric phenomenon: aurora borealis (E. Juliax, A. Luciani).
- First piece (image and music) created with the technique of ACROE, by ACROE: *ESQUISSES* - 1993. (ACROE).
- First versions of GENESIS and MIMESIS (1995, C. Cadoz, O. Corbun, A. Luciani, A. Habibi).
- First international Workshops with GENESIS and MIMESIS (1996).
- Starting of the R_APM project (Network of Mobile Pedagogic Workshop) in the Rhône-Alpes Region, and of teaching of artistic creation with computer by ACROE in several institutions (conservatory and school of art in Grenoble, Turin, Karlsurhe,…).
- Residences of artists in ACROE, creation of several musical and/or visual pieces by tens of artists, organization of public events (concerts, expositions, conferences).

2000-2003:

- Development of high level versions of GENESIS and MIMESIS and of their didactic environments (N. Castagné, C. Cadoz, A. Luciani, P. Fourcade).
- New TGR improving the performance of the CRM technology, and development of a complete panoply of "habillages": keyboard, 2D, 3D, 6D joysticks, etc. (J-L. Florens, G. Brocard).
- Development of the TELLURIS simulator on powerful multi-processor platforms (J-L. Florens, D. Muniz, Y. Chara).

- Development of fundamental experiments in haptic perception and multisensory interaction (A. Luciani, P. Ostorero, T. Olhmann).
- Development of scientific and technologic applications of the IMS and TGR:
  - introduction of the TGR in car industry (J-L. Florens, A. Luciani)
  - development of a "nanomanipulator with force-feedback" in collaboration with the LEPES laboratory (CNRS-Grenoble) (J-L. Florens, A. Luciani, C. Cadoz, J. Chevrier, S. Marlière, D. Urma).
- Development of new models: a new version of the bowed-string has been developed by J-L; Florens (invited conference in the Forum Acousticum 2002) where two strings can be played in a 2D space allowing to apply the bow on a realistic way on the both, or on one of them, with a very accurate gesture.
- First musical piece created by C. Cadoz ("pico..TERA") illustrating the principle of creation at the compositional structure level by the physical model paradigm.
- "Mémoires-Vives", film from A. Luciani.
- Development of wide structured libraries of models for musical creation and of the basis of the composition by physical modeling (C. Cadoz).

[see Figures 5X, 5Y, 6 and 7 at the end]

## 7. The team

Jean-Claude Risset is President of ACROE, and Max Mathews President of Honor, Annie Luciani and Claude Cadoz manage ACROE-ICA, Jean-Loup Florens has general scientific and technical high responsibility, Maria Guglielmi is Secretary of ACROE, Guy Diard was our so kind multimedia engineer that left us so suddenly in 2001, Nicolas Castagné joints us as permanent member just after his PhD.

Numerous non-permanent members working during periods on the different axes complete this basic team. More than 20 Univeritary Thesis have been written on the ACROE-ICA program. About 300 trainee students have worked on different subjects during the 20 last years. The list of names of these numerous contributors of the results we can show today is too long to be placed there. But we address them warm thanks.

## 8. Acknowledgements

The works of ACROE-ICA are supported by:

- French Ministry of Culture and Communication
- French Ministry of Research and Technology
- Institut National Polytechnique de Grenoble
- Rhône-Alpes Regional Council
- General Council of Isère
- Culture dept. of Grenoble

These supports are completed by specific contracts with:

- Ministry of National Education, Research and Technology.
- Renault-United Parts France.
- CNRS (Force-feedback Nanomanipulateur Project), etc.

## Bibliography ACROE-ICA

### *Force-feedback Transducers*

FLORENS (JL), 1978 - "Coupleur gestuel interactif pour la commande et le contrôle de sons synthétisés en temps réel" - Thèse Docteur Ingénieur - Spécialité Electronique - I.N.P.G. - Grenoble 1978.

CADOZ (C), LUCIANI (A), FLORENS (JL), 1984 - "Responsive Input Devices and Sound Synthesis by Simulation of Instrumental Mechanisms : The Cordis System" - Computer Music Journal, 8, N°3, pp. 60-73. M.I.T. Press, Cambridge Mass. 1984.

CADOZ (C), LUCIANI (A), FLORENS (JL), 1989 - "Responsive Input Devices and Sound Synthesis by Simulation of Instrumental Mechanisms : The Cordis System" - In *Music Machine* Curtis Roads Ed. MIT Press, Cambridge Mass. 1989.

LISOWSKI (L), CADOZ (C), 1989 - "Conception, optimization and realisation of extra flat DC linear motors" - Proc. of 4th International Conference on Electrical Machines and Drives - IEEE - Sept 1989.

CADOZ (C), LISOWSKI (L), FLORENS (JL), 1989 - "Clavier rétroactif modulaire et actionneur modulaire plat" - Brevet Français déposé le 13 / 10 / 88. n°88 14064, US n° 07/420 242 - Europe n° 0 365 441 - Japon, 1989/11/10 - n° 265299 - Titulaire : ACROE.

CADOZ (C), LISOWSKI (L), FLORENS (JL) - "Modular Feedback Keyboard" - International Computer Music Conference - Glasgow 1990.

CADOZ (C), LISOWSKI (L), FLORENS (JL), 1990 - "A modular Feedback Keyboard design" - Computer Music Journal, 14, N°2, pp. 47-5. M.I.T. Press, Cambridge Mass. 1990.

LUCIANI (A), CADOZ (C), FLORENS (JL), 1994 - "The CRM device : a force feedback gestural transducer to real-time computer animation" - Displays , Vol. 15 Number 3 - 1994 - Butterworth-Heinemann, Oxford OX2 8DP UK, pp. 149-155.

NOUIRI (J), FLORENS (JL), CADOZ (C), 1994 - "Actionneurs modulaires plats à flux réparti" - Colloque EPE Chapter Symposium, Lausanne 17-21 Oct 1994 pp. 331-334.

NOUIRI (J), 1995 - "Etude pour la conception de transducteurs gestuels à retour d'effort et de modèles de systèmes mécaniques pour leur contrôle" - Thèse de l'INPG, Spécialité Signal-Image-Parole, Grenoble Nov. 1995.

### *CORDIS-ANIMA language*

CADOZ (C), "Synthèse sonore par simulation de mécanismes vibratoires", Thèse de Docteur Ingénieur, Spécialité Electronique - INPG. - Grenoble 1979.

CADOZ (C), LUCIANI (A), FLORENS (JL), "Synthèse musicale par simulation des mécanismes instrumentaux. Transducteurs Gestuels Rétroactifs pour l'étude du jeu instrumental", Revue d'Acoustique N° 59, pp. 279-292. Paris 1981.

CADOZ (C), LUCIANI (A), FLORENS (JL), "Synthèse sonore par modélisation mécanique de l'instrument. Le langage CORDIS", Onzième ICA, Paris 1983.

LUCIANI (A.), CADOZ (C.), "Modélisation et animation gestuelle d'objets - Le système ANIMA", CESTA - 1er Colloque Image, Biarritz 1984.

CADOZ (C), LUCIANI (A), FLORENS (JL), "Gesture, Instrument and Musical Creation. The CORDIS-AQNIMA system", Preprint n° 2086, 75th AES Convention Paris 1984.

CADOZ (C), LUCIANI (A), FLORENS (JL), "Responsive Input Devices and Sound Synthesis by Simulation of Instrumental Mechanisms : The Cordis System", Computer Music Journal, 8, N°3, pp. 60-73. M.I.T. Press, Cambridge Mass 1984.

CADOZ (C), LUCIANI (A) - "Processes, Sound Synthesis Models and the Computer Designed for Musical Creation", ICMC - Paris IRCAM.1984.

LUCIANI (A), "Un Outil Informatique de Création d'Images Animées : Modèles d'objets, Langage, Contrôle Gestuel en temps réel. Le Système ANIMA", Thèse de Docteur Ingénieur - INPG. Grenoble 1985.
RAZAFINDRAKOTO (A), "Langage d'objets pour l'animation, implantation temps réel", Thèse d'Université - Grenoble 1986.
CADOZ (C), LUCIANI (A), FLORENS (JL), "Responsive Input Devices and Sound Synthesis by Simulation of Instrumental Mechanisms : The Cordis System", In *Music Machine* Curtis Roads Ed. MIT Press, Cambridge Mass 1989.
FLORENS (JL), CADOZ (C), "Modular Modelisation and Simulation of the Instrument", International Computer Music Conference - Glasgow 1990.
LUCIANI (A), "Modèles Physiques et Animation", Imagina 90 - Monte-Carlo 1990.
FLORENS (C), CADOZ (C), "The physical Model, Modelisation and Simulation Systems of the Instrumental Universe", In *Representations of Musical Signals*. G. De Poli, A. Picciali, C. Roads, Ed. MIT Press, 1991.
CADOZ (C), "Modèles physiques, temps réel et contrôle gestuel avec retour d'effort dans la synthèse sonore pour la création musicale", Imagina '91 - Monte-Carlo 1991.
CADOZ (C), LUCIANI (A), FLORENS (JL), "CORDIS-ANIMA : a Modeling and Simulation System for Sound and Image Synthesis- The General Formalism", Computer Music Journal, Vol 17-1 spring 1993, MIT Press.
HABIBI (A), LUCIANI (A) - " From the physical model to the eye", 4th Eurographics Workshop on Animation and Simulation, Ed. A. Luciani, D. Thalmann - Barcelone - Espagna - September 1993.
CADOZ (C), LUCIANI (A), FLORENS (JL), "CORDIS-ANIMA : a Modeling and Simulation System for Sound and Image Synthesis- The General Formalism", CMJ Vol 17-1 spring 1993.
CADOZ (C), LUCIANI (A), FLORENS (JL), "CORDIS ANIMA : Système de modélisation et de simulation d'instruments et d'objets physiques pour la création musicale et l'image animée", Collection "Recherche Musique et Danse", Actes du colloque "Modèle physique, création musicale et ordinateur", organisé par l'ACROE Grenoble 1990, Ed. Maison des Sciences de l'Homme, Paris 1994.
LUCIANI (A), "Informatique graphique", numéro spécial de TSI, Vol 14, n°10, Hermes Ed., Paris 1995.

## *Real-Time Simulators*

DARS-BERBERYAN (T), "Etude et réalisation d'un calculateur spécialisé pour la synthèse sonore en temps réel par simulation de mécanismes instrumentaux", Thèse l'Institut National Polytechnique de Grenoble 1982.
DARS-BERBERYAN (T), CADOZ (C) & FLORENS (JL), "Processeur spécialisé pour le calcul de sons en temps réel par simulation instrumentale", Onzième ICA, Paris 1983.
FLORENS (JL), LUCIANI (A), CADOZ (C), "Optimized real-time simulation of objects for musical synthesis and animated images synthesisInternational", ICMC La Haye 1986.
UHL(C), LUCIANI(A), CADOZ (C), "Le simulateur temps réel pour le langage CORDIS-ANIMA. Modèles Physiques, Gestes, Sons, Images" - 3ième séminaire Animation et Simulation. Lille 1994.
UHL (C), "Architecture de machines pour la simulation d'objets physiques en temps réel", Thèse de l'Institut National Polytechnique de Grenoble 1996.
FLORENS (JL), LUCIANI (A), CADOZ (C), "A real-time workstation for physical model of multi-sensorial and gesturally controlled instrument", ICMC 1998.
GIRAUD (O), "Architecture logicielle haute performance pour la simulation temps-réel synchrone d'objets physiques multisensoriels : retour d'effort, synthèse de sons, synthèse d'images", Thèse de l'Institut National Polytechnique de Grenoble1999.

## *Instrumental Interaction*

CADOZ (C), LUCIANI (A), FLORENS (JL), "Gesture, Instrument and Musical Creation. The CORDIS-ANIMA system", Preprint n° 2086, 75th AES Convention Paris 1984
GIBET (S), "Codage, représentation et traitement du geste instrumental", Thèse de Docteur de l'Institut National Polytechnique de Grenoble - Spécialité Electronique, - Grenoble 1987
CADOZ (C.), "Instrumental Gesture and Musical Composition", International Computer Music Conference - Cologne 1988.
GIBET (S) & FLORENS (JL), "Instrumental gesture modeling by identification with time varying mechanical models", International Computer Music Conference - Cologne 1988.
CADOZ (C) & RAMSTEIN (C), "Capture, Representation and Composition of the Instrumental Gesture", International Computer Music Conference - Glasgow 1990.
RAMSTEIN (C), "Analyse, représentation et traitement du geste instrumental", Thèse pour obtenir le grade de Docteur de l'I.N.P.G., Spécialité Informatique, décembre 1991
CADOZ (C) "Interface de Communication Instrumentale - Interaction gestuelle", Conférence "L'Interface des Mondes Réels et Virtuels" - Montpellier - Mars 1992
CADOZ (C), "Geste Instrumental et Informatique Musicale" , 5ièmes Entretiens du Centre Jacques Cartier sur "La performance Musicale Avancée : une perspective convergente de l'Artiste et du Scientifique" - Montréal, 6 - 10 Octobre 1992
LUCIANI (A), "Ordinateur, Images et Mouvements. Apparition de l'instrumentalité dans l'art du mouvement visuel", Revue Les Cahiers de l'Ircam n° 2, novembre 1992
CADOZ.(C), "Le geste, canal de communication homme/machine. La communication instrumentale" - Technique et science informatiques . Volume 13 - n° 1/1994, pages 31-61
CADOZ (C) , "Relation instrumentale et Création musicale avec l'ordinateur" - Conférence Culture Technologie Convergence - Conférence de l'Ontario et des 4 moteurs de l'Europe - MacLuhan program in Culture and Technology - University de Toronto - Mai 1994
BOUZOUITA(A), CADOZ(C) - "Interaction Instrumentale Multisensorielle : Eléments pour l'étude des transformations de l'espace de l'opérateur et de l'espace de la tâche" Actes IHM'95, Toulouse 1995.
CADOZ (C), "Physical Modelling and Gestural Interaction in Computer Music" Workshop in "Currents trends of research on Acoustics of Musical Instruments, Musical and non Musical Applications" of the 98th AES Convention - Paris, 1995.
CADOZ (C), " Technologies and Concepts of the Instrumental Communication,", 2nd Franco-Japonese Conference on Synthetics Worlds, Paris, January 25-27, 1995, John Wiley & Sons Publ.
BOUZOUITA(A), UHL (C), LUCIANI (A), CADOZ(C) - "Manipulation complexe d'objets virtuels" - 5ième Journées Internationales "Interface des mondes réels et virtuels", AJIIMD Ed., Montpellier, 21-24 mai 1996, 1996, pp459-470
CADOZ (C), "Réintroduire les sensations physiques : des interfaces manuelles pour montrer, toucher, palper, éprouver", Numéro spécial "la Recherche" "l'ordinateur au doigt et à l'oeil" - n°285 Mars 1996

LUCIANI (A), "Ordinateur, gestes, images, sons : une nouvelle ère pour les systèmes dynamiques", Conférence "Voir, entendre, calculer, raisonner", CNRS SPI & SPM, Paris, la Villette, 10-13 juin 1997

BOUZOUITA (A) "Contribution à l'étude de la perception du retour d'effort dans la commuication Homme - Machine" - Thèse de l'Université Joseph Fourier de Grenoble , Spécialité Signal - Image - Parole, 1997

CADOZ (C), "Musique, gestes, technologies". in "Les nouveaux gestes de la musique". sous la direction de Hugues Genevois et Raphaël de Vivo. Editions Parenthèses, Marseille1999.

CADOZ (C), WANDERLEY (M), "Gesture and Music" in Trends in Gestural Control of Music. Editeur M. Wanderley et M. Battier - IRCAM, Paris 2000, avec CDROM.

CADOZ (C), "Continuum énergétique du geste au son : simulation multisensorielles d'objets physiques". in "les interfaces pour la création musicale". HERMES Editeur. 1999.

## *Artistic creation Interfaces*

HABIBI (A) "Modèles physiques pour la visualisation d'objets très déformables : Relation, mouvement - forme - image" , Thèse de l'Institut National Polytechnique de Grenoble, Spécialité Informatique, Grenoble 1997.

CASTAGNE (N), CADOZ (C) , "L'environnement GENESIS pour la création musicale à l'aide des modèles physiques particulaires" - Journées d'Informatique Musicale – Marseille 2002.

CASTAGNE (N), CADOZ (C) , "GENESIS: A Friendly Musician-Oriented Environment for Mass-Interaction Physical Modeling" - ICMC – Goteborg 2002.

CADOZ (C), "Le Modèle Physique, métaphore pour la création musicale" - Journées d'Informatique Musicale – Marseille 2002.

CADOZ (C), "The Physical Model as Metaphor for Musical Creation *"pico..TERA"*, a piece entirely generated by physical model" - ICMC – Goteborg 2002.

Voir aussi www-acroe.imag.fr

## *General recent publications*

HABIBI (A), LUCIANI (A). "Dynamic Particle Coating", IEEE Transactions on Visualisation and Computer Graphics. A paraître.

TIXIER (N), "Street listening", in Helmi Järviluoma (dir.) *Soundscape Studies and Methods*, Université de Turku, Finlande, 2001, sous presse.

DJOHARIAN (P), "Material Design in Physical Modelling Sound Synthesis". Proceedings of DAFx. Trondheim, Norway. Dec. 1999.

FOURCADE (P), MANGIAROTTI (S), CADOZ (C), "Generating pitches in transients by a percussive excitation. Proceedings of ICMC. Beijing. China. Oct. 1999.

FOURCADE (P), CADOZ (C) " Generating /p/ /t/ /k/ Consonants by a physical Modelling of Musical Percussions". The 8th International Workshop on Cognitive Science of natural Language. Galway. Ireland. August 1999.

FLORENS (J.L), GERMOND (J), "Reed Instruments Modular Representation and their Interactive Real Time Simulation". Proceedings of ICMC. Beijing. China. Oct. 1999.

CADOZ (C), GUBIAN (S), "From Song of Birds to extended virtual reed instruments : Musical modelling of bird vocal tracks and application to a new kind of sound sustain virtual instruments". Proceedings of ICMC. Beijing. China. Oct. 1999.

CASTAGNE (N), CADOZ (C) "Physical Modeling : balance between realism and computing speed", DAFx 2000 - Italy. December 2000.

DJOHARIAN (P), "Shape and Material Design in Physical Modelling Sound Synthesis". Proceedings of ICMC. ICMA ed. Berlin. Sept 2000.

GUILBAUD (C), LUCIANI (A), LECLERC (A), "Volume construction from moving unorganized points". Proceedings of Graphicon Conference. Moscow. Sept. 2000.

LUCIANI (A), "From granular avalanches to fluid turbulences through oozing pastes : a mesoscopic physically-based particle model. Proceedings of Graphicon Conference. Moscow. Sept. 2000.

FOURCADE (P), CADOZ (C), "Physical Modeling for Sound Synthesis with Particle – Interaction Networks : Study of the Percussion Phenomena and Design of generic rules for Percussions", ICMC 2001. Cuba. Sept 2001.

TIXIER (N), "Toward a caracterisation of the sound environment : the method of *qualified listening in motion* ", *Actes Inter.noise 2000*, Nice, août 2000.

TIXIER (N), "Street listening ", *Actes du colloque Sound Practice*, Devon, Angleterre, Février 2001.

FOURCADE (P), LUCIANI (A), TIXIER (N), "Modeling of sound ambiences : contributions of the physical model ", *Actes Inter.noise 2001*, La Haye, Pays-Bas, août 2001.

LUCIANI A., "Gravure virtuelle : de geste dans l'argile à l'ordinateur". in Nouvelles de l'Estampe. Edité par le Comité National de la Gravure Française. Numéro spécial "de l'estampe traditionnelle à l'estampe numérique". N°167-8. Décembre 1999/Janvier 2000.

## Other references

GIBSON (JJ), "The Ecological Approach to Visual Perception" – Houghton Mifflin ed. – Boston 1979.

RHEINGOLG (H), "Virtual Reality" - Summit Books, New York 1991.

KALAWSKY (R-S), "The Science of Virtual Reality and Virtual Environments", Addison-Wesley 1993.

QUEAU (P), "Le Virtuel - Vertus et Vertiges" - Ed. Champ Vallon, Seyssel 1993.

CADOZ (C), "Les réalités virtuelles" - Collection DOMINOS - Flammarion - Paris 1994.

KUNII (TL), LUCIANI (A), "Cyberworlds" - Springer-Verlag Tokyo Berlin Heidelberg New York 1998.

WANDERLEY M. "Interaction musicien – instrument : applications au contrôle gestuel de la synthèse sonore". Doctorat de musicologie. Université de Paris VI. Juin 2001.

## 9. Recent artistic creations

LUCIANI A. « Mémoires Vives », Images de synthèse et textes. Co-production ICA / ACROE. 2001

CADOZ C. « pico..TERA » - Sons de syntèse GENESIS- Co-production ICA / ACROE. 2001.

BRUEMMER L. « Thrill ». Sons de synthèse GENESIS. Co-production ZKM / ICA / ACROE. 2001

BRUEMMER L. « Medusa ». Sons de synthèse GENESIS et percussions. . Co-production ZKM / ICA / ACROE. 2001

MAIGUASHCA M. Tiefen. Pièce pour Sons de synthèse GENESIS et percussions. Co-production ZKM / ICA / ACROE. 2001

GAVAZZA G. « La natura delle cose ». Co-production AIAM / ICA / ACROE. 1999.

GAVAZZA G. « Variazioni su un Preludio di Skrjabin ». Pièce pour Disklavier et sons de synthèse GENESIS. Co-production AIAM / ICA / ACROE. 2001.

PETIT A. « Song from Waves », pièce pour Voix et Sons de synthèse GENESIS, Co-production CNR / ICA / ACROE. 2001

STUBBE H.P. « Rhizome » Harpe et sons de synthèse GENESIS. Co-production DIEM / ICA / ACROE. 2001.

DADACHOVA V. «Eks ». Pièce pour Orchestre et sons de synthèse GENESIS. Co-production CNR / ICA / ACROE. 2001

DOUVITSAS P. « Three songs ». Pièce pour Percussion, voix, piano et sons de synthèse GENESIS. Co-production CNR / ICA / ACROE. 2001

ZBOROWSKI M. « Communication ». 5 tableaux. 2000.

ZBOROWSKI M. « Wittgenstein ». 11 tableaux. 2001.

LUCIANI A. Images pour la chorégraphie «Lettre Capitale» de Joel BORGES. Opéra Bastille 1998, et Nantes 1999

LUCIANI A. Images pour la chorégraphie « Epopée pour jamais de Joel BORGES. Villa Médicis. 2000

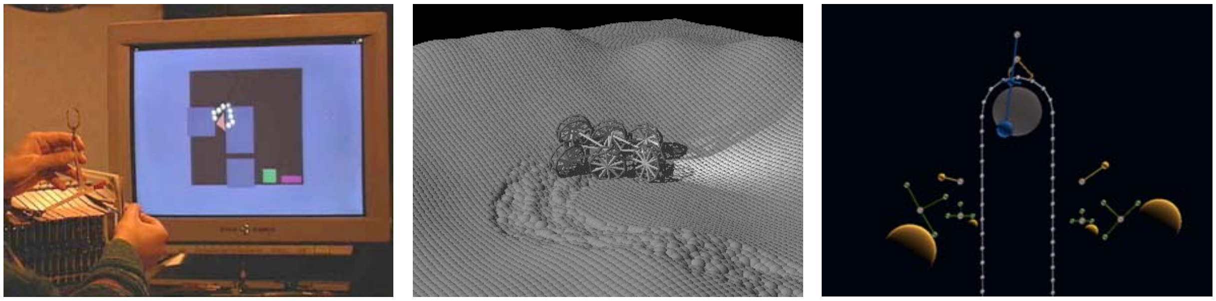


**Figure 5X**

**Figure 5Y**

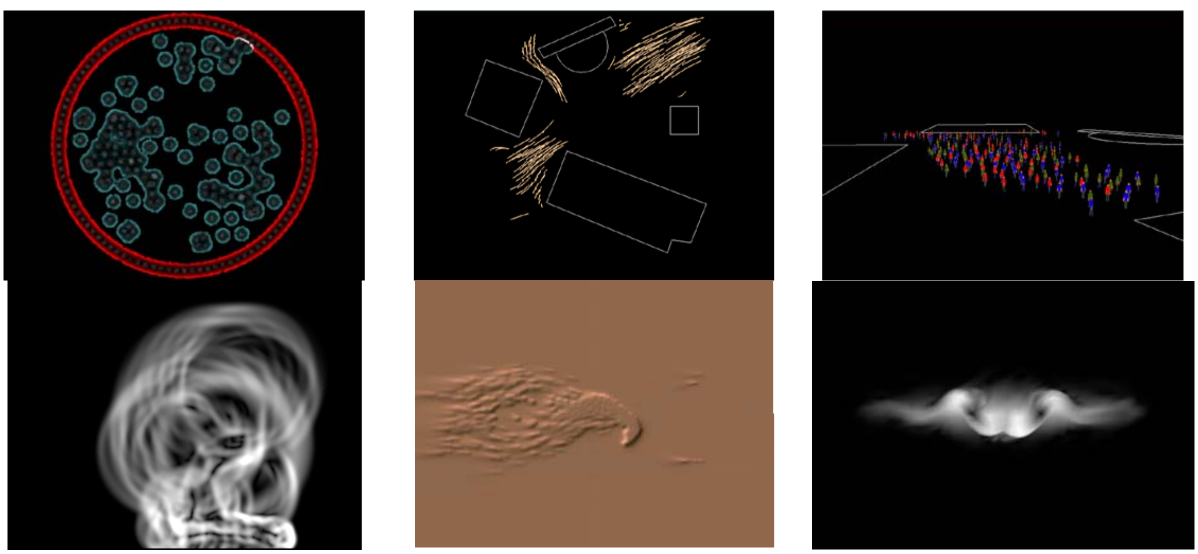


**Fig.6 - Images from "Mémoires Vives" - A. Luciani 2001**

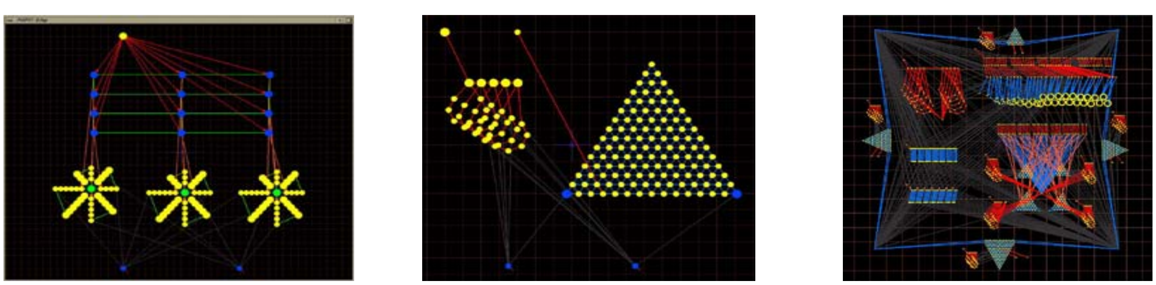


**Fig.7 - GENESIS models for "pico..TERA" - C. Cadoz 2001**

# Author Index