Constructing a T-Forest

Paul Buser

Music Technology Area
Department of Music Research
Schulich School of Music
McGill University
Montreal, Canada

August 2023

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Master of Arts.

2023/08/14

Abstract

Digital musical instruments (DMIs) offer new sonic and expressive possibilities, but their usage is often hampered by difficulty in setup and usage. To address these issues, I created a technical artifact called the "T-Tree" which is a device that attempts to mitigate these difficulties for a subset of DMIs known as "T-Sticks." The goals of the T-Tree are to (a) allow for easy setup and initial configuration of T-Sticks allowing for immediate if limited use, and (b) mitigate obsolescence of older T-Sticks. I created three T-Trees in the course of my research; two using an initial design, and a third improving on that design. The T-Trees were used in two public exhibitions: an art installation and concert. The T-Trees greatly simplify the setup and usage of T-Sticks, and aspects of their design can be applied to others working in DMI creation.

Résumé

Les instruments de musique numériques (IMN) offrent de nouvelles possibilités sonores et expressives, mais leur utilisation est souvent entravée par des difficultés de configuration et d'utilisation. Face à ces problèmes, j'ai créé un artefact technique appelé « T-Tree » : un dispositif qui tente d'alléger ces problèmes pour un sous-ensemble d'IMN connu sous le nom de « T-Sticks ». Les objectifs du T-Tree sont (a) de permettre une installation facile et une configuration initiale des T-Sticks qui permet une utilisation immédiate et peut-être limitée, et (b) d'atténuer l'obsolescence des anciens T-Sticks. J'ai créé trois T-Trees au cours de ma recherches; deux utilisant une conception initiale et une troisième améliorant cette conception. Les T-Trees ont été présentés dans deux expositions publiques : une installation artistique et un concert. Les T-Trees simplifient grandement la configuration et l'utilisation des T-Sticks, et certains aspects de leur conception peuvent être appliqués aux autres qui travaillent dans la création d'IMN.

Acknowledgments

I couldn't have done it alone!

Huge thanks to Marcelo Wanderley for guiding me through this process. Thank you to Linnea Kirby for taking a chance on a crazy idea with me. The musical jungle gym is still really cool. Thanks to Kasey Pocius for their incredible sound design work and helping me in countless ways during our time together at McGill. And thank you to Lucy Fandel for letting us use the T-Tree in The Windy Days! Shout out to John for working on Chime Time with me (for 2.0 we'll make it more modular)! Big thanks to Edu Meneses for his excellent work on the Puara project. Thank you Albert Niyonsenga for your tireless work on T-Sticks and for helping me build the T-Tree. Thanks to Meesh Fradkin for helping build the T-Tree. Special thanks to the folks at the Centre for Interdisciplinary Research in Music Media and Technology for hosting us in their concert series and for their generous travel award.

Hello, *eZone*! (You know who you are.)

Contents

List of Figures

List of Tables

List of Acronyms

ABC	Abstract Base Class
AP	Access Point
CI	Continuous Integration
CIRMMT	Centre for Interdisciplinary Research in Music Media and Technology
DAC	Digital-to-Analog Converter
DMI	Digital Music Instrument
FSM	Finite State Machine
GIL	Global Interpreter Lock
GPIO	General-Purpose Input-Output
IDMIL	Input Devices and Music Interaction Laboratory
IMS	Interactive Musical System
IP	Internet Protocol
IPC	Inter-Process Communication
JSON	JavaScript Object Notation
JST	Japan Solderless Terminal
LED	Light-Emitting Diode
LiPo	Lithium Polymer
MIDI	Musical Instrument Digital Interface
MMR	Multimedia Room
NIME	New Interface(s) for Musical Expression
OS	Operating System
OSC	OpenSoundControl
OSI	Open Systems Interconnection
PSK	Pre-Shared Key
PWM	Pulse-Width Modulation
SBC	Single-Board Computer

SLIP	Serial Line Internet Protocol
SPI	Serial Peripheral Interface
SPIFFS	SPI Flash File Storage
SSID	Service Set Identifier
USB	Universal Serial Bus
UWB	Ultra-Wideband
WFS	Wavefield Synthesis

Chapter 1

Introduction

This chapter introduces digital musical instruments (DMIs) and their affordances,[1] possibilities, and challenges. It establishes the goals of this thesis, documents the decisions that lead to those goals, and introduces the titular interface, the "T-Tree."

1.1 Digital Musical Instruments and Usability

DMIs open up myriad possibilities to make music using computing technologies [3], [4]. Though potentially innovative, widespread use of DMIs is hampered by several factors. Among them, (1) DMIs typically rely on general-purpose computing devices whose hardware and software become rapidly outdated [5]; (2) DMIs of various versions do not always share features, and may depend on specialized software for mapping and sound synthesis [6]; (3) the technical protocols and standards upon which DMIs are built will change over time and may fall into obsolescence [7]; (4) DMIs can be difficult to set up, often falling into disuse when their authors can no longer be contacted [8]. Figuring out how older instruments have been built can be a daunting task, as demonstrated by Torre and colleagues' substantial amount of work that went into discovering details and documentation about Michel Waisvisz's "The Hands" [9]. There is a clear need to create strategies to overcome these limitations if one wants to create DMIs that will be used over time [10], [11].

[1] First proposed in [1] and later applied to technology in [2].

1.1.1 Goals of this Thesis

This thesis documents the creation and iterative design of a technical artifact called the "T-Tree" designed to overcome these four factors in the specific context of a gestural controller known as the "T-Stick." I document the process of constructing three T-Trees, explain their construction process, and present their usefulness in the context of making the T-Stick a more robust, usable, and sustainable instrument.

The goals of the T-Tree are (a) to facilitate easy setup and initial configuration of T-Sticks allowing for immediate, even if limited, use; and (b) mitigate obsolescence of older T-Sticks.

1.2 Issues in DMIs

This section examines the four issues mentioned above in detail.

1.2.1 Outdated General-Purpose Computing Devices

Although general-purpose computing devices are readily available, highly versatile, and customizable to specific tasks (including interactive music-making with DMIs), they also introduce additional complexity, security risks, and update difficulties. For example, updating a general-purpose computing device may cause audio computing software to stop working, but failing to update it can leave it vulnerable to malware. As any general-purpose computing devices, it becomes slower and less secure [12].

1.2.2 Multiplicity of Versions

DMIs are rarely, if ever, static artifacts. They change over time according to the changing needs of the community in which they are used, which includes musicians, composers, engineers, and businesspeople [13]. Although this can be an advantage over static instruments—as it allows DMIs to suit the needs of their users—as DMIs change they may no longer be supported by their creators and fall into disuse. This is especially the case if they rely on specific software, as newer devices

may not share the same sensors or software as previous versions.

1.2.3 Changing Standards

DMIs are technical artifacts and depend on technical means to achieve their performance. Technological standards change. For example, although the Musical Instrument Digital Interface (MIDI) standard set out in 1983 is still in use today, it has a number of limitations that make it unsuitable for many kinds of DMIs [14]. OpenSoundControl (OSC), which is more suitable for a number of DMIs, is more flexible but must operate at a higher level in the open systems interconnection (OSI) stack [15], making it more complex to implement. Bespoke protocols can be tailor-made for the instrument they power. But without communal support or common practice these devices often fall into obscurity, for example, Waisvisz's "The Hands" [9]. These protocols are also most likely to change, as the smaller an audience for a protocol, the fewer stakeholders there are to consider when changing that protocol.

1.2.4 Difficulty in Setup

The longevity of an instrument is tied to a number of factors, among them ease of setup. For example, a survey of in *NIME*[2] conferences between 2010-2014 found that 47.1% of DMIs created by respondents would not be able to be played in performance without at least "a few hours" of work [8]. Such results highlight the need for methods that ease DMI setup to limit instrument obsolescence.

1.3 Reasons for Creating the T-Tree

This section outlines the *raison d'être* of the T-Tree, and documents the value it brings to the DMI scene.

Ultimately, my goal was to create a device that could embed multiple T-Sticks, cross-modulate their signals, and present them in an appealing way to people who had never played a DMI before

[2] New Interfaces for Musical Expression

and had no intention of becoming experts at doing so.

1.3.1 Reuse of Existing Interfaces

Why bother creating another digital music interface? There are undoubtedly many ways to make compelling DMIs, but the T-Tree offers some unique opportunities. For one thing, the T-Stick as a digital gestural controller is one of the oldest in continuous use today. Designing a new DMI has its benefits, but creating something that interfaces with and preserves an existing DMI is an under-utilized path: "The current scene is peppered with unique and fascinating digital instruments with a performer base of one" [10]. In my limited time as a Master's student at McGill's Input Devices and Music Interaction Laboratory (IDMIL), I decided to maximize my impact by creating a technical artifact that would augment an existing instrument in the lab rather than create something completely new. Creating new DMIs is of course valuable, but there is also value to be had in iterating on the design of existing instruments, expanding the community around them, and reducing the rate at which they become obsolete.

1.3.2 Adaptation and Improvement

Second, why create something that interfaces with the T-Stick, rather than changing or improving the design of the T-Stick itself? The design of the T-Stick can and will be improved in the future, as it already has in the past. At the time of writing, there is already a plan in IDMIL to update the T-Sticks internals with a custom printed circuit board (PCB) instead of loose wires and conductive ink instead of directly-soldered capacitive touch sensing.[3] However, there are still many older T-Sticks that don't use contemporary microcontrollers and make it difficult to flash their firmware. We are confronted by "variations in the choice of microcontrollers and sensors" [6], as well as variations in design and software.

Rather than rely on changing the existing instrument, the T-Tree functions as an adapter to convert the various signals sent by T-Sticks into a uniform standard that can be easily patched

[3] Both welcome improvements.

for instant music creation. In this way, we can avoid retrofitting existing T-Sticks, a time- and resource-consuming process, and instead solve the problem once in a way that is adaptable to all T-Sticks.

1.4 A Note on Terminology

There have been two major versions of the T-Tree: 1.0 and 2.0. Whenever I refer to "the T-Tree," without a specific label of 1.0 or 2.0, this refers to qualities that apply to both versions.

1.5 Structure of this Thesis

This thesis has six chapters. Chapter 1, which you are currently reading, introduces the T-Tree and the main concepts and problems to be solved in the space of DMI usability. Chapter 2 outlines the origins of the T-Tree and the T-Stick. It also documents previous work similar to the T-Tree. Chapter 3 sets forth the design goals of the T-Tree. Chapter 4 describes the technical means of implementing the T-Tree and improvements made from the T-Tree 1.0 to the T-Tree 2.0. Chapter 5 discusses the T-Tree's usage in performance and evaluates it in the context of the criteria set forth here and in previously published works. Chapter 6 summarizes the impacts of the T-Tree and suggests future work for the project.

1.5.1 Supplementary Material

Source code for the T-Tree can be found at `https://github.com/IDMIL/T-Tree`. A video overview and performance footage of the T-Tree can be found at `https://github.com/IDMIL/T-Tree/wiki/Videos`.

1.6 Intellectual Property Notice

Portions of this thesis have been adapted from a New Interfaces for Musical Expression (NIME) 2022 paper called "Introducing the t-Tree: Using Multiple t-Sticks for Performance and Installa-

tion" [16], for which I was the second author and Linnea Kirby was first author. The adapted section in Chapter 3 was originally written in collaboration with Kirby and appears with her permission.

Section 5.1 adapts sections of a NIME 2023 paper called "Towards the T-Tree 2.0: Lessons Learned From Performance With a Novel DMI and Instrument Hub") [17] for which I was the first author, Kasey Pocius was the second author, and Linnea Kirby was the third author. Pocius did the sound design for both public exhibitions of the T-Tree and the adaptation of this section appears with Pocius' permission.

Chapter 2

Background

This chapter documents the history of the T-Tree and its precursor, the T-Stick. It briefly outlines the original motivations and design goals of the T-Tree. It also documents three kinds of interfaces that both inspired and serve as a useful point of comparison to the T-Tree: sonic installations, interfaces for novices, and interfaces for collective improvisation.

2.1 The T-Stick

The T-Tree builds on the T-Stick. This section details what the T-Stick is, how it came about, its goals, and some difficulties of the T-Stick that have been mitigated with the creation of the T-Tree.

2.1.1 Origin of the T-Stick

The T-Stick is a family of gestural controllers originally designed by Joe Malloch in 2007 [18]. Each T-Stick is a sensor-rich device containing capacitive touch sensing, a force-sensitive resistor (FSR), an accelerometer or an inertial measurement unit (IMU),[1] and in older T-Sticks, a piezoelectric sensor.

[1] Which contains an accelerometer, gyroscope, and magnetomer.

The T-Stick is a "family" of gestural controllers designed to accommodate different sizes of the instrument with the same set of signals. There have been five different sizes of T-Stick created; from smallest to largest: sopranino, soprano, alto, tenor, and bass.

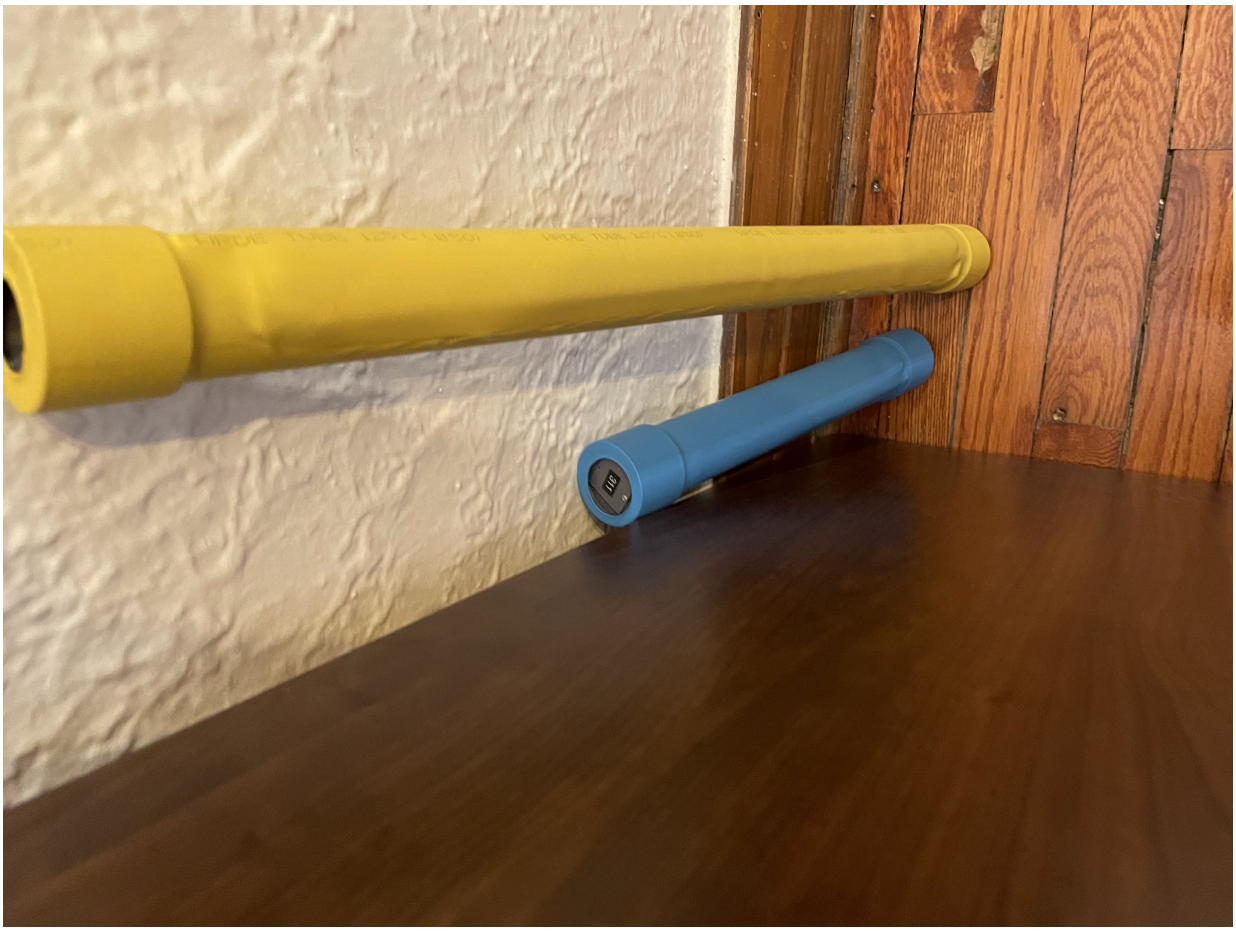

Fig. 2.1 A sopranino and soprano T-Stick.

The T-Stick has been the subject of numerous studies. It has been used as the focal point to investigate the relationship between DMI composition and a shared vocabulary of performer gestures [19]. It has been used as part of a study of composing new works for DMI in the context of the McGill Digital Orchestra [10]. It has been integrated into other DMIs, such as the TorqueTuner [20]. A 2018 study examined the work involved in maintaining T-Sticks and overcoming the

differences in "variations in the choice of microcontrollers and sensors" [6]. Finally, the "T-Stick Music Creation Project", proposed in 2021, addresses the challenges of building a community around a DMI. The T-Stick Music Creation Project included "a series of musical commissions along with workshops, mentorship, and technical support, meant to foment composition and performance using the T-Stick and provide an opportunity to improve technical and pedagogical support for the instrument" [21].

2.1.2 Goals (and non-goals) of the T-Stick

The T-Stick was not originally designed to have a "low entry fee," as defined by Wessel and Wright: "'getting started' with a computer-based instrument should be relatively easy, but this early stage ease-of-use should not stand in the way of the continued development of musical expressivity" [22]. Instead, "the instrument requires training and practice to reach a significant level of virtuosity" [6]. Given its sensor-rich design, it does indeed take practice to harness the many available control signals of the instrument.

This high entry fee also extends to the setup of the instrument itself. The T-Stick, without any additional hardware or software, constitutes one part of the tripartite definition of a DMI: an input device (or devices), a synthesis device, and a mapping layer between them [23]. For a user of the T-Stick to begin playing the instrument—for their gestures on the instrument to produce sound—they must first create a synthesis device and a mapping layer that connects the T-Stick to it. Thus, the T-Stick functions as a DMI if and only if there is additional setup by the end-user in the form of patch creation and mapping. For more details on how the T-Tree mitigates this difficulty in setup, see Section 3.3.1.

The T-Stick offers few clues about how it is meant to be played. Its nondescript PVC enclosure wrapped in heat-shrink tubing does not have any markings. The original version of the T-Stick did have ridges (from capacitive touch sensing strips) that appeared to many users as frets which led to the expectation that the instrument would have a mapping of the spatial dimension along the instrument to discrete pitches. The design was later changed to cover up the "fretted" appearance

of the instrument [18].

2.1.3 Versions of the T-Stick

Early versions of the T-Stick sent signals via universal serial bus (USB) and Bluetooth [24]. Some T-Sticks used a bespoke serial protocol over USB, whereas others used a protocol known as Serial Line Internet Protocol (SLIP).

As of 2018, there were eight operational T-Sticks in the IDMIL that used a wired connection [6]. Nieva's Master's thesis explored the process of maintaining those instruments and documented the difficulties in homogenizing software and signals across instruments that had different sizes, sensors, microcontrollers, and firmware [25].

Since then, the latest version of the T-Stick sends signals via OSC and libmapper[2] over a wireless connection. For these recent T-Sticks the USB connection is used only for charging the built-in Lithium Polymer (LiPo) battery and sending basic debugging information. However, as documented in Section 4.2.2, this USB data connection can also be used to configure and exchange information with compatible T-Sticks.

2.1.4 Barriers to Usability in the T-Stick

For the purposes of this thesis, it is useful to distinguish the usability barriers in *setting up* the T-Stick from the usability barriers in *playing* the instrument. The latter constitutes an explicit design choice, as discussed in previous paragraphs. Any instrument, whether digital or not, may be difficult to learn (e.g., developing a trumpet embouchure) but this is not necessarily a design flaw of the instrument itself. However, the difficulty in getting the instrument into a playable state is a hindrance to the instrument's playability and longevity: "A long preparation time might hinder the will of the musician to pick up the DMI" [8].

The long and difficult preparation time of the T-Stick is the focus of this thesis and the issue that the T-Tree attempts to solve.

[2] `libmapper.org`

2.2 Introducing the T-Tree

To overcome limitations of the T-Stick I designed the T-Tree in the fall of 2021 with Linnea Kirby. This section briefly outlines its main design features. More in-depth documentation of the T-Tree can be found in chapters 3 and 4.

2.2.1 Initial Ideas

The T-Tree was originally conceived as a "bass" T-Stick that would have been over 2 meters tall. When we realized that the feasibility of this was low (mostly due to the need to hand-solder hundreds of capacitive touch sensing strips), we decided to make something that would interface with existing T-Sticks, rather than create a new DMI.

The T-Tree was motivated by the observation that there was an interest in playing the T-Stick, but the most difficult part of the instrument was the setup. This included setup for older T-Sticks (including wired T-Sticks), which often relied on protocols or software that had fallen out of date or become obsolete.

2.2.2 Design Features

We wanted to create a multimodal device that would function as a DMI in its own right, as an "interactive music system" (IMS) in the spirit of "Musicking with an interactive musical system" [26], and as a "hub" around which musicians and curious passersby could gather and have a shared musical experience.

The T-Tree is a technical artifact that fulfills these requirements. It can embed multiple T-Sticks, cross-modulate their signals, and present DMIs in an accessible and appealing way to laypeople with no prior knowledge of DMIs.

2.3 Previous Work and Relationship to T-Tree

This section details a number of previous works and interfaces that the T-Tree has drawn inspiration from. Although this is not an exhaustive list, I have tried to include works that serve as useful points of reference for the T-Tree.

2.3.1 Sonic Installations

The T-Tree could be considered an instance of a "sonic installation" as it incorporates elements of "sculpture, sound, temporal composition, spatial architecture and audience interactivity" [27]. In making T-Sticks easily playable together, the T-Tree embodies the "idea of a sociable listening experience that people could collectively create and share if they wished" [27]. The T-Tree fosters communal experience and allows for collective music creation by offering T-Sticks to be played without further setup.

One example of an installation from which the T-Tree draws inspiration is *Ground Me!* [28]. This was a site-specific installation that utilized the particular shape of the rooms on offer at the Sonic Arts Research Centre (SARC) at Queen's University Belfast. It used the properties of skin conductance to create a reactive (in the sense of [29]) installation where participants could wander through a room touching copper poles that triggered different sounds. The direct correlation between gesture and sound was made clear in *Ground Me!*, and although the T-Tree does not prescribe a particular mapping strategy between gesture and sound, a direct mapping can make participants aware of the consequences of their actions. For an example of this phenomenon in a public exhibition of the T-Tree, see Section 5.1.1.

There are a couple of differences between the the T-Tree and *Ground Me!*. One difference is that *Ground Me!* was created from scratch, using all-new materials, and did not incorporate any previous DMIs. Another is that the installation was specifically designed to accommodate the room in which it was housed, whereas the T-Tree is designed to be portable and able to be used in a variety of environments. These differences highlight a distinction in design philosophy

and context for the different projects. In designing the T-Tree, the desire for the system to be self-contained overrode the desire for tight integration within the space, although one does not necessarily preclude the other.

Another compelling sound installation created for a specific site is *GrainStick* [30]. This sound installation used a room at IRCAM in Paris with many loudspeakers using wavefield synthesis (WFS). The sounds were controlled by participants with specially-outfitted smartphones (or Wii Remotes). *GrainStick* used accelerometer and motion-tracking data as input to a synthesis engine that mixed percussive and ambient sounds.

GrainStick is similar to the T-Tree in a number of ways. First, both projects use held objects to transmit accelerometer data to a synthesis engine. In the case of the T-Tree 1.0, both it and *GrainStick* also require an external sound synthesis engine in order to function. The sensors onboard the handheld gesture device are also similar: the T-Stick has an IMU that measures acceleration in six axes as does the Nintendo Wii Remote. One difference is that *GrainStick* performs motion tracking to derive inclination and length of the line between two of the handheld gesture sensors as well as their azimuth.

The T-Tree does not rely on a specially-outfitted room like *GrainStick*. However, the T-Tree may be considered less "immersive," as the sound source is either the T-Tree itself (only with version 2.0) or an external speaker system. *GrainStick*, via WFS, allows participants to perceive sound as originating from a number of different origins.

Note that the T-Tree does not preclude having multichannel audio; it just does not provide the means to do so as part of its self-contained system. For an example of the T-Tree with a large multichannel speaker array, see Section 5.1.2.

One final point of comparison is *Sound Forest*, which was exhibited at the Swedish Museum of Performing Arts in Stockholm in 2017 [31]. This installation utilized interwoven force-sensitive thread and fiber optic cable, creating illuminated "strings" that participants could interact with. The movement of the strings was captured using infrared sensors and contact microphones. Auditory feedback was sent to users via three loudspeakers, one above and two below a vibrating

platform.

Both *Sound Forest* and the T-Tree have multimodal feedback in the form of lights and sound. *Sound Forest* has another dimension of feedback by vibrating a platform underneath participants, experienced as haptic feedback. In the future it is possible that T-Sticks will have haptic feedback, but that is beyond the scope of this thesis.

Similar to the aforementioned interfaces, *Sound Forest* requires the outfitting of a room to function, whereas the T-Tree does not require a particular space. Again, this can be considered a specific design choice that leads to tradeoffs between the affordances of the space and the portability of the installation.

2.3.2 Interfaces for Novices

As a result of the T-Tree's collaborative nature and ability to function as an installation, it can also be used as a musical interface for novices. In this section I document some previous installations for novices as a point of comparison between them and the T-Tree.

Interfaces for use by novice performers (people with no intention to become experts) have been extensively documented and studied. In 2003, authors documented no fewer than 18 such interfaces, categorizing them along a number of axes, including scale (number of simultaneous players), learning curve, and physical interface (sensors) [32]. The reader may refer to this work for a more comprehensive treatment of the subject of interfaces for novices, which is beyond the scope of this thesis.

The T-Tree was inspired in part by the principles of IMS design documented in [26]. The authors of the IMS principles also developed an "interface designed for nonexpert users for exploratory and experiential purposes" [26] called the "MTBox." The MTBox illustrates a number of principles of IMS design from which the original T-Tree 1.0 took inspiration. A further discussion of these three qualities and the T-Tree's identity as an IMS can be found in Section 3.4.3.

Some interfaces for novices focus specifically on children. For example, the "Toy Symphony." The Toy Symphony is a project that focuses on interfaces usable by professional musicians and

novices, in particular, children. Some examples of instruments developed for the Toy Symphony include "BeatBugs" and "Music Shapers" [33]. Both the Toy Symphony project and the T-Tree focus on interfaces that are immediately accessible to people without previous experience with DMIs. However, the Toy Symphony project also allows novices to *compose* as well as perform, which is not a goal of the T-Tree at this time. Furthermore, although children could be a potential audience for the T-Tree, it is not designed with them in mind specifically.

The Music Shapers of the Toy Symphony share a number of similarities with the T-Tree. The Music Shapers do not make any sound themselves, but send signals to a computer to be converted into sound like the T-Stick with the T-Tree. Both Music Shapers and T-Sticks feature squeezing as a primary gesture. And both Music Shapers and the T-Tree are accessible to novices by eschewing the precise timing, rhythm, and manual dexterity of traditional acoustic instruments. Music Shapers, like the T-Tree, have also been used in an installation context [34].

The *iltur* system, which was a further development of BeatBugs [35], could also be considered an IMS, though its creation predates that term in the sense of [26]. The *iltur* system facilitates collaborative musical experiences between players of similar skill levels and between expert and novice players. Although this is not a specific design goal of the T-Tree it is possible with new patching.

The *iltur* system shares a number of similarities with the T-Tree. Both systems utilize multimodal feedback (audio/visual/haptic and audio/visual). Users interact indirectly with the systems via a gestural control object (i.e., a BeatBug or T-Stick). Both the T-Tree and *iltur* "provide a supporting infrastructure for the interdependent connections among human players" [35]. And both *iltur* and the T-Tree both adapt an existing DMI in a new context.

2.3.3 Interfaces for Collective Improvisation

The T-Tree has drawn inspiration from numerous interfaces that facilitate collective improvisation.

One such project was the "Brain Opera." The Brain Opera project "was a first-of-its-kind musical experience that included contributions from both on-line participants and live audiences"

[36]. As reviewed by Jon Ippolito in 1996:

> "...we were told that sounds made earlier by visitors [to Brain Opera] playing hyperinstruments would crop up in the performance, and that remote players joining in from the World Wide Web could exert some control over the sounds heard during a short section of the opera. Perhaps the most important instruction in Machover's recipe was to blend the contents thoroughly. For unlike the abrupt transitions found in more discordant postmodern compositions, Machover's careful engineering of the overall sound mellowed the potential cacophony of all the contributors..." [37]

The design philosophy of the T-Tree follows along similar lines in that the sound design that has been used often takes input from multiple players and synthesizes that input into concordant soundscapes. The T-Tree does not allow participants on the Internet to participate in musicmaking, but this could be a potential future addition.

The Brain Opera also included instruments with a "tree" metaphor, namely, "The Singing Tree," "The Speaking Tree," and "The Rhythm Tree" [38]. The three different trees perform different functions, as described below:

In interacting with the Speaking Trees,

> "A floormat switch detects the user's presence, and starts an interactive experience, where the user engages in a dialog with Marvin Minsky, who asks the participant several questions (all responses are recorded for use in the Brain Opera performances). The user then navigates through the interactive database using a hand-held piezoresistive mouse that detects the center of pressure of the thumb and tracks the mouse pointer accordingly, although only the mouse click button is used (as a recording control) in this device at the moment." [38]

Of note is the usage of a microphone to record audio from the user to use in future Brain Opera performances. The T-Tree does not have any such sensors, but this may be a possibility in the future; for more detail, see Section 6.2.2.

The Singing Trees "have only a microphone as their input interface, and, like the speaking trees, have an LCD screen and stereo headphones (plus speakers in this case) as output devices" [38]. Both The Singing Tree and the T-Tree share multimodal feedback in the form of auditory and visual output. However, the Singing Tree supports more complex visual output via its full-color LCD screen. The T-Tree (and the Rhythm Tree) have comparatively low-bandwidth visual output via light-emitting diodes (LEDs).

The Rhythm Tree is a percussion instrument with many pads that detect when they have been struck. Within it, "up to 32 pads can be daisy-chained (like a string of Christmas lights) onto a single host and bus line. We have 10 such strings running in the Brain Opera Lobby. Each pad also houses a bright LED, which can be illuminated with a dynamically variable intensity" [38].

Both the T-Tree and the Rhythm Tree have auditory and visual output via speakers and LEDs. They also share a modular design. The "branches" of the T-Tree can be connected via Japan Solderless Terminal (JST) connectors to form a single strand of LEDs, strikingly similar to the description of the pads of the Rhythm Tree that can be strung together "like a string of Christmas lights."[3]

Another interface that allows for collective improvisation is the "reacTable," which is a tabletop tangible user interface (in the sense of [39]) used for live electronic music performance [40]. Users of the reacTable place specially-designed blocks on the surface of the table, which has a built-in projector and camera system that simultaneously reads the position and orientation of the blocks and projects visual feedback onto the surface of the table. Different blocks can act as sound sources or modulators.

Many design features of the reacTable contribute to its suitability for collective improvisation. Its circular design makes it easier for many people to gather around it. The usage of blocks makes each person's work at the table visible and editable by anyone else.

Both the reacTable and the T-Tree are interfaces that have a central "hub" fixed in space and satellite objects that users manipulate. For the reacTable, the table is the hub and the satellite

[3] As an aside, the T-Tree has used a Christmas Tree stand as its base.

objects are the blocks. With the T-Tree, the tree is the hub and T-Sticks are the satellite objects. One difference is that blocks must be used in conjunction with the reacTable whereas T-Sticks can be used outside the context of the T-Tree. The T-Tree also allows for more freedom of movement than the reacTable: users can move around anywhere in range of the T-Tree's WiFi network.

2.3.4 T-Stick "Adapters"

One final past effort that is relevant to the T-Tree is the work that has been done to adapt T-Stick signals from a legacy to a modern format.

As part of the work in [6] to maintain T-Sticks, the authors developed a "driver patch" in Max[4] that served as an adapter from the serial protocols of the older wired T-Sticks into the newer OSC-based message format of wireless T-Sticks.

The software adapter approach is an excellent step towards consolidating multiple standards. The T-Tree further improves on this work in two key ways:

First, all hardware and software used in the T-Tree, including the translation layer for wired T-Sticks, is open-source. The driver patch written by the authors above no longer functions on modern versions of Max nor can somebody without a paid license use it. Using open-source software makes DMIs more easily maintainable [41].

Second, the software is flexible in a way that makes it more difficult for beginners to approach the T-Stick. Developing configurable software can be "a curse" [42] that adds complexity. The T-Tree simplifies this complexity by automatically translating T-Sticks signals with no configuration required.

[4] `https://cycling74.com/products/max`

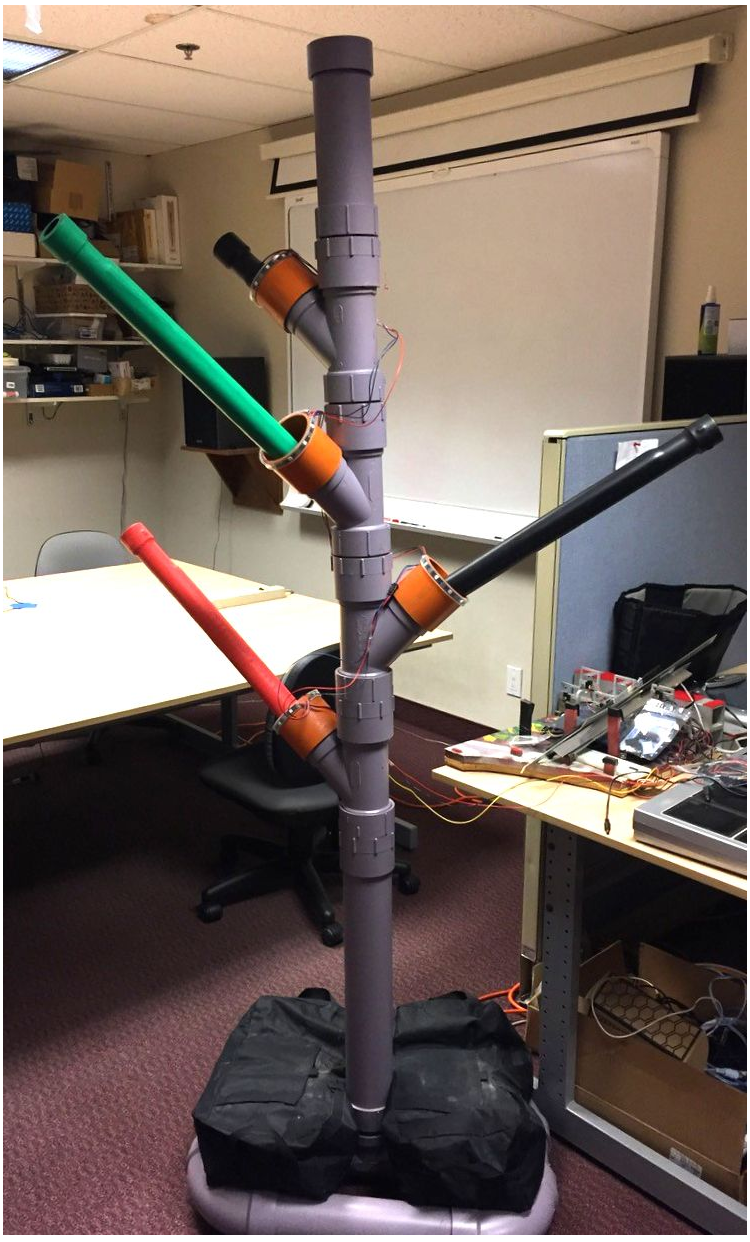

Fig. 2.2 The T-Tree 1.0 with four T-Sticks in IDMIL. This photo appears courtesy of Linnea Kirby and Marcelo Wanderley.

Chapter 3

Design of the T-Tree

This chapter documents the design of the T-Tree (1.0 and 2.0) as it emerged from the concept of a large bass T-Stick and changed into the structure that stands today. It specifically details the challenges with the T-Stick, proposed solutions to those challenges, and the various modalities in which the T-Tree can function.

3.1 Physical Design of the T-Tree

The T-Tree was originally conceived as a T-Stick so large that users would be able to play on it like a jungle gym, creating music with their interactions with the structure. However, it soon became apparent that this was infeasible within the time constraints we had. We shifted from the idea of a very large T-Stick to a device that interfaces with multiple T-Sticks. The name "T-Tree" came from extending the metaphor of the "stick."

Some of the original design philosophy of play within a large structure exists in the T-Tree. The T-Tree is much larger than a T-Stick, and designed to attract the attention of passersby with its size, colors, and lights. It is meant to invite play from curious onlookers and encourage exploratory music-making.

The T-Tree is designed to be modular, portable, and easy to assemble and disassemble. Our goals were to use the T-Tree in different settings—performance, installation, and public places—

and to make the physical structure as robust as possible. Both versions of the T-Tree use threaded PVC pipe to achieve stability and modularity. In addition, the parts of the T-Tree can be unscrewed and customized to suit the needs of different environments.

3.1.1 The Importance of Being Modular

The T-Tree's modular design was in part inspired by my experience with another project: a musical clock entitled "Chime Time." Chime Time is a collaboration between myself and John Williamson that is a 7-foot-high structure filled with chimes. The chimes can be activated via solenoids to "tell the time" encoded in a melody. Chime Time is constructed out of wood and uses screws and glue to connect its components. This means that while it is solidly built it cannot be deconstructed and is thus quite difficult to move.

This experience taught me that modularity in design, especially for devices that will need to be moved often, is extremely important.

3.2 Challenges With the T-Stick

The T-Stick, although a robust DMI, has some difficulties with its usability and collaborative potential. In this section, we document those challenges as they existed at the time of writing (2022-2023).

3.2.1 Flexibility and Programmability

The T-Stick is a sensor-rich device that embeds an ESP32 microcontroller and sends data to a sound-producing source via OpenSoundControl (OSC). This flexibility means that the instrument can produce virtually infinite sounds, with myriad possibilities for mapping.

However, this flexibility can be burdensome for a novice user who wants to have immediate music creation: "instant music, subtlety later" [42].

The T-Stick itself does not make sound. This leaves the task of "patch creation"—the selection of an appropriate synthesis device and creation of a mapping layer—to the end user.

Without prior knowledge of common frameworks that interface with OSC, such as Pure Data, SuperCollider, or Max, a novice user will have difficulty getting started with OSC [43]. In my limited experience at the IDMIL one of the most frequent questions for new users of the T-Stick was "how do I get it to make sound?"

3.2.2 Setup of T-Stick Communication

It is difficult to configure the T-Stick so that it is connected to the right Wi-Fi network. The process of setting up Wi-Fi information, although easier than it used to be in previous firmware versions, is still a major pain point for T-Stick users. A user of the T-Stick must be familiar with the concept of internet protocol (IP) addresses, be able to find their own IP address, and connect to several different Wi-Fi networks.

3.3 Proposed Solutions

This section details how to solve the aforementioned challenges with the T-Stick in the context of the T-Tree.

3.3.1 Separating Roles

One way that the T-Tree addresses the difficulty in flexibility and programmability is by assigning users to two categories: *Patch Creator* and *Patch User*.

By separating these roles, the majority of users can easily get started making music with the T-Stick as Patch Users, while those with more technical knowledge can configure the device as Patch Creators. Although it would be ideal to make the process of patch creation less technical, that is beyond the scope of this thesis. For one recent attempt at easing the process of patch creation, please see [44].

Patch Creator

The work of creating a patch, testing it with the T-Stick, and uploading it to the T-Tree is undertaken by the Patch Creator. This role requires some technical knowledge and skill.

The Patch Creator chooses a synthesis engine (either on the single-board computer [SBC] of the T-Tree or an external computer), creates a mapping layer between T-Stick gestures and sound, and uploads their patch to the T-Tree. This setup is eased by the T-Tree, as described in Section 4.5.4, but the Patch Creator must have knowledge of which ports their patch will run on and how to replicate and modify it for multiple T-Sticks.

Patch User

Any user, regardless of previous experience with the T-Stick, T-Tree, or DMIs in general, can fulfill the role of Patch User. All it requires is interacting with the T-Tree after it has been properly configured by the Patch Creator.

The Patch User takes a T-Stick from the T-Tree and moves it, touches it, and listens. They can experience music-making immediately without any further setup or technical knowledge.

Proper Configuration

The T-Tree is "properly configured" if and only if:

- It has at least one patch uploaded to it that produces sound.
- It has at least one T-Stick connected to it.
- It is connected to an external speaker or using its internal speaker.[1]

3.3.2 Easy Connection

Under the T-Tree 2.0, the process of connecting a T-Stick is as follows:

[1] The internal speaker is on the T-Tree 2.0 only.

1. A user plugs a T-Stick into the free USB port of the T-Tree 2.0.
2. The T-Tree 2.0 recognizes that a T-Stick has been plugged in and illuminates the buttons that correspond to a free branch.
3. A user presses a button to associate the plugged-in T-Stick with that particular branch.
4. The T-Stick is configured to send data to a patch running on that branch.

The details of this are further outlined in section 4.5.4.

3.4 Modalities of the T-Tree

The T-Tree can be viewed as a "digital musical instrument (DMI), an interactive music system (IMS), a hub, and/or a docking station that embeds several t-Sticks" [16]. The T-Tree may play one of these roles at any given moment or be used as several simultaneously. This section details each of these roles.

3.4.1 T-Tree as Docking Station

The T-Tree operates as a docking station in several ways. When a T-Stick is connected to it, it takes the appropriate action:

- For newer T-Sticks that send OSC over Wi-Fi, the T-Tree configures them with appropriate Wi-Fi information.
- For older T-Sticks that send data over USB, the T-Tree automatically translates their signals into the newer OSC format.
- All T-Sticks with a rechargeable battery will charge via USB.

The physical structure of the T-Tree makes it possible to leave T-Sticks in, either as parts of an interactive sculpture or for musicians to have their instruments ready before a performance. For one example of this, see Section 5.1.2.

3.4.2 T-Tree as Digital Musical Instrument (DMI)

The T-Tree can be considered a "DMI in two parts," in that it consists of the T-Tree itself, 1–4 T-Sticks, and a patch containing a sound engine and a mapping layer. As documented in [16]:

> "When multiple T-Sticks are connected to the T-Tree, they respond not only to the individual input of each performer, but can cross-modulate one another to create audiovisual output that would not be possible with four individual T-Sticks. Thus, it can function as a complex, multi-performer DMI, with the ability for composers to create many-to-many mappings."

This functionality could be replicated elsewhere without the need for the T-Tree (e.g., by using a laptop computer with several T-Sticks sending it OSC messages). However, the *physical object* of the T-Tree constitutes a DMI by embedding the three required components: sound generator, gestural controller, and mapping layer. It is less ephemeral and has a stronger identity as an instrument than the same functionality achieved by means of general-purpose computing technology like a laptop.

3.4.3 T-Tree as Interactive Music System (IMS)

The T-Tree can serve as an IMS, fulfilling the three criteria proposed in [26]:

- "Emphasize the experience"
- "Emphasize the intuitiveness"
- "Emphasize the liveness"

The following considers these three critera from the perspective of a Patch User.

Emphasize the Experience

The T-Tree emphasizes the experience of musicmaking because it does not require any further setup or thought about how to set up the instrument. Instead of giving the user a challenge to

overcome, the T-Tree offers the user an effortless audiovisual experience.

Emphasize the Intuitiveness

By removing the setup challenge from the T-Stick, the T-Tree can function as a more intuitive instrument than the T-Stick. Mapping design can be a rewarding artistic process but for somebody with no experience with DMIs the requirement to create one's one mappings can be a daunting barrier to entry. By having a Patch Creator design a soundscape that maps gesture to sound, the T-Tree is an intuitive IMS.[2]

Emphasize the Liveness

Using the T-Tree is an immediate and ephemeral experience. A notable absence from the T-Tree is a "record" or "loop" function, which could compromise the liveness of the instrument by taking users' focus away from the spontaneous act of musicmaking. Though such a feature could be developed in the future, it is not a current goal of the T-Tree.

3.4.4 T-Tree as Hub

The T-Tree metaphorically extends the idea of a "hub" in that it:

1. "Uses technology as a literal bridge between devices,
2. Is a central point around which activity takes place, and
3. Creates a space for collaboration." [16]

The T-Tree makes the T-Stick easier to play and presents it to those around it. As noted in [35], "the home studio proliferation undermines one of the most valuable traits of music—its collaborative and social nature—by promoting private and isolated musical practice where the value of live group interaction is marginalized." The *iltur* system reduced some of the effects of

[2]Yes, a Patch Creator could purposefully create an obtuse mapping that makes it difficult to understand the relationship between gesture and sound. But for the purposes of this analysis, I assume otherwise.

this trend "by allowing novices to become actively engaged in rich, thoughtful, and meaningful musical activities" [35]. The T-Tree continues in this tradition, as it allows users with little musical or technical experience to participate in collaborative music-making.

Fig. 3.1 The Chime Time clock, fully constructed. Note its "unibody" design, which cannot be easily disassembled.

Fig. 3.2 The second T-Tree 1.0, deconstructed into its modular components. Clockwise from top-left: the four branches, the base, and the brain. Threaded PVC pipe allows these components to be unscrewed for transport and storage.

Chapter 4

Implementation of the T-Tree

This section details the technical implementation of the solutions proposed in the previous chapter. Specific implementation details of wireless configuration, visual feedback, and backwards compatibility with wired T-Sticks are discussed. Finally, the improved version of the T-Tree, the "T-Tree 2.0," is introduced, and the differences from the T-Tree 1.0 are documented.

4.1 Physical Design

This section details the physical design of the T-Tree. The choices of building materials, the shape of the structure, and stability and robustness are discussed.

4.1.1 Choice of Materials

Both the T-Tree 1.0 and 2.0 are constructed out of polyvinyl chloride, or "PVC." Although PVC raises some ecological concerns [45], [46], it is a cheap, readily-available material, and a number of DMIs use it in their construction, including the T-Stick [47].

4.1.2 Structural Design

The T-Tree stands between five and six feet (1.52m–1.83m) tall, depending on the type of base used. There are a few different bases available for the T-Tree 1.0, and through the use of a PVC

adapter, the T-Tree 2.0 can use these bases or a base that was custom-designed for the T-Tree 2.0.

4.1.3 Ensuring Robustness

The base of the T-Tree 1.0 was designed to be reinforced with two 15-kilogram gravel-filled weight bags for robustness in public environments. For concert settings, including the *live@CIRMMT* concert (see Section 5.1.2 for details), these gravel bags were not necessary. However, if the T-Tree was to be used in a public environment where it could be bumped the gravel bags give it extra stability.

The electronics of the T-Tree sit inside the PVC structure to protect them from damage or tampering.

Fig. 4.1 Three T-Trees, with three different bases. From left to right: T-Tree 1.0 with "Christmas Tree" base, T-Tree 1.0 with PVC base, T-Tree 2.0 with thinner PVC base.

The T-Tree has four "branches" that hold T-Sticks and offer them to users. It uses threaded PVC pipe for easy assembly and disassembly.

4.2 Automatically Configuring Wireless T-Sticks

This section details the challenge and implementation of a solution for configuring T-Sticks to send OSC messages over a Wi-Fi network.

4.2.1 *A Priori* Communication Difficulty

One issue that current wireless T-Sticks face is the problem of *a priori* configuration vis-a-vis their wireless communication protocol. These T-Sticks use a 2.4GHz 802.11g transmission protocol, informally known as "Wi-Fi." Wi-Fi requires a Service Set Identifier (SSID) and pre-shared key (PSK) to begin communicating. But how does a user send this to the T-Stick if no means of communication has yet been established?

There were two different solutions when I arrived at the lab in 2021, neither of which was particularly easy or straightforward to execute.

The first, which required a wired connection to the T-Stick, had the user edit a specific `JSON`[1] file stored in the SPI Flash File Storage (SPIFFS) of the ESP32 microcontroller onboard the T-Stick. The user would plug in the T-Stick, download the firmware and supporting files, prepare a specifically-crafted `JSON` file containing the Wi-Fi information, run a command to create a SPIFFS-compatible binary stream, and upload that stream to the ESP32 controller. Needless to say, this was too complex and practically infeasible for most users of the T-Stick.

The second method, which was easier but not entirely satisfactory, did not require a wired connection to the T-Stick. Instead, it relied on putting the ESP32 microcontroller's onboard Wi-Fi chip into Access Point (AP) mode. A user would need to complete the following steps:

1. Take note of the Internet Protocol (IP) address of the computer on which sound-generating software is run.

[1] JavaScript Object Notation

```
{
    "device": "TStick",
    "id": 195,
    "author": "Edu Meneses",
    "institution": "SAT/IDMIL (2022)",
    "APpasswd": "my_secret_password",
    "wifiSSID": "2tree",
    "wifiPSK": "AP_password",
    "persistentAP": 1,
    "oscIP1": "192.168.4.2",
    "oscPORT1": 8001,
    "oscIP2": "192.168.0.100",
    "oscPORT2": 8002,
    "localPORT": 8000
}
```

Listing 1: An example `config.json` file used to configure a T-Stick.

2. Take note of a port for OSC messages to be processed into sound.

3. Hold down the hardware button on the T-Stick to make it enter Access Point mode.

4. Connect the computer the the Wi-Fi network being broadcast by the T-Stick, with its SSID being the name of the T-Stick.

5. Find the IP address of the T-Stick, which is now acting as the router for the sound-generating computer.

6. Connect to that IP address in a browser, which displays the configuration webapp.

7. Enter the Wi-Fi information of the network that the T-Stick will be using for transport of data.

8. Enter the IP address noted previously, as well as the port for OSC messages.

9. Click the appropriate button to restart the T-Stick, saving its configuration and exiting AP mode.

10. Connect to the Wi-Fi network used previously.

Although this process was less technically involved than the previous one, it still has a large number of steps, and requires advance preparation and knowledge of concepts like IP addresses and ports. In my casual experience with nontechnical users and T-Sticks, this setup process was one of the major barriers to entry.

The open-source signals library *libmapper*[2] can be used to obviate some of these steps, but the overall difficulty remains similar.

4.2.2 Puara Serial Manager

My solution to this problem was to implement a program that automatically detects most of the information required to configure a T-Stick and that does so automatically upon plugging in a T-Stick. Although this means that the configuration cannot be wireless it also greatly reduces the steps, difficulty, and possibility of error when configuring T-Sticks.

This program is written in Python 3, and supports any instrument that uses the Puara framework. Puara is "a framework for building and deploying New Media installations and New Interfaces for Musical Expression"[3] that powers the Media Processing Unit (MPU).[4] [48] [49] It is developed jointly at the Input Devices and Music Interaction Laboratory (IDMIL) and Société des Arts Technologiques (SAT) Metalab.[5]

As noted in [17]:

> "[Puara] comes preconfigured with common audio processing tools, including SuperCollider, Pure Data, and JackTrip. One beneficial side effect of using the Puara framework is that the T-Tree is now compatible not only with T-Sticks, but with any instrument that uses the Puara DMI toolkit. This could allow instrument designers to

[2] `libmapper.org`
[3] `https://github.com/Puara`
[4] The MPU was formerly known as the GuitarAMI Sound Processing Unit.
[5] `https://sat.qc.ca/fr/metalab`

more easily create DMIs that interface with the lights of the T-Tree or any future sensors/actuators."

The T-Tree runs the Puara Serial Manager when it boots, meaning that any wireless T-Stick (or other Puara-compatible device) connected to the T-Tree will, with zero intervention, be configured with the correct Wi-Fi information, IP addresses, and port numbers. Puara Serial Manager automatically increments the port number for each connected instrument so that different instruments will not conflict with one another.

4.3 Visual Feedback With LEDs

One striking feature of the T-Tree is its LED lights on each of its branches. The T-Stick by design does not give any feedback to the user. Once it is connected to a sound-generating system via a mapping layer it can take on some of these capabilities but the onboard hardware does not support it.

To provide multisensory feedback for those playing T-Sticks, LED lights were included in the design. These lights are available to patch designers via an OSC endpoint. The lights are individually-addressable LED strips that use the WS2812 Integrated Light Source,[6] which contain both full-color LEDs and an integrated circuit (IC) controller on the same physical strip [50].

4.3.1 Implementation of LED Code

The T-Tree contains code that allows users to control the LEDs via OSC. The code is written in Python and runs at boot time on the T-Tree via a `systemd` service.[7] It uses the `neopixel` Python library provided by Adafruit, which abstracts communication via the general-purpose input-output (GPIO) pins of the Raspberry Pi. The code exposes two OSC addresses:

- `/led_ring/set_ring/<ring>/`

[6] Informally known as "NeoPixels."
[7] `https://systemd.io/`

- `/led_ring/set_led/<ring>/<led>`

These OSC endpoints allow for a single branch (ring of LEDs) and an individual LED to be lit in a particular color. By sending a 3-tuple to these addresses, replacing `ring` with the branch number and `led` with the LED number, the user can light an LED or a ring with the RGB value represented by that tuple. The T-Tree LED code also includes example Max and VCV Rack[8] patches for testing the LED OSC server.

The T-Tree uses a Raspberry Pi 4 as its onboard SBC. Originally, the LED code sent signals to the LEDs via the software pulse-width modulation (PWM) interface of the Raspberry Pi's GPIO pins. However, this method prevented the usage of the onboard digital-to-analog converter (DAC) of the Raspberry Pi. Sound needed to be disabled for the LEDs to be enabled. I updated the code to send signals via the Raspberry Pi's Serial Peripheral Interface (SPI), which can be used to generate the requisite signal. The LEDs operate at a clock rate of 400 Hz [50], which both the PWM and SPI interface of the Raspberry Pi can operate at. The PWM output has a max clock rate of 8 kHz, and the SPI operates at the core clock frequency of the Raspberry Pi 4—1.5Ghz—which can be divided to achieve the desired clock rate [51]. By switching to SPI the T-Tree is capable of simultaneously outputting analog audio and a 400 Hz control signal for the LEDs.

To step up the voltage from the 3.3-volt signal sent by the GPIO pins of the Raspberry Pi to the 5 volts required for a signal to the NeoPixels, a Texas Instruments SN54AHCT125 Logic-Level Shifter is used between the Pi's SPI signal and the data input of the NeoPixels.

4.4 Translating Wired T-Stick Signals

This section details the process of creating a translation layer for older wired T-Sticks in supporting backward-compatibility with the T-Tree.

[8] `https://vcvrack.com`

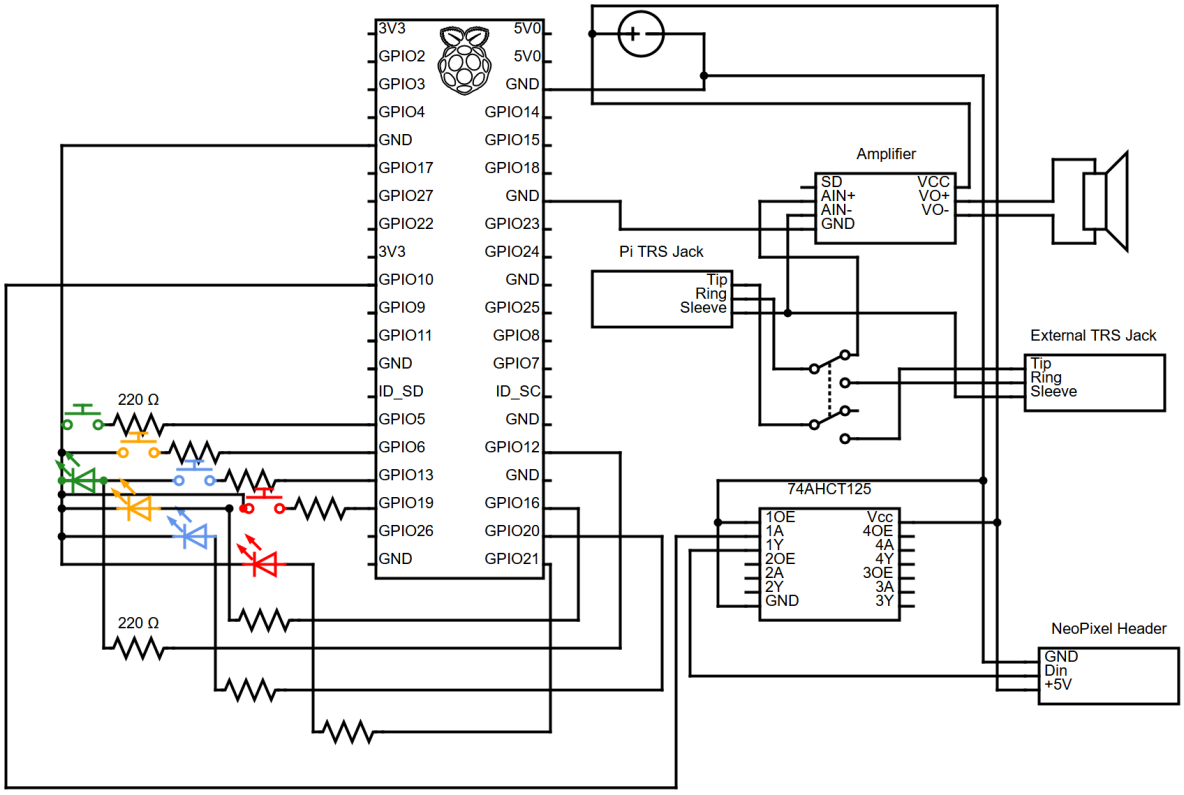

Fig. 4.2 A schematic diagram for wiring the T-Tree 2.0. The T-Tree 1.0 shares the same design for the NeoPixels but lacks the buttons, amplifier, speaker, and external TRS jack.

4.4.1 Introduction to Wired T-Sticks

There are currently eight wired T-Sticks in the IDMIL. A table of their properties follows.

The source code for the firmware, documentation, and schematic diagrams were all available to me when designing and building the T-Tree.[9]

[9] `https://github.com/IDMIL/Wired-T-Stick`

Number	Type	Variants
6	Soprano	2G, 2GX, 2G-IMU
1	Tenor	2G
1	Sopranino	2G

Table 4.1 Wired T-Sticks in IDMIL. *2G* means "second generation," with *2GX* and *2G-IMU* having additional capabilities.

4.4.2 Structure of the Wired T-Stick Signals

By analyzing the source code of the firmware, I was able to ascertain the logic of how the wired T-Sticks send signals over a serial connection. The source code was written in the Arduino programming language (a subset of C++) and specifically formatted as an "Arduino sketch." The source code contained function titles and comments that referred to the SLIP data format, which uses a special delimiter character to begin and end transmissions of data [52]. However, the actual implementation was much simpler, with the only special characters during transmission being a delimiter and an escape character, presumably to send a literal delimiter or escape code. In the SLIP implementation, a delimiter is used to start and end transmissions, but in the case of wired T-Sticks, the delimiter is only used to end transmissions. This means that when a connection is first established, we have no way of determining what previous communication has occurred nor the context of any signals received. Therefore, we must accommodate for this unknown state by discarding all messages until the delimiter character is received.

Wired T-Sticks break up their messages into different types using a particular "code" integer, followed by some data that corresponds to that code. For example, if the T-Stick is sending "periodic data," that is, data that is constantly updating (i.e., accelerometer or gyroscope) it will first send the byte `00000011`, followed by a series of bytes corresponding to the periodic data, followed by the delimiter character, `00000100`.

I designed a finite state machine (FSM) to decode the signals from the wired T-Sticks after I determined that this conceptual model would fit the signals being sent. Finite state machines are useful for representing particular models of communication and, under certain circumstances, can encapsulate all possible signals sent [53].

The FSM I designed assumes no previous communication with the sender and waits for a delimiter character to begin decoding the message. From there, it determines if the message is a well-formed signal, and if so, outputs it. Otherwise, it discards the message and begins the process again.

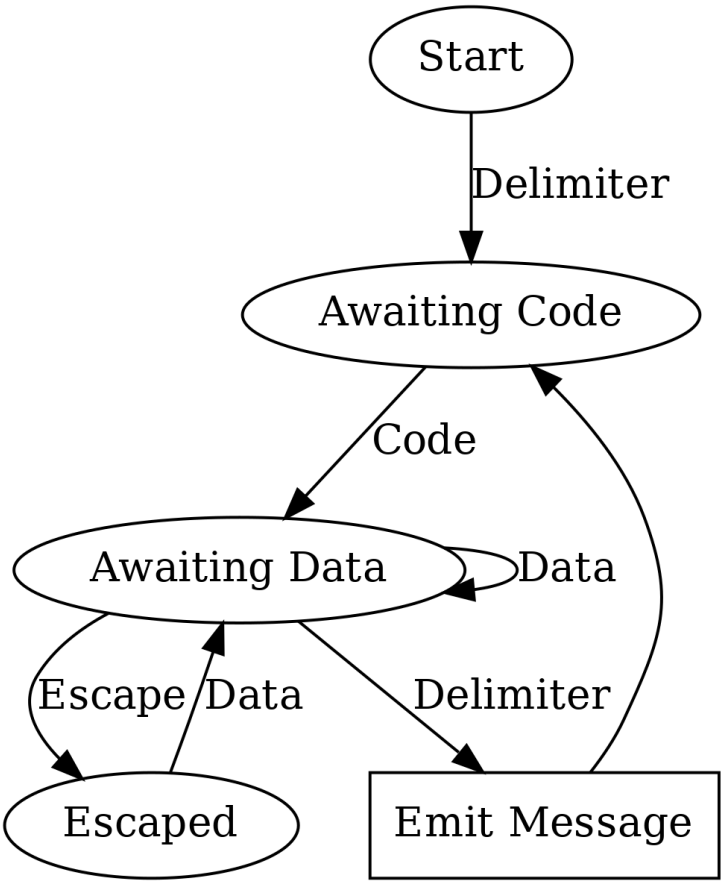

Fig. 4.3 Diagram of the wired T-Stick finite state machine.

Normally, signals over the serial connection are sent in one-byte increments. The `slipOutInt` function, however, encodes a 16-bit integer as two bytes, with the high-order bytes sent first, followed by the low-order bytes. This is handled by a corresponding function in software that reconstructs the original 16-bit integer from two 8-bit integers sent over the wire.

Touch data output by capacitive touch sensors is sent as a series of bytes that each correspond to a section of the T-Stick and whether that section detects the presence of a substance that

induces capacitance in that area of the instrument. This is binary data encoded as a series of bytes with a bitmask. To decode it, we read the series of bytes, apply the inverse bitmask, and reconstruct the touch data.

4.4.3 Structure of the Translation Program

The program I wrote to translate the serial data from wired T-Sticks into OSC data is written in the Python 3 programming language.[10] It is broken into three parts, a lexer, a parser, and the OSC server. The lexer, contained in `tstick_lexer.py`, reads the raw data from the stream of bytes over the serial connection and converts this data into tokens, which I call "messages." A message contains two fields: `code` and `data`. The code specifies the type of message. The data carries the information and can be arbitrarily long.

The lexer queues messages to be read in a separate thread by the parser. In order to accommodate T-Stick firmware with slightly different encoding methods, I created an abstract base class (ABC) that represents a T-Stick, including its baud rate, name, and different parsing methods. The T-Stick version can be selected at runtime and instantiates the correct concrete implementation of the T-Stick ABC. At the time of writing, there are two concrete implementations of a T-Stick translator, the Soprano T-Stick with serial number 012 and the Sopranino numbered 172.

The lexer, parser, and OSC server run in separate threads. Although Python is not well suited to multithreaded CPU-bound software due to its Global Interpreter Lock (GIL) [54] [55], for this project it handles the problem adequately.

The lexer and parser use an event-based publish/subscribe model, which can allow for further additions to the software to be more easily added in the future [56]. For example, currently the software translates serial data from wired T-Sticks into OSC messages that are "looped back" to the local host. In the future, it may be preferable to use a form of inter-process communication (IPC)[11] for transferring messages from the parser directly to the sound-generating software. With the modular, asynchronous, and decoupled design of the wired T-Stick translation software, this

[10] `https://github.com/Puara/puara-serial-manager/tree/main/wired`
[11] For example, UNIX sockets.

change could be created more easily in the future.

4.5 Details of the T-Tree 2.0

Numerous improvements in both software and hardware have been implemented since the original version of the T-Tree. A new version of the T-Tree that includes these improvements has been constructed: the "T-Tree 2.0." This section outlines the various improvements that the T-Tree 2.0 makes over the T-Tree 1.0.

4.5.1 Unified Software Platform: Puara

The T-Tree 1.0 used a generic Raspberry Pi operating system (OS) image.[12] It did not allow for onboard audio and required that the user downloaded all requisite programs for signal processing.

The T-Tree 2.0 uses Puara, which provides a stable, standardized environment for sound and signal processing. I chose to adopt Puara as the standard computing environment for the T-Tree because it was already optimized for embedded computing and would reduce the number of different computing environments in the lab. It also comes preconfigured with common audio processing tools, including SuperCollider and Pure Data, which are "established software used by other communities within NIME" [57].

Using Puara eases the setup of the computing environment of the T-Tree and makes it easier for developers of patches for the T-Tree to rely on a standardized computing environment. By establishing a consistent set of hardware and software, the T-Tree can help to alleviate some of the issues caused by a multiplicity of microcontrollers and sensors [6].

4.5.2 Improved Physical Structure and Environmental Concerns

When designing the T-Tree 2.0, I wanted to keep the things that worked from the original T-Tree while making it lighter, easier to set up, and require less materials and build time. The 3-inch (76.2mm) Schedule 40 PVC and ABS pipe that was used for the entire structure of the T-Tree 1.0

[12] `https://www.raspberrypi.com/software/`

has been replaced with 1-inch (25.6mm) furniture-grade and 2-inch (50.8mm) Schedule 40 PVC pipe. Because furniture-grade PVC does not need to be rated for pressurization or water tightness it is also less expensive than Schedule 40 PVC and ABS pipe.

As noted in [45] and [46], PVC pipe is not an environmentally-friendly material. In future iterations beyond the T-Tree 2.0, using recycled materials would be preferable to buying off-the-shelf materials. For the time being, I have reduced the overall amount of material required to construct the T-Tree by reducing the diameter of its PVC.

The T-Tree 2.0 features an adapter that allows its "brain" to be used on the T-Tree 1.0. This is accomplished via a reducing 3-inch x 2-inch (76.2mm x 50.8mm) PVC nipple that allows the threaded ends of the T-Tree 2.0 to attach to those of the T-Tree 1.0. This will reduce waste by allowing reuse of the previous iterations of the T-Tree.

4.5.3 External User Interface

The T-Tree 2.0 has more connectivity than the T-Tree 1.0. For its user interface, it features a panel with four colored buttons representing the four different branches. For ports, has an Ethernet port for internet connectivity, a DC barrel jack, a USB-C port for power, and a USB-A port for connecting and configuring T-Sticks. It also has a 3.5mm TRS jack for connecting to an external amplification system and a double pole double throw (DPDT) switch that toggles between the internal speaker and the external TRS jack.

Internally, the T-Tree 2.0 faceplate connects to the Raspberry Pi 4 via a 40-pin ribbon cable, an internal Ethernet cable, two internal USB cables, an internal 3.5mm TRS cable, and individual wires for the buttons and LEDs. There is a circuit board with the components needed to route the GPIO connections for the buttons and the 5-volt power connection to the amplification system and NeoPixels. The faceplate is installed in the brain of the T-Tree 2.0 with hot glue, striking a balance between solidity and ease of removal for maintenance.

The T-Tree 2.0 also has a mono, 3-watt, 4-ohm speaker embedded in its brain powered by a class D amplifier. The addition of an internal speaker makes the T-Tree 2.0 more portable and

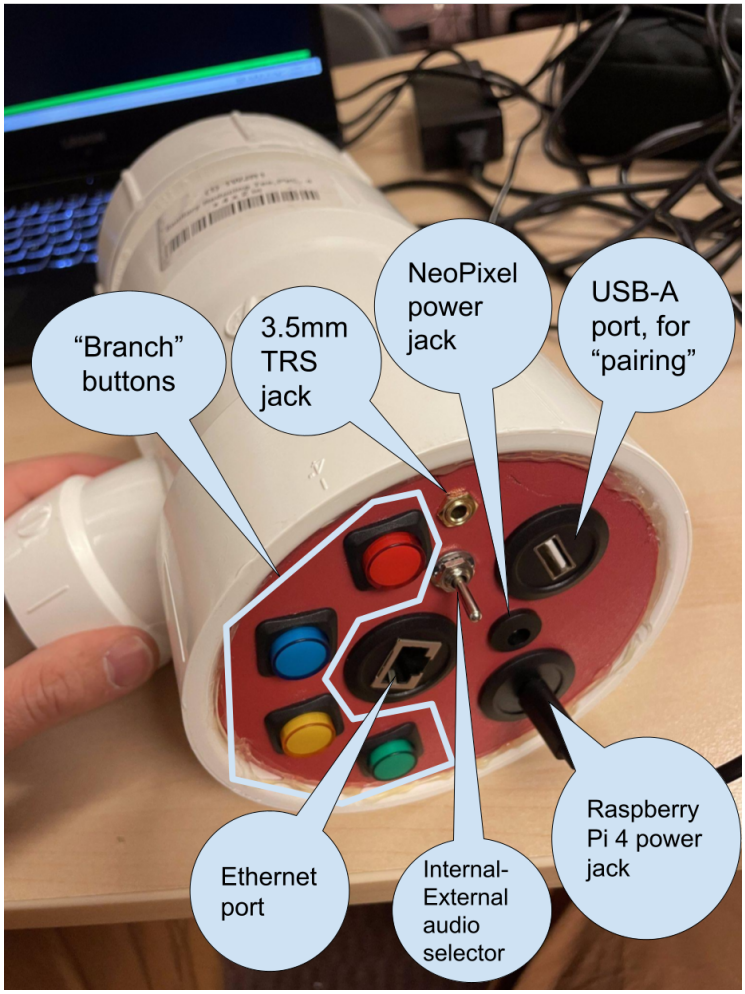

Fig. 4.4 The faceplate of the brain of the T-Tree 2.0 with its external connections labeled.

more of a self-contained instrument.

Originally, I had planned to power the amplifier, LEDs, and Raspbery Pi from the same 5-volt power supply. In testing this the Raspbery Pi would reboot and became unstable. For this reason the Raspberry Pi is powered by a separate 5-volt USB-C connector. In the future more work could be done to unify the power supplies.

4.5.4 T-Stick "Pairing"

The T-Tree 2.0 has a concept of "pairing" a T-Stick with a particular branch of the tree. I borrow this term from Bluetooth devices, where pairing refers to the cryptographic handshake between device and host [58]. Neither the T-Tree nor the T-Stick currently have any kind of data encryption although that may be a consideration in the future.

The T-Tree 2.0 improves upon the Puara Serial Manager for easily connecting T-Sticks to a synthesis device via a mapping layer. First, a Patch Creator creates patches and transfers them to a specific directory on the disk of the T-Tree 2.0. Then, the T-Tree 2.0 software will automatically load them when a T-Stick is connected.

The T-Tree 2.0 maintains an association between different T-Sticks and branches and allows the user to easily pair a T-Stick with a branch of the tree. When a T-Stick is plugged into the USB-A port of the faceplate the T-Tree 2.0 recognizes it and begins the pairing process. At this point, the four buttons on the T-Tree blink indicating that the user must choose which branch to pair with. When the user presses a button, the T-Stick is configured to send OSC data to the T-Tree on a port that is associated with that particular branch, and the default patch is loaded. Subsequent button presses change the currently loaded patch on that branch.

Branch Index	Branch Color	OSC Port
0	Green	8000
1	Yellow	8001
2	Blue	8002
3	Red	8003

Table 4.2 The different branches of the T-Tree, their associated button colors, and their OSC ports.

Chapter 5

Discussion

This chapter discusses how the T-Tree has been used in "real-world" contexts, including performance and installation. The original goals of the T-Tree are restated and its effectiveness in meeting them is evaluated.

5.1 Performance Usage

At the time of writing, the T-Tree has been used in two public performances. The first was an interactive art installation called "The Windy Days" that took place in June 2022. The second was an electroacoustic concert called "Improvising New Winds" in December 2022 that was part of an ongoing concert series called "*live*@CIRMMT."

5.1.1 The Windy Days - Art Neuf Residency

The T-Tree was used in an interactive art installation known as The Windy Days, which took place in June 2022 at a venue called Art Neuf. The Windy Days, created by Lucy Fandel, describes itself as examining "how to weave together the discarded pieces of our habitats (plastic bags, rustling leaves, an act as complex as a slow walk) to (re)connect us to the subtle and urgent transformations of our most familiar landscapes."[1]

[1] `https://thewindydays.ca/`

In one performance of The Windy Days, "dancers could interact with the installation and demonstrate the interactions possible within the space, which the public would also be free to explore. The T-Tree 1.0 and two T-Sticks were used as the interactive hub for the installation." [17]

Kasey Pocius was the sound designer for this project, and created a soundscape running in Max that featured sounds of leaves rustling, wind blowing, and other natural phenomena. The T-Tree was connected to a laptop that ran Max, and the sounds were sent to a small speaker hidden inside a wall.

The mapping design of the two T-Sticks in The Windy Days was such that the two instruments would cross-modulate each other in subtle ways. For example, one T-Stick controlled the amount of reverb, while the other controlled the level of a resonant filter bank. The sonic installation would produce vastly different sounds depending on whether one T-Stick, or the other, or both were being used.

The T-Tree and T-Sticks were preconfigured to make sound when squeezed or moved, and did not require any prior knowledge to operate. Though I was not involved in the creation of the project, I did attend as a spectator to observe how people interacted with the T-Tree. I noticed that only the dancers interacted with the T-Tree, most likely because the other participants did not know it was interactive.

Interactive or Not?

The notion of a "composed instrument" [59] is a useful lens with which to analyze the T-Tree in its usage as an installation. A composed instrument "carries as much the notion of an instrument as that of a score" [59]. The T-Tree could be considered a composed instrument, as it imparts this "notion" to the users of its T-Sticks. A further question is how to make passersby aware that the device is interactive.

As mentioned previously, I observed that the only people to interact with the T-Tree were those who already knew it was interactive (the dancers, Pocius, and myself). In other words,

Fig. 5.1 The T-Tree broken down into its component parts in Art Neuf before the exhibition.

the device did not effectively communicate its interactivity to a wide audience. This could be a problem with the physical structure of the T-Tree. The branches, although they are literally holding the T-Sticks outwards, do not show that the T-Sticks can be taken from the device.

Because the T-Tree is portable and does not rely on a specially-outfitted space its function can be opaque. Furthermore, if the sound source is not the T-Tree itself it is unclear that the T-Tree creates what one hears. The addition of an internal speaker in the T-Tree 2.0 helps reinforce the association between the T-Tree and the sound it creates. People could also be informed of the T-Tree's interactivity simply with a sign or vocal instructions.

Fig. 5.2 The T-Tree fully constructed in Art Neuf during the exhibition.

5.1.2 The *live@CIRMMT* Performance

The T-Tree 1.0 was exhibited at a concert called "Improvising New Winds" which was part of an ongoing concert series called *live*@CIRMMT. For this concert, "we wanted to create something that would clearly associate the gestures of each performer with a particular branch of the T-Tree" [17]. To achieve that goal, "one performer controlled the bass voices, one had noise material, and one had a mixture of melodic and percussive voices" [17].

This concert gave us the opportunity to use the T-Tree in a large multichannel environment. "Improvising New Winds" took place in the Centre for Interdisciplinary Research in Music Media

and Technology (CIRMMT)[2] multimedia room (MMR). The MMR contains an array of 70 loudspeakers,[3] allowing for highly spatialized audio. To take advantage of this system Pocius designed a soundscape that used the T-Sticks' IMUs to "place" our sounds in different parts of the room and "the spatialization also acted to add additional emphasis to the movement of each T-Stick" [17].

In addition to spatialization, we used the LEDs on the T-Tree to associate gesture with sound. To begin our performance each T-Stick was in a T-Tree branch and the branches were unlit. When we came onstage we each took our T-Stick from a branch which then illuminated in a color mapped to the timbre of the sound we produced.

Fig. 5.3 T-Tree performance in *live@CIRMMT*, January 4, 2021. Pictured left to right: Buser, Pocius, Kirby.

[2] https://www.cirmmt.org/
[3] https://www.cirmmt.org/en/facilities

T-Stick Failure In Concert

For completeness' sake, I document here a T-Stick failure that occurred during the *live@CIRMMT* concert.

Although we had successfully rehearsed numerous times throughout the week with neither the T-Tree nor any of the T-Sticks having any issues, we did have a T-Stick fail in performance. Once the show began one T-Stick malfunctioned and did not send any signals to the T-Tree. The ultimate cause was determined to be a failure of the battery system of the T-Stick.

The T-Tree continued to route the signals from the other two T-Sticks and we were able to continue our piece without stopping the show.

Though improvements in T-Stick reliability are beyond the scope of this thesis, at the time of writing T-Stick reliability is being actively improved [60].

5.2 Evaluation of the T-Tree

This section presents an evaluation of the T-Trees (both 1.0 and 2.0) in the context of their goals.

5.2.1 T-Tree 1.0

The goals of the T-Tree 1.0 have already been outlined in Section 1.1.1 but I will reiterate them here. The original goals of the T-Tree 1.0 were twofold:

- (a) to lower the "entry fee" of using a subset of DMIs (T-Sticks)
- (b) mitigating obsolescence of older T-Sticks.

Let us consider these two goals in turn.

Lower "Entry Fee" for T-Sticks

Although a formal usability study of the T-Tree has not been performed, I believe there is sufficient evidence to conclude that the T-Tree 1.0 has lowered the "entry fee" [22] of the T-Stick family of

DMIs. As documented in Section 2.1.4, the T-Stick has had a number of barriers to usability that the T-Tree has addressed.

The setup of the instrument has been greatly reduced in complexity. This includes setting up Wi-Fi or in the case of wired T-Sticks, providing a translation layer that allows the signals to be interpreted as OSC data.

Furthermore, the setup of the instrument in terms of sound design has been simplified by separating Patch Creators and Patch Users, as described in 3.3.1. This allows a novice performer who has no experience with DMIs nor the technical knowledge to create their own patches (e.g., in Pure Data or Supercollider), to use the instrument right from the T-Tree with no additional setup.

The T-Tree has helped overcome two major barriers to entry for new T-Stick players: setup of communication and patching the instrument.

Mitigate Obsolescence of Older T-Sticks

The T-Tree has continued in the tradition of maintaining the T-Stick for future users as described in Section 2.3.4. By creating a standalone hardware device that interfaces with both newer, wireless T-Sticks and older, wired T-Sticks, they can continue to be used rather than scrapped for parts or thrown away.

Although hardware maintenance will continue to be a challenge the T-Tree improves stability and uniformity in the T-Stick ecosystem, helping to preserve the instrument and its communal practice.

5.2.2 T-Tree 2.0

The T-Tree 2.0 is an extension and improvement over the T-Tree 1.0 and is evaluated here on its own and in context of the goals of the T-Tree project.

Continuing to Meet Goals of T-Tree 1.0

The T-Tree 2.0 contains a strict superset of the functionality contained in the T-Tree 1.0. Although it has not been formally evaluated nor publicly exhibited, it meets the same functional requirements as the T-Tree 1.0 such as the inclusion of LED lights, SBC, stability of structure, and modular design.

Meeting the Goals Set Forth in "Introducing the T-Tree"

The first published work on the T-Tree presents three goals for future development: [16]

1. Allowing T-Sticks to update their firmware via the T-Tree.
2. Using the GuitarAMI Sound Processing Unit as the embedded processor to further standardize the sound generation in the T-Tree.
3. Implementing libmapper as a means to map input parameters to synthesis parameters and LED colors/intensities.

This section addresses these goals vis-a-vis the T-Tree 2.0.

First, the ability to update firmware via the T-Tree is theoretically possible using open-source tools like `espflash` or `esptool`. However, this would not be a good user experience in the context of the roles outlined in Section 3.3.1. The Patch Creator will have the technical expertise to update T-Stick firmware without assistance from the T-Tree. And as a Patch Creator the primary focus is sound design and creating an experience for the Patch User. Updating firmware—although it can improve the signals via which this experience can be achieved—is orthogonal to this process. As a Patch User updating the firmware is even less of a priority.

Updating the firmware of the T-Stick must also be considered in the context of the system it comprises in conjunction with the T-Tree. The T-Tree's software is designed to work with specific T-Stick firmwares. If a new firmware for T-Stick is developed a corresponding patch to the T-Tree software must also be applied. Therefore, updating the firmware of T-Sticks and the software of

the T-Tree is an activity that is tightly coupled and that happens "out-of-band" with the process of creating and using musical patches. For this reason and the above, the goal of allowing T-Sticks to update their firmware from the T-Tree is not a priority at this time.

The second goal of "using the GuitarAMI Sound Processing Unit (SPU) as the embedded processor" for the T-Tree 2.0 has been accomplished insofar as the two systems now share a common software platform: Puara [17]. The SPU is now known as the Media Processing Unit (MPU). Literally using the MPU—including case, screen, and buttons—was not implemented because the devices serve two fundamentally different purposes: the MPU is a general-purpose audio processing unit whereas the T-Tree is specifically designed to interface with T-Sticks. The physical design of the T-Tree 2.0 is inspired by the physical structure of the MPU, in particular the choice to include physical buttons on the interface and have external connections for power, audio, and Ethernet. The software whenever possible has been standardized to share a common platform. To this extent, the T-Tree 2.0 meets the aforementioned goal.

The third goal of "implementing libmapper as a means to map input parameters to synthesis parameters and LED colors/intensities" remains yet to be accomplished. The Raspberry Pi 4 uses an ARM processor. Although there is some evidence of individual users compiling libmapper on ARM it remains at the discretion of libmapper developers to create a simple and easy-to-use binary release of their software that runs on ARM. There is some promising work in this direction, in particular the efforts of [61] to apply Continuous Integration (CI) practices to the software release cycle of libmapper.

Meeting the Goals in "Towards the T-Tree 2.0"

The 2023 paper "Towards the T-Tree 2.0" outlines three main areas for improvement from the T-T-Tree 1.0 to the T-Tree 2.0: [17]

1. Structural redesign.

2. Optimization of the T-Tree as installation.

3. Tighter integration with the T-Stick.

Now that the T-Tree 2.0 has been constructed, I evaluate it in the context of these goals.

Structural Redesign

The structural redesign goals for the T-Tree 2.0 have been met. The brain of the T-Tree 2.0 has been expanded in size and can now house the SBC, a circuit board, and an internal speaker. External ports for USB-C, a DC barrel jack, a 3.5mm TRS jack, a DPDT switch, and a USB-A jack have been added. Instead of multiple USB-A jacks as proposed in [17] I opted for a single USB-A jack to reduce the complexity of pairing T-Sticks with the T-Tree.

In addition, the "rudimentary user interface" described in [17] has been implemented, with four light-up buttons that give users a way to change patches for up to four paired T-Sticks, as well as provide visual feedback during the pairing process. Furthermore, the T-Tree 2.0 uses 2-inch (50.8mm) PVC pipe as opposed to the T-Tree 1.0's 3-inch (76.2mm) PVC. This reduces the total amount and cost of the material, making it a lower-cost and more environmentally sustainable instrument.

Optimization of T-Tree as Installation

The second goal of optimizing the T-Tree for installation use has not yet been accomplished. Meeting this goal requires formal user testing and evaluation as opposed to the informal evaluations that the T-Tree has undergone. For future work in this direction, see Section 6.2.4.

Tighter Integration with the T-Stick

There remains work to make the T-Tree and T-Stick function as a single instrument. The T-Tree has improved the usability of the T-Stick. However, the original purpose of the T-Tree was to adapt the existing capabilities of T-Sticks, rather than change them. Now that the T-Tree is more established, there can be greater creative development between the T-Tree and T-Stick. For future work in this direction, see Section 6.2.3.

Chapter 6

Conclusions and Future Work

In this thesis I have introduced a new technical artifact—the T-Tree—that adapts and augments a subset of DMIs—T-Sticks. Previous work in sonic installation, collective improvisation, novice music-making, and T-Stick adaptation has been documented and their influence on the T-Tree catalogued. The design of the T-Tree was summarized, and its implementation detailed. Finally, the project was evaluated in context of the goals previously set.

This chapter summarizes this work in the context of the broader field of DMI research, evaluates the impact of the T-Tree project, and suggests future work to be done.

6.1 Impact of the T-Tree

At the time of writing, there are three T-Trees constructed: two T-Tree 1.0s and one T-Tree 2.0. These technical artifacts are in IDMIL at McGill University, where they can be accessed by members of the IDMIL and other music technology students.

In the two years since the T-Tree's creation it has improved the user experience of using the T-Stick. It has also been used as an instrument and an installation. This project continues the work of [6] in the process of keeping the T-Stick—a 15-year-old DMI—alive.

The combination of Puara software with a hardware interface specifically designed for T-Sticks means that users with no previous technical knowledge can begin making sound with the T-Stick

right away. More proficient users can create their own patches, utilizing the multimodal visual and auditory feedback that the T-Tree affords them.

More broadly, the T-Tree acts as an *adapter* for T-Sticks, especially wired T-Sticks (see Section 4.4). It allows these devices to function in a new context without needing to make changes to the devices themselves. This model of creating a new device as an adapter while not requiring structural changes to existing devices can serve as a useful example for others working in the DMI space.

6.2 Future Work

There remains some crucial work to be done on the T-Tree. This section will highlight some key future directions in the T-Tree project.

6.2.1 Formal User Study and Evaluation

The T-Tree has been evaluated in an informal, anecdotal, and ad-hoc context by myself and my peers at IDMIL and in two public exhibitions. This has yielded a groundwork for what has worked and not worked but there is still much we don't know. For example, using the T-Tree in "The Windy Days" allowed for informal observation of interactions with the T-Tree, but did not control for variables such as participants' familiarity with DMIs.

A more rigorous and controlled user study with participants of varying levels of exposure to DMIs could provide a number of useful findings about how to improve the device.

6.2.2 More Sensors

Neither the T-Tree 1.0 nor 2.0 have any onboard sensors, though the onboard SBC is capable of processing sensor signals. In the future, the T-Tree could make use of sensors such as an infrared distance sensor, ambient light sensor, or microphone to enhance the multisensory experience of playing it.

The addition of various sensors could also help solidify the T-Tree's identity as a DMI.

6.2.3 Tighter Integration with the T-Stick

Although it was not a goal of the original T-Tree to change the design of the T-Stick, the T-Tree is now a mature enough instrument that both it and the T-Stick could benefit from features developed jointly between the two interfaces. For example, [17] suggests the addition of ultra-wideband (UWB) transceivers on the T-Tree and T-Stick, which would facilitate ultra-low-latency communication between the two devices.

6.2.4 Further Installation Use and Development

Further installation use of the T-Tree is crucial to understand what appeals, what does not, what is obvious and what is not, and how people interact with the device in a real-world context.

The continued use in an installation context is also part of a virtuous cycle of development that can lead to improvements and optimizations of the T-Tree as a sonic installation.

Furthermore, more research is needed on ways of effectively presenting the T-Tree and its component T-Sticks to users, especially those with no previous exposure to DMIs.

6.2.5 Support More Puara-Based Instruments

At the time of writing (2022-2023) the following instruments use the Puara framework:

- T-Tree
- T-Stick
- Media Processing Unit (MPU)
- GuitarAMI
- AMIWrist
- Probatio

These instruments share a common "vocabulary" of gestures, making it easier to develop software for them.

There remains work to expand the T-Tree as a unified hardware/software platform for Puara-based instruments. For example differently-shaped branches for instruments of different weights and sizes could be added. Another possibility is that the T-Tree itself be instrumented to react to Puara gestures rather than embedding a mapping layer that reacts to these gestures. The sensors mentioned in Section 6.2.2 could assist with this goal.

All of these improvements would help not only the T-Tree project but also the health of the Puara ecosystem.

Bibliography

[1] J. Gibson, *The Ecological Approach to Visual Perception*. New York, NY: Houghton Mifflin, 1979.

[2] W. W. Gaver, "Technology Affordances," in *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 1991, pp. 79–84.

[3] E. R. Miranda and M. M. Wanderley, *New Digital Musical Instruments: Control and Interaction Beyond the Keyboard*. AR Editions, Inc., 2006.

[4] S. S. Fels and M. J. Lyons, "Advances in New Interfaces for Musical Expression," in *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, 2011.

[5] I. Franco and M. M. Wanderley, "Prynth: A Framework for Self-Contained Digital Music Instruments," in *Proceedings of the 12th International Symposium on Computer Music Multidisciplinary Research*, 2017, pp. 357–370.

[6] A. Nieva, J. Wang, J. W. Malloch, and M. M. Wanderley, "The T-Stick: Maintaining a 12 year-old Digital Musical Instrument," in *Proceedings of the 2018 Conference on New Interfaces for Musical Expression*, 2018, pp. 198–199.

[7] J. Sullivan and M. M. Wanderley, "Stability, Reliability, Compatibility: Reviewing 40 Years of NIME Design," McGill University, Research Report, Mar. 2018. [Online]. Available: `https://hal.science/hal-01745984` (visited on 07/08/2023).

[8] F. Morreale and A. McPherson, "Design for Longevity: Ongoing Use of Instruments from NIME 2010-14," in *Proceedings of the 2017 Conference on New Interfaces for Musical Expression*, 2017, pp. 192–197.

[9] G. Torre, K. Andersen, and F. Baldé, "The Hands: The Making of a Digital Musical Instrument," *Computer Music Journal*, vol. 40, no. 2, pp. 22–34, 2016.

[10] S. Ferguson and M. M. Wanderley, "The McGill Digital Orchestra: An Interdisciplinary Project on Digital Musical Instruments," *Journal of Interdisciplinary Music Studies*, vol. 4, no. 2, pp. 17–35, 2010.

[11] A. McPherson and Y. E. Kim, "The Problem of the Second Performer: Building a Community Around an Augmented Piano," *Computer Music Journal*, vol. 36, no. 4, pp. 10–27, 2012.

[12] K. E. Vaniea, E. Rader, and R. Wash, "Betrayed by Updates: How Negative Experiences Affect Future Security," in *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2014, pp. 2671–2674.

[13] R. Masu, N. N. Correia, and T. Romao, "NIME Scores: a Systematic Review of How Scores Have Shaped Performance Ecologies in NIME," in *Proceedings of the 2021 Conference on New Interfaces for Musical Expression*, 2021.

[14] G. Loy, "Musicians Make a Standard: The MIDI Phenomenon," *Computer Music Journal*, vol. 9, no. 4, pp. 8–26, 1985.

[15] A. Schmeder, A. Freed, and D. Wessel, "Best Practices for Open Sound Control," in *Proceedings of the Linux Audio Conference*, 2010.

[16] L. Kirby, P. Buser, and M. M. Wanderley, "Introducing the t-Tree: Using Multiple t-Sticks for Performance and Installation," in *Proceedings of the 2022 Conference on New Interfaces for Musical Expression*, 2022.

[17] P. Buser, K. Pocius, L. Kirby, and M. M. Wanderley, "Towards the T-Tree 2.0: Lessons Learned From Performance With a Novel DMI and Instrument Hub," in *Proceedings of the 2023 Conference on New Interfaces for Musical Expression*, 2023.

[18] J. W. Malloch and M. M. Wanderley, "The T-Stick: From Musical Interface to Musical Instrument," in *Proceedings of the 2007 Conference on New Interfaces for Musical Expression*, 2007, pp. 66–70.

[19] D. A. Stewart, "Digital Musical Instrument Composition: Limits and Constraints," in *Proceedings of the Electroacoustic Music Studies Network Conference*, 2009, pp. 3–8.

[20] M. Kirkegaard, M. Bredholt, C. Frisson, and M. M. Wanderley, "Torquetuner: a Self Contained Module for Designing Rotary Haptic Force Feedback for Digital Musical Instruments," in *Proceedings of the 2020 Conference on New Interfaces for Musical Expression*, 2020, pp. 273–278.

[21] T. Fukuda, E. A. Meneses, T. West, and M. M. Wanderley, "The T-Stick Music Creation Project: An Approach to Building a Creative Community Around a DMI," in *Proceedings of the 2021 Conference on New Interfaces for Musical Expression*, 2021.

[22] D. Wessel and M. Wright, "Problems and Prospects for Intimate Musical Control of Computers," *Computer Music Journal*, vol. 26, no. 3, pp. 1–22, 2002.

[23] M. M. Wanderley, N. Orio, and N. Schnell, "Towards an Analysis of Interaction in Sound Generating Systems," in *Proceedings of the International Symposium on Electronic Arts*, 2000.

[24] J. W. Malloch, "A Consort of Gestural Musical Controllers: Design, Construction, and Performance," M.S. thesis, McGill University, 2008.

[25] A. Nieva, "Maintenance Strategies and Design Recommendations on Input Devices for Musical Expression," M.S. thesis, McGill University, 2018.

[26] Y. Wu and N. Bryan-Kinns, "Musicking with an interactive musical system: The effects of task motivation and user interface mode on non-musicians' creative engagement," *International Journal of Human-Computer Studies*, vol. 122, pp. 61–77, 2019.

[27] R. Bandt, "Sound Installation: Blurring the Boundaries of the Eye, the Ear, Space and Time," *Contemporary Music Review*, vol. 25, no. 4, pp. 353–365, 2006.

[28] J. Jaimovich, "Ground Me! An Interactive Sound Art Installation.," in *Proceedings of the 2010 Conference on New Interfaces for Musical Expression*, 2010, pp. 391–394.

[29] B. Bongers, "Physical Interfaces in the Electronic Arts," in *Trends in Gestural Control of Music*, M. M. Wanderley and M. Battier, Eds., Paris, France: IRCAM, 2000, pp. 41–70.

[30] G. Leslie, B. Zamborlin, P. Jodlowski, and N. Schnell, "Grainstick: a Collaborative, Interactive Sound Installation," in *Proceedings of the International Computer Music Conference*, 2010.

[31] E. Frid, H. Lindetorp, K. F. Hansen, L. Elblaus, and R. Bresin, "Sound Forest: Evaluation of an Accessible Multisensory Music Installation," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–12.

[32] T. Blaine and S. S. Fels, "Contexts of Collaborative Musical Experiences," in *Proceedings of the 2003 Conference on New Interfaces for Musical Expression*, A. R. Jensenius and M. J. Lyons, Eds., 2003, pp. 129–134.

[33] K. Jennings, "'Toy Symphony': an International Music Technology Project for Children," *Music Education International*, vol. 2, pp. 3–21, 2003.

[34] R. Aimi. "Music shapers in an indoor garden." (Nov. 23, 2005), [Online]. Available: `https://opera.media.mit.edu/papers/shapers-mf-paper.pdf` (visited on 07/31/2023).

[35] G. Weinberg and S. Driscoll, ""iltur" - Connecting Novices and Experts Through Collaborative Improvisation," in *Proceedings of the 2005 Conference on New Interfaces for Musical Expression*, 2005, pp. 17–22.

[36] "Welcome to the Brain Opera." Archived from the original. (Apr. 26, 2006), [Online]. Available: `https://web.archive.org/web/20060426045519/http://brainop.media.mit.edu/indexold.html` (visited on 07/18/2023).

[37] J. Ippolito, "Review Essay: Brain Opera," *Postmodern Culture*, vol. 7, no. 1, 1996.

[38] "BO Tech Systems Index." (Dec. 2, 1998), [Online]. Available: `http://paradiso.media.mit.edu/TTT.BO/index.html` (visited on 07/18/2023).

[39] B. Ullmer and H. Ishii, "Emerging Frameworks for Tangible User Interfaces," *IBM Systems Journal*, vol. 39, no. 3.4, pp. 915–931, 2000.

[40] M. Kaltenbrunner, S. Jorda, G. Geiger, and M. Alonso, "The reacTable*: A Collaborative Musical Instrument," in *Proceedings of the 15th IEEE International Workshops on Enabling Technologies*, 2006.

[41] F. Calegario, J. Tragtenberg, J. Wang, I. Franco, E. Meneses, and M. M. Wanderley, "Open Source DMIs: Towards a Replication Certification for Online Shared Projects of Digital Musical Instruments," in *Proceedings of the HCI International Conference*, Cham, 2020, pp. 84–97.

[42] P. R. Cook, "Re-Designing Principles for Computer Music Controllers: A Case Study of SqueezeVox Maggie," in *Proceedings of the 2009 Conference on New Interfaces for Musical Expression*, 2009, pp. 218–221.

[43] F. Calegario, M. M. Wanderley, S. Huot, G. Cabral, and G. Ramalho, "A Method and Toolkit for Digital Musical Instruments: Generating Ideas and Prototypes," *IEEE Multimedia*, vol. 24, no. 1, pp. 63–71, 2017.

[44] T. Fukuda and M. M. Wanderley, "T-Patch: A Software Application for T-Stick Digital Musical Instruments," in *Proceedings of the 2023 Conference on New Interfaces for Musical Expression*, 2023.

[45] R. Masu, A. P. Melbye, J. Sullivan, and A. R. Jensenius, "NIME and the Environment: Toward a More Sustainable NIME Practice," in *Proceedings of the 2021 Conference on New Interfaces for Musical Expression*, 2021.

[46] G. Akovali, "2 - Plastic materials: polyvinyl chloride (PVC)," in *Toxicity of Building Materials*, F. Pacheco-Torgal, S. Jalali, and A. Fucic, Eds., Cambridge, U.K.: Woodhead Publishing Limited, 2012, pp. 23–53.

[47] D. A. Stewart, "Vigorous Music-Making: The Inherent "Liveliness" Of A T-Stick Instrumentalist," in *Proceedings of the 2010 International Computer Music Conference*, 2010.

[48] E. A. Meneses, T. Fukuda, and M. M. Wanderley, "Expanding and Embedding a High-Level Gesture Vocabulary for Digital and Augmented Musical Instruments," in *Proceedings of the International Conference on Human-Computer Interaction*, 2020, pp. 375–384.

[49] E. A. Meneses, S. Freire, and M. M. Wanderley, "GuitarAMI and GuiaRT: Two Independent Yet Complementary Augmented Nylon Guitar Projects," in *Proceedings of the 2018 Conference on New Interfaces for Musical Expression*, 2018.

[50] *WS2812 Integrated Light Source*, V1.0, WorldSemi, 2020. [Online]. Available: `https://www.mouser.com/pdfDocs/WS2812B-2020_V10_EN_181106150240761.pdf`.

[51] *Raspberry Pi 4 Model B*, Release 1, Raspberry Pi (Trading) Ltd., 2019. [Online]. Available: `https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf`.

[52] *Nonstandard for transmission of IP datagrams over serial lines: SLIP*, RFC 1055, Jun. 1988. [Online]. Available: `https://www.rfc-editor.org/info/rfc1055`.

[53] D. Brand and P. Zafiropulo, "On Communicating Finite-State Machines," *Journal of the ACM*, vol. 30, no. 2, pp. 323–342, 1983.

[54] D. Beazley, "Understanding the Python GIL," in *Proceedings of the PyCON Python Conference*, 2010.

[55] "PEP 703 – Making the Global Interpreter Lock Optional in CPython." (Jan. 9, 2023), [Online]. Available: `https://peps.python.org/pep-0703/` (visited on 08/01/2023).

[56] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The Many Faces of Publish/Subscribe," *ACM Computing Surveys*, vol. 35, no. 2, pp. 114–131, 2003.

[57] A. Marquez-Borbon and J. P. Martinez Avila, "The Problem of DMI Adoption and Longevity: Envisioning a NIME Performance Pedagogy," in *Proceedings of the 2018 Conference on New Interfaces for Musical Expression*, 2018, pp. 190–195.

[58] R. Chang and V. Shmatikov, "Formal Analysis of Authentication in Bluetooth Device Pairing," in *Proceedings of the Foundations of Computer Security and Automated Reasoning for Security Protocol Analysis*, 2007.

[59] M. Battier and N. Schnell, "Introducing Composed Instruments, Technical and Musicological Implications," in *Proceedings of the 2002 Conference on New Instruments for Musical Expression*, 2002.

[60] A.-N. Niyonsenga and M. M. Wanderley, "Tools and Techniques for the Maintenance and Support of Digital Musical Instruments," in *Proceedings of the 2023 Conference on New Interfaces for Musical Expression*, 2023.

[61] B. Boettcher, "Developing Maturity in DMIs and Mapping Tools," M.S. thesis, McGill University, 2023.