

Tiny trainable instruments

by

Aarón Montoya-Moraga

B.S., Pontificia Universidad Católica de Chile (2014)

M.P.S, New York University (2017)

Submitted to the Program of Media Arts and Sciences
in partial fulfillment of the requirements for the degree of

Master of Science in Media Arts and Sciences

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2021

© Massachusetts Institute of Technology 2021. All rights reserved.

Author
Program of Media Arts and Sciences
September 2021

Certified by
Tod Machover
Muriel R. Cooper Professor of Music and Media
Thesis Supervisor

Accepted by
Tod Machover
Academic Head, Program in Media Arts and Sciences

Tiny trainable instruments

by

Aarón Montoya-Moraga

Submitted to the Program of Media Arts and Sciences
on September 2021, in partial fulfillment of the
requirements for the degree of
Master of Science in Media Arts and Sciences

Abstract

How can we build flexible multimedia instruments that we can train instead of program? How can we build and publish our own databases for artistic purposes? Tiny trainable instruments is a collection of instruments for media arts, using machine learning techniques and deployed in microcontrollers.

This thesis offers a proposal for the use of tiny machine learning for multimedia artistic purposes, with an emphasis on open source and machine learning ethics.

Thesis Supervisor: Tod Machover

Title: Muriel R. Cooper Professor of Music and Media

Acknowledgments

This thesis could not have been possible with the support of these wonderful people, in alphabetical order:

Aaron, Agnis, Aillaly, Alexandra, Alonso, Ana María, Andrea, Andrew, Beatriz, Belén, Benjamín, Bernardita, Braulio, Camila, Camilo, Carla, Carmelo, Carolina, Casey, Catalina, Charles, Christian, Claudia, Corbin, Cynthia, Daniel, Daniella, Devora, Erik, Felipe, Fernanda, Francesca, Francisco, Gabriela, Gaurav, Guillermo, Hannah, Jasmine, Javiera, Jen, Jennifer, Jorge, José, Juan, Julian, Juniper, Karina, Karsten, Kathy, Katya, Lauren, Lily, Lindsey, Lisa, Luna, Mahy, Manaswi, Marco, Margarita, Martin, Maxwell, Mel*, Mitchel, Monica, Namira, Natalia, Natalie, Nathier, Newén, Nicolás, Nicole, Nikhil, Nouf, Nushin, Olivia, Pablo, Patricio, Pedro, Peter, Priscilla, Rafael, Raúl, Rebecca, Rébecca, Renata, Ricardo, Rodrigo, Roy, Russell, Ruta, Sankalp, Sasha, Sayén, Sean, Sebastián, Sejo, Shir, Sofía, Sokio, Soledad, Tiri, TK, Tod, Tom, Verónica, Víctor, Victoria, Viniyata, Will, Wipawe, Yuan, Yuli, Yusuf, Zach

Contents

1	Introduction	12
1.1	Context	12
1.2	Objectives	14
1.3	Outline	15
2	Tiny trainable instruments	17
2.1	Design principles	17
2.1.1	Cheap	18
2.2	Antidisciplinarity	18
2.3	Technology	18
2.4	Open	19
2.5	Philosophy and experience	19
2.6	Inputs	19

2.6.1	Color	19
2.6.2	Gesture	20
2.6.3	Speech	20
2.7	Outputs	20
2.7.1	Buzzer	21
2.7.2	Servo motor	21
2.7.3	MIDI	21
2.7.4	Thermal printer	21
2.8	Development	22
2.9	Code	22
2.9.1	src/	23
2.10	Opera of the Future projects	24
2.10.1	Squishies, by Hannah Lienhard	24
2.10.2	Fluid Music, by Charles Holbrow	24
3	Early experiments	25
3.1	Microcontrollers	25
3.2	Machine learning	28
3.3	Interfaces and instruments	29

4	Background and inspiration	30
4.1	Research at MIT	30
4.1.1	Classes at MIT	30
4.1.2	Projects at MIT	31
4.2	Computational media arts instruments	32
4.2.1	Bastl Instruments	33
4.2.2	Critter & Guitari	37
4.2.3	monome	38
4.2.4	Shbobo	40
4.3	Education	41
4.4	Creative machine learning	42
4.5	Digital rights	44
4.6	Education	45
4.7	Machine learning	45
4.8	Digital rights	45
5	Project evaluation	47
5.1	Overview	47
5.2	Digital release	48

5.3	Workshop	48
5.4	Multimedia documentation	50
6	Conclusions	53
6.1	Contributions	54
6.2	Lessons learned	54
6.3	Future work	55
6.3.1	Hardware for new instruments	55
6.3.2	Software for new instruments	56
6.3.3	Educational impact	57
A	Context	58
A.1	Software	59
A.2	Hardware	59
A.3	Collaborators	59
B	Scripts	61
B.1	Formatting code with clang-format	61
B.2	Converting formats with ffmpeg	62
B.3	Deleting metadata with exiftool	64

B.4 Converting formats with pandoc 66

List of Figures

4-1	SiguesAhi project	32
4-2	Bastl Instruments Servo module	34
4-3	Bastl Instruments microGranny 2	34
4-4	Bastl Instruments Kastle v1.5	35
4-5	Bastl Instruments Kastle Drum	35
4-6	Bastl Instruments Illuminati	36
4-7	Bastl Instruments OMSynth	36
4-8	Critter & Guitari Kaleidoloop	37
4-9	Critter & Guitari Organelle M	38
4-10	Critter & Guitari EYESY	38
4-11	monome grid	39
4-12	monome aleph	39
4-13	monome norns	40

4-14	Shbobo Shnth	40
4-15	Shbobo Shtar	41
4-16	Sam Lavigne, Training Poses, 2018	43
5-1	Workshop packages	48
5-2	Workshop flyer cover in English	50
5-3	Workshop flyer cover in Spanish	51
5-4	Workshop multimedia output in English	51
5-5	Workshop multimedia output in Spanish	52

List of Tables

4.1	Technical details of media arts instruments	33
4.2	Influence of media arts instruments	33

Chapter 1

Introduction

Me caí, me paré, caminé, me subí
me fui contra la corriente
y también me perdí

Soy Yo

Bomba Estéreo, 2015

1.1 Context

This thesis is the capstone project of my master's program, between the academic years 2019-2021 at MIT Media Lab, where I am a Research Assistant at the groups Opera of the Future and Future Sketches. The work presented here has been developed mostly working remotely during the COVID19 pandemic, while at home in Boston MA.

This thesis is a collection of media arts instruments, using tiny machine learning, and with a strong emphasis on digital ethics and Do-It-Yourself (DIY). Its main audience is beginners and artists, and it is my hope that this work can inform the discourse

and practice of a new generation of artists, designers, educators, programmers, policy makers, activists, and enthusiasts.

Machine learning can be really difficult and often relies on proprietary software and hardware. They often need to be trained in borrowed and rented resources, such as the service Google Colab, which allows you to share and train ML models on the cloud. This thesis shows users how to build machine learning art systems that can circumvent this and allow for more control of their data.

Artists have unprecedented access to new tools for making new tools and instruments, and this thesis intends to be a foundation for a new generation of instruments for manipulating audiovisual material, using machine learning. This approach is really exciting because it allows beginners and artists to train their instruments instead of programming them, by inputting data for tuning, instead of having to write lines of code and fixed thresholds for changing their behavior.

Machine learning algorithms are written by humans, and then trained on biased data, and they are deployed to the world, where they affect our lives. These imperfections are highlighted over the course of this thesis, by being explicit about the assumptions and quantizations performed when working with data.

TODO: Add example of biased dataset that is commonly used.

The proliferation of surveillance tools and now of microcontrollers allows for an even more pervasive surveillance and data leaks by governments and corporations. This thesis allows for beginners and artists to develop their own databases, by self sensing and surveillance. TODO: explain self-sensing and surveillance, and explain harms of surveillance, with an example.

Low power and repurposable Tiny machine learning is defined as machine learning performed with low-power devices TODO: add citation or mention that it is my own definition. The proposal of media arts instruments that can run on little power,

rechargeable batteries is a friendly use of resources for experimentation in arts. This is in contrast with the high power use and critique of other emerging fields such as Non Fungible Tokens (NFTs) and cryptoart. The proposal of these scriptable open source instruments allows for users to continuously tweak and modify their instruments, repurpose the hardware, and enable sharing.

TODO: Add a reference to an early tiny ML paper.

1.2 Objectives

With the release of TensorFlow Lite Micro, the TinyML Foundation, new avenues have been opened for creative expression using machine learning in microcontrollers.

TensorFlow is a library by Google for machine learning. They have released different flavors for different devices, including TensorFlow.js for working with KavaScript and in the browser, and TensorFlow Lite for devices with less memory, like mobile devices. The newer TensorFlow Lite Micro is even lighter and intended for working on microcontrollers, and that is the flavor I am using on this thesis.

in a paragraph here that combines the points you make in the first paragraphs -> the main contributions of YOUR thesis... which it seems is basically just addressing a bunch of the context section? Blakeley Payne's thesis does a good job of outlining contributions. It's also good to do here to understand where you are going

Homebrew examples, teach people how to build their own databases, and how to circumvent corporations view,

TODO: Terms and conditions comic book. If people were to read terms and conditions, it would take them X years

You can train an instrument to only detect your voice, your accent, your living con-

ditions.

TODO: add blurb main inspiration about trail building and skatepark, and playpens, sandbox

TODO: add Marina Berry's paper on playgrounds, child development

TODO: Technology and Public Art with Rafael Lozano-Hemmer, June 3, 2020

<https://www.youtube.com/watch?v=QgVdEmqmuEE> 36m28s Rafael Lozano-Hemmer: "Face recognition needs to be banned in all applications except art."

TODO: add Zoom example of garbage speech to text

TODO: Add contextual stories to frame each aspect TODO: Raspberry Pi example, it's cheap but it needs a lot of extra hardware to be used TODO: Also add examples about April Fools Day and Terms and Conditions, about owning your soul TODO: link <https://www.youtube.com/watch?v=jOQ-9S3lOnM&t=125s> TODO: Terms and conditions citation: (copied this in from my thesis, but check out McDonald and Cranor 2008) <https://kb.osu.edu/handle/1811/72839> "Just considering privacy policies shown to people online in one year, it's been estimated that consumers would need approximately 244 hours to read, not skim, privacy policies shown to them in one calendar year."

1.3 Outline

This thesis will cover the following chapters:

1. Chapter 2: Background: the inspiration and context of this thesis. TODO: add literature that informs my work.

2. Chapter 3: Early experiments: my earlier work that led to this thesis, in the topics of media arts education, microcontrollers, and machine learning.
3. Chapter 4: Tiny Trainable Instruments: description of design strategies for the software and hardware, description of the support team working on this thesis.
4. Chapter 5: Project evaluation: user feedback, field notes.
5. Chapter 6: Future work: next iterations of the instruments, and their proposed use for educators and artists.

Chapter 2

Tiny trainable instruments

All the modern things
have always existed
they've just been waiting

The Modern Things
Björk, 1995

This chapter describes what this project is, its design principles, and scope of this thesis.

2.1 Design principles

1. Affordable
2. Hackable
3. Open
4. Private

2.1.1 Cheap

The materials for this thesis

2.2 Antidisciplinarity

This project draws from different disciplines, all intertwined and in different capacities, including:

1. Artificial intelligence
2. Electrical engineering
3. Computer science
4. Media arts
5. Music

2.3 Technology

This project is built with microcontrollers and

This project is based on the Arduino Nano 33 BLE Sense microcontroller, and its software dependencies include the Arduino KNN library, the Arduino TensorFlow Lite Micro library, and Adafruit software libraries.

2.4 Open

All examples included with this library were written with the aim of showing the fundamentals of how to build the instruments and different machine learning enabled manipulation of multimedia material, so that people could build on top of it and make it their own, by changing the values of variables and adding more functionalities.

2.5 Philosophy and experience

Throughout this project, the magic number was 3. The machine learning algorithms were hardcoded to be able to distinguish between 3 different categories: 3 colors, 3 physical gestures, 3 sound utterances.

2.6 Inputs

We are using the RGB color, proximity, gyroscope, accelerometer, and microphone sensors on the microcontroller, in order to capture the Inputs

1. Color
2. Gesture
3. Speech

2.6.1 Color

This approach uses the RGB color sensor from the microcontroller, with the auxiliary help from the proximity sensor, that is used to capture color information at a certain

distance threshold.

The data is passed to a k-Nearest-Neighbor algorithm, programmed using the Arduino KNN library.

2.6.2 Gesture

This input uses the information from the Inertial Measurement Unit (IMU) of the microcontroller, including a gyroscope and accelerometer. It captures data after a certain threshold of movement is detected.

The data is passed to a TensorFlow neural network, programmed using the Arduino TensorFlow Lite library, and based on the included magic_wand example.

2.6.3 Speech

This input uses the information from the microphone of the microcontroller.

The data is passed to a TensorFlow neural network, programmed using the Arduino TensorFlow Lite library, and based on the included micro_speech example.

2.7 Outputs

The different outputs were picked, because of their low cost, ubiquity, and possibilities of expansion and combining them.

2.7.1 Buzzer

This output creates pitched sound, by using a PWM output.

2.7.2 Servo motor

This output creates movement and through that, rhythmic sounds.

The main inspiration for this output was the emerging use of motor-activated percussive instruments, such as the Polyend Perc.

2.7.3 MIDI

We wrote functionalities to manipulate MIDI instruments, and included examples to interface with some popular and cheap MIDI instruments, such as the Korg volca beats.

We included examples for rhythmic and melodic elements, using two very ubiquitous and inexpensive MIDI musical instruments, which are the Korg volca beats, and the Korg volca keys.

2.7.4 Thermal printer

A thermal printer is the basis for creating written and literary output, inspired by the field of computational poetry.

We used the popular Adafruit Thermal printer kit, which is documented on their website and includes a software library, distributed over GitHub and Arduino IDE, and also as a submodule on this project's TinyTrainable software library.

2.8 Development

This thesis has been developed with the invaluable help of undergrad researchers Peter Tone and Maxwell Wang.

They have cloned both repositories, the main one and the Arduino library one, and have continuously submitted pull requests with their contributions.

Peter Tone has helped with research in data structures, library writing, and we have shared back and forth code, going from experimental proofs of concepts, and has also helped with the design of the user-facing library.

Maxwell Wang has proofread the code, has run the examples, and has helped with the writing of the documentation for self-learners and for the workshops.

We all share a Google Drive folder, where we all share notes about our research and development of the library and the educational material.

2.9 Code

This thesis is distributed as a repository, hosted on the GitHub platform, and available at <https://github.com/montoyamoraga/tiny-trainable-instruments>.

The auxiliary files, such as the LaTeX project for this document, and the auxiliary Jupyter notebooks, and documentation and tutorials are included on this repository.

The main software component of this project is the TinyTrainable library, available at <https://github.com/montoyamoraga/TinyTrainable> and also through the Arduino IDE.

The code included on this library is distributed on the folders:

1. examples/
2. src/

2.9.1 src/

The source code for where there is a TinyTrainable.h and TinyTrainable.cpp file where we included all the basic functionality of the library. Additional subfolders include

inputs/

Base class Input and inherited classes for each one of the other inputs.

outputs/

Base class Output and inherited classes for each one of the other outputs.

tensorflow/

Auxiliary files, copied from the examples from the Arduino TensorFlow Lite that we are building on top of, and also from the newer TinyML library by the EdX team. These, unless otherwise noted, are included without modifications and distributed through the Apache License included on each file's headers.

2.10 Opera of the Future projects

During the development of this thesis, I have been fortunate to collaborate on different capacities with other thesis projects by classmates at Opera of the Future, which has directly inspired my work.

2.10.1 Squishies, by Hannah Lienhard

Squishies is Hannah Lienhard’s master’s thesis, and consists of novel squishable interfaces for musical expression. We shared discussions about low-level sound design, code reusability, sound art education, and digital instruments. We were part of a master’s thesis working group, facilitated by Roya Moussapour with two other Media Lab classmates, where we workshopped drafts of our thesis. This practice and feedback has been critical in shaping the language and discourse of this thesis document.

2.10.2 Fluid Music, by Charles Holbrow

Fluid Music is Charles Holbrow’s PhD thesis. It is a library for library design, documentation for contributors. The design of the interface, documentation, and scope of the thesis were a direct influence on the documentation and API coding style of this project.

Overall thoughts: I think it would be great if your Chapter 3 clearly led you to the design principles you lay out in Chapter 4. So, why is “open” important -> because of the collaborations or arduino libraries you used. Why is “cheap”/“privacy” -> coded bias and ability to tinker, etc.

Chapter 3

Early experiments

And you may ask yourself
"Well... how did I get here?"

Once in a Lifetime
Talking Heads, 1981

In this chapter I showcase my journey, experience, pitfalls, breakthroughs and what led me to working on this thesis, and I will try to be open, detailed, celebratory, and critical.

3.1 Microcontrollers

I learned how to program microcontrollers around 2010 as an undergraduate student of electrical engineering in Chile, using PIC microcontrollers, C# and Windows machines. In parallel, with some classmates we discovered the Arduino microcontrollers, and with my friend Braulio we built a robotic guitar tuner, where the Arduino where the Arduino detected the pitch of a string, and then in turn controlled a motor to

move the tuning gear to match the desired pitch.

Fast forward to thesis in 2013, we were expected to implement many low-level programming techniques for our capstone undergraduate thesis, so Arduinos were not allowed because they were considered a shortcut. With my friend Guillermo we built a robotic device using a cumbersome setup, with an Apple computer barely running Windows, in order to write and compile C# code and writing code that was very specific to that particular chip and hardly reusable or interesting for a wider audience.

Since then I never programmed PICs again, but Arduinos became a huge part of my practice. I am grateful about the community's efforts on sharing resources online, and the extensive documentation available.

After graduation I freelanced as a software and technology designer and developer for artists. I learned computer protocols and networks, and wrote custom software for live multimedia performance, theater and music, and software deployed on iPads and cellphones. I discovered Processing, a project that Arduino was based on, and continued learning code on my own. With Processing I was able to learn how to build apps with computer graphics, interactivity, and it quickly became central to my practice and work.

Realizing that I needed a bigger community of people to learn media arts from, I researched communities where I thought I could learn this craft in an focused and immersive way, so I applied to New York University's Interactive Telecommunications Program, where I joined as a graduate student in 2015.

In my first semester I took the amazing class Introduction to Physical Computing, with one of Arduino's co-creators Tom Igoe, and I learned about haptic design, open source hardware and software, and physical computing education.

I was introduced to a wider ecosystem of microcontrollers beyond Arduino, like the

Teensy by PJRC, which captivated me by its USB MIDI capabilities, which allowed for standalone MIDI operation, and the creation of plug and play devices that needed no additional setup or scripting. Also by its audio library, which allowed me to create interactive standalone experiences, playing audio samples and applying audio effects.

At NYU ITP I slowly learned about hardware, but mostly about web and scripting, and my thesis concentrated on open source, performance art, with only a small hardware component in the form of a Raspberry Pi computer with a countdown timer to my projected death time, according to data by the United Nations, based on my assigned-at-birth-gender and my birth place.

After graduating from NYU ITP I focused on media arts education, writing tutorials, teaching introduction to programming workshops for artists.

When I joined MIT Media Lab in 2019, I made the conscious decision of focusing on hardware, to give my creations a life outside of my computer, also inspired by newer restrictive developments by Apple, such as restricting the use of apps created by unregistered developers. In my first semester, which ended up being the only on-campus semester I had, I was introduced by my friend Will Freudenheim to the Shbobo synthesizers by Peter Blasser.

I was partially aware of the instruments made by Peter Blasser, in particular the analog ones.

While at MIT Media Lab, I was delighted by the newer versions of Teensy, which are even faster and more powerful, and which led me to start designing handheld samplers for field recordings, and other standalone devices.

This in turn led me to review the current NYU ITP materials for physical computing, where they currently stopped using the now classic Arduino Uno, and have incorporated in their teaching the new series of Arduino Nano microcontrollers, which I based my thesis on.

In particular, the Arduino Nano 33 BLE Sense I am using, comes with 9 sensors, to measure and detect acceleration, movement, distance, color, and a microphone. This is an amazing breakthrough, since now we can use all this data without having to purchase, install, or calibrate the sensors.

I am using the Arduino's sensors to gather multimedia input data, analyze it with machine learning, and then responding with multimedia outputs.

3.2 Machine learning

My first experiment with creative machine learning, was with my NYU ITP classmate Corbin Ordell, who was a student at Gene Kogan's machine learning class, and we teamed up to hack a project we called Piano Die Hard, built with open source tools such as Wekinator, Arduino, openFrameworks, and using the machine learning algorithm KNN. We created a video database of explosions in the Die Hard movie franchise, and another one of other 1980's movies with no explosions, and we trained our machine learning algorithm to distinguish between the categories explosion and no explosion. We featured this project at a NYU ITP show, were written up at the Daily Beast newspaper, and exhibited our work at the alt-ai conference.

In 2017, while I was finishing my appointment as research resident at NYU ITP, Cristóbal Valenzuela had started the project RunwayML as his master's thesis, which is now a company led by Cristóbal, Alejandro Matamala and Anastasis Germanidis.

At NYU ITP I also saw the first experiments with `deeplearn.js`, later `TensorFlow.js`, which soon became the foundation of the `ml5.js` library, a wrapper for `TensorFlow.js`, for on-the-browser machine learning.

I decided I wanted to dip my toes in machine learning, so I took a month-long intensive class at the School of Machines in Berlin, Germany, facilitated by Gene Kogan and

Andreas Refsgaard, and organized by Rachel Uwa.

A big inspiration for this thesis has been the book on GANs by Casey Reas, published by Anteism, as of 2021 on its second edition. It's an arts-first book that contextualizes the use of machine learning algorithms for the creation of images, and uses the metaphor of these algorithms as being similar to the development of the camera. Artists don't need to understand all the physics or mechanics behind a camera in order to make art with it, but it can help to understand it too. I think machine learning is also a game changer for instrument making, and machine learning introduces new civic complexities, and my thesis tries to follow the example of this book, to introduce the technology and contextualize for a new generation of artists and instrument makers.

Since many machine learning projects rely on proprietary hardware, such as NVIDIA GPUs, or rely on the cloud for faster compilation times, for this thesis I decided to make open source machine learning projects that people could read and understand and remix and hack.

TODO: mention the impact of the documentary Coded Bias, and how these researchers impacted my desire to make my thesis. Also mention how right before pandemic I had started a pottery class, with the intention of making clay-based instruments for thesis, as a metaphor of making code and hardware and software feel fluid and not static, I want to empower people to program, in particular machine learning because of its dangerous implementations by oppressive governments and corporations, and in particular for arts, for making artists dream come true.

3.3 Interfaces and instruments

I am an enthusiast of new interfaces and instruments for music and media art production.

Chapter 4

Background and inspiration

It has to start somewhere
it has to start sometime
what better place than here?
what better time than now?

Guerrilla Radio

Rage Against the Machine, 1999

4.1 Research at MIT

As part of the research that directly inspired this thesis, here are some courses, projects, and studies I have undertaken while at MIT Media Lab.

4.1.1 Classes at MIT

1. Fall 2019, CMS.901 Current Debates in Media, by professor Sasha Costanza-Chock

2. Fall 2019, MAS.S65 Recreating The Past, by professor Zach Lieberman
3. Spring 2020, MAS.826 Sound: Past and Future, by Tod Machover
4. Spring 2020, MAS.712 Learning Creative Learning, by professor Mitchel Resnick

In the class Current Debates in Media, topics covered included fake news, surveillance, algorithmic bias, data colonialism, climate justice, algorithms of oppression, among others. For my final paper I wrote on the role of the media during the 2019 Chilean protests.

In the class Recreating the Past, I learned about media arts history, and I furthered my learning of the language C++ which I ended up using for writing the library for my thesis, and of the package and community of openFrameworks, which is one of the most popular open source frameworks for media arts.

In the class Sound: Past and Future, I learned more about the history of different computational advancements for sound, with a strong focus on projects at MIT Media Lab's own research groups including Opera of the Future, Hyperinstruments, and Music, Mind, and Machine.

4.1.2 Projects at MIT

Some other projects I created during these years include:

1. SiguesAhi: an instrument to detect when oppressive institutions have ceased to exist. It is achieved with microcontrollers with Internet connectivity.
2. Open Drawing Machine, with Gaurav Patekar: an open source low cost programmable drawing machine

3. Introduction to networks for artists: a series of tutorials for beginners, to learn how to set up their own networks and collaborate in peer-to-peer ways for making art.

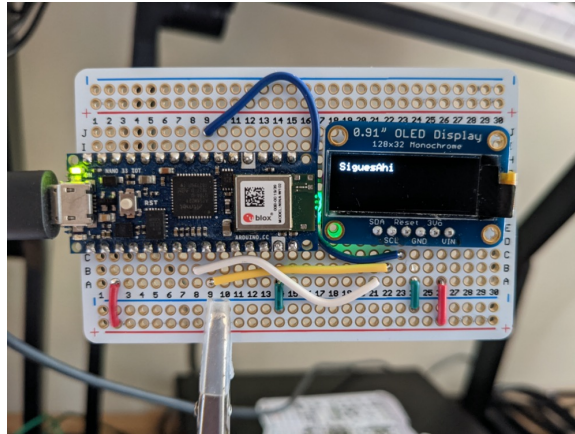


Figure 4-1: SiguesAhi project

Picture taken at home

4.2 Computational media arts instruments

On top of the proliferation of personal portable computers capable of performing real-time audio, and their creative live use by compute

In particular, here I will highlight some media arts instruments that have inspired my research, because of their use and promotion of open source software and hardware, scripting capabilities, and other design considerations. These instruments often sit at my desk for inspiration, or I spend hours playing with them for my art and learning from them and the communities around them.

The tables 4.1 and 4.2 are respectively a technical and influence summary of the instruments that I reference in this section.

Company	Instrument	Year	Computing	Software
Bastl Instruments	Kastle v1.5	2017	microcontroller	Arduino
Bastl Instruments	microGranny 2	2016	microcontroller	Arduino
Bastl Instruments	Servo	2016	microcontroller	Arduino
Critter & Guitari	Organelle	2016	computer	Linux
Critter & Guitari	EYESY	2020	computer	Linux
monome	aleph	2013	computer	Linux
monome	norns	2018	computer	Linux
Shbobo	Shnth	2013	microcontroller	Shlisp
Shbobo	Shtar	2017	microcontroller	Shlisp

Table 4.1: Technical details of media arts instruments

Company	Instrument	Influence
Bastl Instruments	Kastle v1.5	placeholder
Bastl Instruments	microGranny 2	placeholder
Bastl Instruments	Servo	placeholder
Critter & Guitari	Organelle	placeholder
Critter & Guitari	EYESY	placeholder
monome	aleph	placeholder
monome	norns	placeholder
Shbobo	Shnth	placeholder
Shbobo	Shtar	placeholder

Table 4.2: Influence of media arts instruments

4.2.1 Bastl Instruments

Bastl Instruments is a Czech company of multimedia instruments, which has had a huge impact and influence on my research and practice. When I first started researching the Eurorack format some years ago, I visited the shop Control in Brooklyn NY, and some modules by Bastl stood out to me, because of their wooden panels and interaction with classic physical computing educational materials, such as motors and solenoids, which was an inspiration for me to include support for servo motors in Tiny Trainable Instruments.

Another inspiration comes from their microgranny 2 granular sampler which is made with an Atmega microcontroller and its firmware is open source and available as a repository on their GitHub account, along with many other of their instruments.



Figure 4-2: Bastl Instruments Servo module

Retrieved from [3]

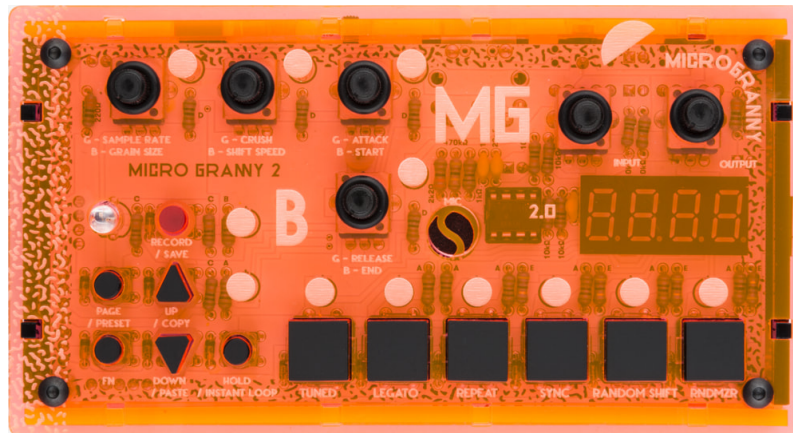


Figure 4-3: Bastl Instruments microGranny 2

Retrieved from [3]

Their Kastle synthesizers are also based on microcontrollers, and feature a patchbay for making connections with jumper wires, the same used for prototyping in electronic breadboards, which influenced me to make the Tiny Trainable Instruments built with breadboards and jumper wires, instead of custom printed circuit boards. Also, the Kastle sythns are forgiving instruments, their inputs and outputs are robust enough to allow for mistakes in connections, in an electrical and mechanical way, which I think it's perfect for safe experimentation, it would be a bummer if the instrument was easy to break, or if it demanded a huge effort in understanding electronics for using it.



Figure 4-4: Bastl Instruments Kastle v1.5

Retrieved from [3]

As of writing, 2 different units are in production, both retailing for 100.00 USD, the Kastle v1.5 melodic / drone synthesizer, and the Kastle Drum, for rhythm. The only difference between these synthesizers is the firmware and the labels on the faceplate. The community is encouraged to write new firmware to modify their behavior.



Figure 4-5: Bastl Instruments Kastle Drum

Retrieved from [3]

Another instrument I want to highlight is the Illuminati, currently discontinued, a device that uses different inputs (audio, control voltage, MIDI messages), to control the light intensity of connected USB lamps, which influenced the conception of Tiny Trainable Instruments as multimedia arts instruments, not only focusing on audio and music, but also printed text, light, and screen output.



Figure 4-6: Bastl Instruments Illuminati

Retrieved from [3]

The final instrument from this company that I want to highlight is the OMSynth, one of many collaborations with Casper Electronics. This device is an educational and maker circuit development tool for creating synthesizers, it includes basic fundamental blocks, such as battery power, audio input and output, potentiometers for attenuating and boosting signals, and a suite of parts kits for building devices including sequencers, oscillators, and samplers, on the included breadboard. Its release as a kit was also a direct influence in the release of Tiny Trainable Instruments as a kit with instructions.

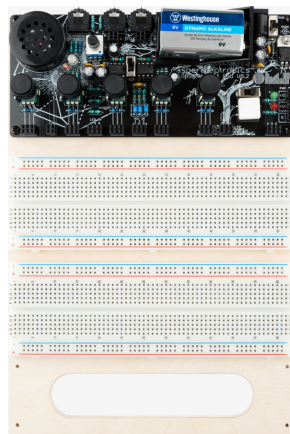


Figure 4-7: Bastl Instruments OMSynth

Retrieved from [3]

Many BASTL standalone instruments are 200.00 USD or less, which is a huge contrast

to the 1960s, when a Moog analog system II cost 6,200.00 USD, which was enough to buy a small house [6]. Also, many of their instruments are sold as kits for building and soldering them yourself, for the cheaper cost and the added educational aspect of having a hands-on experience.

4.2.2 Critter & Guitari

Critter & Guitari is an American company based in Brooklyn NY, which have released in the past computational microcontroller-based audiovisual instruments, from which my favorite is the Kaleidoloop, currently discontinued. It is a sampler with an internal mic and speaker, that allows you to record audio and then control its output with 2 knobs for volume and playback rate. It is designed to be portable for doing field recordings, and it was an influence on the design of the Tiny Trainable Instruments library, which allows the construction of standalone instruments, that with a USB power bank or a battery, you can take for a walk or place anywhere you want.



Figure 4-8: Critter & Guitari Kaleidoloop

Retrieved from [1]

This company has released standalone scriptable computers for arts, which run Linux operating system + Pure Data software.

Organelle computer for sound, scriptable, Linux operating system + Pure Data soft-

ware.



Figure 4-9: Critter & Guitari Organelle M

Retrieved from [2]

ETC and EYESY computers for visuals, scriptable, Linux operating system + Python / pygame environment or openFrameworks.



Figure 4-10: Critter & Guitari EYESY

Retrieved from [2]

They can run on power supplies, and are also portable by the use of batteries.

4.2.3 monome

Aleph: earlier sound computer.

Norns: sound computer, currently on its second iteration, with expanded hard drive.

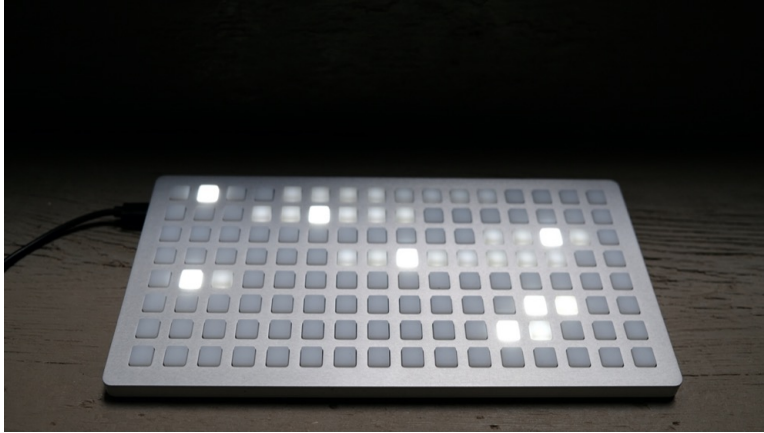


Figure 4-11: monome grid

Retrieved from [5]

Also there is a DIY version which is cheaper and runs on a Raspberry Pi. Norns is a Linux machine, running SuperCollider for the sound engine, and Lua scripts.



Figure 4-12: monome aleph

Retrieved from [5]

Norns: sound computer, currently on its second iteration, with expanded hard drive. Also there is a DIY version which is cheaper and runs on a Raspberry Pi.

Norns is a Linux machine, running SuperCollider for the sound engine, and Lua scripts. It has spawned a community that continually releases new scripts and software updates.



Figure 4-13: monome norns

Retrieved from [5]

4.2.4 Shbobo

Peter Blasser has released several collections / companies of musical instruments, the most famous one being Ciat-Lonbarde. Peter also runs Shbobo, which to date has two different instruments, the Shnth and the Shtar.



Figure 4-14: Shbobo Shnth

Retrieved from [7]

Both run on microcontrollers, and they use a new proposed language called Shlisp, based on Lisp, and also they can be programmed using the Fish IDE.

As of 2021, their firmware and editor became open source, and it is available at



Figure 4-15: Shbobo Shtar

Retrieved from [7]

github.com/pblasser/shbobo.

COMMENT ON OPEN SOURCE: something being open source doesn't necessarily mean it is accessible to a wider audience. is one of the goals of your work to create instruments that are accessible to a wider audience?

They promote computer-centric approaches to making sound, such as the use of integers and metaphors of finite state machines, and also allow for different ways of playing and sensing, such as the use of antennas for detecting hand distance, a microphone for detecting speech and whistling, and wooden bars with piezos for detecting pressure.

TODO: write how this inspired the new interactions i am inventing or appropriating for Tiny Trainable Instruments, such as a drum machine you can talk to, Alexa spinoff.

4.3 Education

This thesis is inspired by the work of the research group Lifelong Kindergarten at MIT Media Lab, led by professor Mitchel Resnick. On the book with the same title, he builds on Seymour Papert's work, and proposes that educational projects should have "Low floor, wide walls, high ceilings", and that learners thrive when they engage in the 4 Ps: "Peers, projects, passion, play".

In terms of peers, I have been lucky to have been supported by the MIT UROP office and MIT Media Lab, and had the opportunity to work with MIT undergrad researchers Peter Tone and Maxwell Wang. Also, this project was taught in collaborative workshops where people could discuss their ideas with their peers.

In terms of projects, this thesis includes the release of a software library, so that people can make the software their own, and spin-off their own projects. It is also open source so that people can learn from my mistakes and also fork to adapt to their needs.

In terms of passion, this thesis is

In terms of play, this thesis project is not about correct answers, or even excellent classification with machine learning, it's all about finding innovative ways to interact with multimedia material, celebrate the small victories and the big glitches, and iterate over and over again.

4.4 Creative machine learning

COMMENT: what is the main argument of the whole piece and how does each independent part connect to that? you should start with a story from previous experiences that is particularly relevant as to why you were inspired to do this work. think "papert and the gears"

While being a graduate student and research resident at NYU ITP I saw how quick things changed in terms of machine learning. I saw how the project `deeplearn.js` allowed for people to train and deploy machine learning on their browsers, and how this library was acquired by Google and repurposed as `TensorFlow.js`, a JavaScript version of their machine learning framework `TensorFlow`.

In turn, at NYU ITP a team of artists and programmers built the library and community of ml5.js, with the 5 being an homage to p5.js. Technically, ml5.js is a wrapper for TensorFlow.js, in the same spirit that p5.js is a wrapper for HTML5 elements such as the canvas.

Another huge contribution to the landscape of machine learning for arts has been the release of the app Runway, which started as Cristóbal Valenzuela’s thesis, and is now a company with Alejandro Matamala and Anastasis Germanidis.

After leaving NYU ITP in 2018, I was a student at the month-long workshop “Autonomous Generative Spirit” taught by Gene Kogan and Andreas Refsgaard at the School of Machines, Make and Make Believe in Berlin 2018. We experimented with quick and cheap methods for machine learning, such as the k-nearest neighbors algorithm using the artist and beginner-friendly app Wekinator by Rebecca Fiebrink, and also more computer-intensive algorithms, which sometimes required proprietary hardware such as NVIDIA graphics cards to be trained in a matter of hours, instead of days or weeks using our computers.

There is a tradeoff between speed and cost, and also between monetary cost and time cost.



Figure 4-16: Sam Lavigne, Training Poses, 2018

Retrieved from [4]

The last spark that led me to this thesis was the release of 2 libraries for machine learning on the Arduino platform: The currently beta version Arduino KNN, which allows for on-device training and resembles my earlier studies with Wekinator, in a more portable and private way, no data leaves the microcontroller, and the whole neural network can be wiped with one click of a button.

At a more complex level, I am also working with the TensorFlow Lite Micro, which I learned from Arduino blogs, and which currently is supported by the hardware Arduino Nano 33 BLE Sense.

COMMENT TO THE ABOVE PARAGRAPH: you may not even need to include the specific details here, but just highlight in larger overviews the types of projects you’ve worked on or the educational fields that influence your work

In late 2020 and early 2021 I completed the just released series of 3 courses of the TinyML Professional Certificate by Harvard at the online platform edx.org

Newer books and references that this thesis was inspired by include the books “You Look Like a Thing and I Love You: How Artificial Intelligence Works and Why It’s Making the World a Weirder Place” by Janelle Shane (2019), and the book “Making Pictures with Generative Adversarial Networks” by Casey Reas (2019).

Also Yining Shi created a new class at NYU ITP in 2020, at the intersection of machine learning and physical computing.

4.5 Digital rights

Machine learning algorithms need data to be trained on. I think it’s a human right to not be surveilled, and I hope my thesis can put a positive spin on the gathering of data, by letting users perform auto surveillance, like the Ai Weiwei piece WeiweiCam,

a 2021 project where the artist installed cameras for self surveillance as a protest against the Chinese government.

A huge inspiration for my thesis has been the Guardian Project by the Electronic Frontier Foundation, and the research and activism work by Sasha Sasha Costanza-Chock.

4.6 Education

Mitch Resnick's book Lifelong Kindergarten

Low floor, wide walls, high ceiling

Peers, projects, passion, play

Gene Kogan and Andreas Refsgaard

4.7 Machine learning

ml5.js

Runway

TinyML Professional Certificate HarvardX

4.8 Digital rights

Electronic Frontier Foundation

Edward Snowden

Design Justice Network

Chapter 5

Project evaluation

Yeah you wanted a hit
but tell me
where's the point in it?

You Wanted a Hit
LCD Soundsystem, 2010

5.1 Overview

This thesis lives as a PDF document at the MIT library, as a software library with examples for Arduino, and as repositories on GitHub with all the source code and history of the project.

5.2 Digital release

This thesis project lives on 2 different repositories, hosted on GitHub, to foster collaboration via issues and pull requests, and also with frequent updates, to show people the whole process behind this thesis project.

The main repository contains this thesis document, Jupyter notebooks for machine learning, documentation for beginners and educators, and auxiliary shell scripts. It is hosted at <https://github.com/montoyamoraga/tiny-trainable-instruments>.

The auxiliary repository contains the Arduino library, including its source code and examples. It is hosted at <https://github.com/montoyamoraga/TinyTrainable>.

5.3 Workshop



Figure 5-1: Workshop packages

Picture taken at home

For user testing and sharing this thesis, some workshops were conducted during TODO, with support from a grant at CAMIT for teaching the workshops in English in USA, and in Chile in Spanish, remotely over teleconferencing software and after being approved by the MIT COUHES TODO explain.

The workshop instructions are documented on the docs/ folder of the repository available at <https://github.com/montoyamoraga/tiny-trainable-instruments>

Each workshop consists of 2 sessions of 2 hours each, spread over two consecutive days.

In the first session we will first help people with installation of the software, and then move on to start wiring the materials on the electronic breadboard material. We will concentrate on the simpler examples with color input. We will also collect data of gesture and speech to create custom databases and use them to train other slow machine learning models, that will keep on running on the student's workshops after the workshop is over.

On the second session we will use the result of the trained models to create more advanced instruments that react to gesture and speech. We will also show the participants the other

I applied to and was awarded a grant by the Council for the arts at MIT (CAMIT), which consisted of 2,000.00 USD to buy materials to teach the Tiny Trainable Instruments workshop to 20 people, during June 2021.

I taught this workshop 3 times, 2 of them were in English for people based in USA, and 1 time in Spanish for people based in Chile, with a total of 20 kits being delivered.

The multimedia aspect of this project was featured, in particular the ability to use different inputs, including color, gesture and speech, to control different outputs, including serial messages, sound, and motor movement.

FREE online workshops
for artists and beginners

tiny trainable instruments

Learn how to build your own artisanal,
low-fidelity multimedia instruments with
machine learning and microcontrollers

June 22 - 23 | June 25 - 26



Figure 5-2: Workshop flyer cover in English

By Renata Gauí

5.4 Multimedia documentation

TODO: upload a collection of examples made by people who came to the workshops, featuring the software library and what they learned.

Taller online **gratuito**
para artistas y principiantes

tiny trainable instruments

Aprende a construir tus propios instrumentos
artesanales, multimedia y de baja fidelidad con
aprendizaje de máquinas y microcontroladores

2021 Junio 29 - 30



Figure 5-3: Workshop flyer cover in Spanish

By Renata Gauí

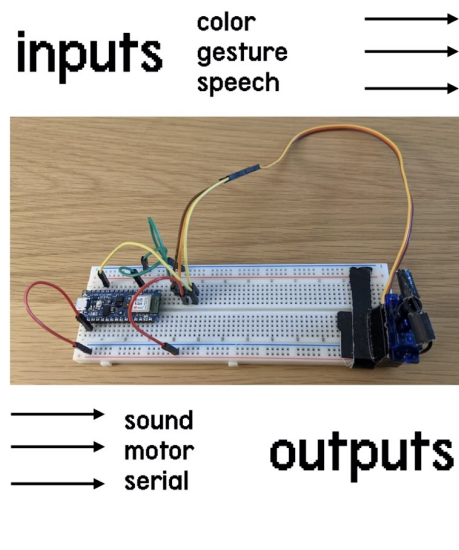


Figure 5-4: Workshop multimedia output in English

By Renata Gauí

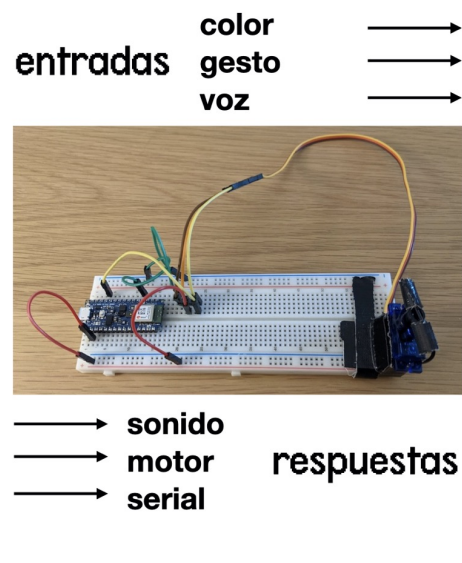


Figure 5-5: Workshop multimedia output in Spanish

By Renata Gaui

Chapter 6

Conclusions

Don't get any big ideas
they're not gonna happen

Nude

Radiohead, 2007

In this thesis I have presented all the stages of the design and development of a software library for creating new standalone multimedia instruments using machine learning and microcontrollers, with a strong focus on AI ethics. The project includes software examples, hardware suggestions, educational material, and strategies for ethical off-cloud machine learning and creation of custom artisanal databases.

This thesis is also the basis for further research, including the creation of subsequent multimedia instruments and software, the writing of new courses and educational units at the intersection of arts, physical computing, interaction design, and computational ethics.

6.1 Contributions

Concrete contributions:

1. Publishing a software library for machine learning with microcontrollers for multimedia art.
2. Design, writing, and teaching of a 4 hour workshop for beginners, enthusiasts, and artists to teach with the software library.
3. Publishing code and tutorials for creating custom databases for gesture and speech.
4. Publishing custom trained models for machine learning.
5. Publishing code and tutorials for training machine learning algorithms on the cloud and on personal computers for privacy.
6. Publishing satellite software libraries, such as `MaquinitasParams` for communication with other instruments, and `MaquinitasRitmos` for rhythmic data.

Abstract contributions:

1. Demonstrating how a broader range of people can use machine learning to support their creative expression." Or do think it's better to focus on more "concrete" descriptions of contributions?

6.2 Lessons learned

1. Writing software for artists is hard, publishing as a library for other artists is even harder.

2. Collaboration with other people is essential to write usable code and educational material.
3. Document all design decisions to explain why and how everything works.

6.3 Future work

6.3.1 Hardware for new instruments

This thesis relies on an Arduino microcontroller because of their open source nature, commercial availability, software and community support, and detailed documentation.

In particular I picked the Arduino Nano 33 BLE Sense, because of 2 main reasons at the time this project started in late 2020: it is the only Arduino supported by TensorFlow Lite for microcontrollers and featured in the HarvardX certificate on Tiny Machine Learning, and also because of its convenience of having embedded sensors, which makes it simpler and cheaper to acquire data for live interaction and for building custom databases, eliminating barriers to instruments makers and prototypers.

Microcontrollers come and go, most probably this board will be discontinued, but the strategies and software can be forked and adapted to other microcontrollers and software architectures. I am particularly looking forward to adapt this thesis project to other open source microcontrollers, including other Arduino boards, PJRC Teensy and Adafruit Circuit Playground, which would enable the adoption of other software stacks, such as Python instead of C++, and also to other communities building multimedia instruments and experiences.

In terms of the outputs of the Tiny Trainable Instruments, I focused on creating many parallel multimedia approaches, including making sounds with piezo buzzers

and MIDI, manipulating light with LEDs, creating movement and rhythm with servo motors, and printing text with thermal printers and screens. This is to appeal to a larger audience of artists and learners, interested in different mediums, and I hope this thesis project inspires more complex artworks and interactive experiences that this library currently allows, and that people can contribute back to the library to share these new capabilities with everyone.

The Tiny Trainable Instruments are built with prototyping electronic breadboards, to make explicit their open-endedness, and to promote experimentation and lower barriers. A further iteration of this project would include the creation of custom printed circuit boards with fixed wiring, and also enclosures and packaging.

6.3.2 Software for new instruments

This thesis has been published as an open source software library for Arduino. It promotes modularity and adaptability, where a Tiny Trainable Instrument can be any combination of the multiple inputs and outputs.

The file structure of the source code and the software dependencies of this library was also built with flexibility in mind, to encourage the remix and adaptation of this library to further projects.

A challenging aspect of this project is the breadth of the disciplines combined, and its novel application of machine learning in microcontrollers. As discussed in previous chapters, there is a trend and new wave of builders and makers creating standalone multimedia instruments, based on open operating systems like Linux, and/or different microcontrollers.

Despite the existence of artists and makers building standalone computational instruments, these skills are still hard to acquire. Additionally, the principles of this project, including being as cheap as possible, and as open as possible, are designed

to encourage experimentation and hacking, but also can pose additional challenges. I hope this project encourages people to learn how to make instruments, and also engages in discourse about the creation of new curricula for the next generation of instrument makers and artists.

Another challenging aspect of writing software for multimedia instruments is its licensing, both choosing a license and also respecting and understanding the license of other code and resources we are using. The dependencies of this software are mostly other libraries by Arduino, Google, Adafruit, and with different licenses including public domain, MIT, and Apache. I hope this document helps to navigate these legal complexities and that this project helps artists and enthusiasts to navigate this landscape and overcome these barriers.

6.3.3 Educational impact

This project was built to inspire and celebrate a new generation of coursework, workshops, books, in the disciplines of ethical machine learning and microcontroller-based instruments.

I hope that this thesis project is adopted by educators, to introduce students to machine learning, physical computing, media arts, and ethics.

Many sections of this project could be adapted to further existing curricula for music, rhythm, ethics, computer science, and to create a new wave of instrument makers and media artists.

I hope to see new classes taught in high school, universities, and cultural centers, at the intersection of multimedia art, physical computing, and ethical machine learning.

Appendix A

Context

You might have to think of
how you got started
sitting in your little room

Little Room
The White Stripes, 2001

Since joining MIT in summer 2019 I didn't leave the USA, and it's the longest stretch I have had of not visiting my home country Chile. This thesis in particular was written between November 2020 and August 2021, mostly in Boston MA USA, while on a F-1 visa.

My native language is Spanish, and this thesis was written in English, using the metric system, and the Gregorian year-month-day format.

A.1 Software

This thesis document was written using LaTeX and the Microsoft Visual Studio Code editor, and then exported to the PDF format.

The TinyTrainable library was written in C++, and packaged as an Arduino library, relying on open source library dependencies by Arduino, Adafruit, and Google.

The auxiliary code is a mix of Python scripts, Jupyter Python notebooks, and shell scripts.

The documentation was written using Markdown.

The workshops were taught using the videoconferencing software Zoom, and organized via Google Forms.

A.2 Hardware

This project was written on a 2017 Macbook Air 13-inch, running macOS Catalina.

The software library and the software examples were written to be deployed on the Arduino Nano 33 BLE Sense microcontroller.

A.3 Collaborators

Priscilla Capistrano is the senior administrative assistant at the Opera of the Future research group and made sure that everything worked.

Peter Tone is a MIT undergraduate student, who was a researcher, designer, pro-

grammer, and tester of the TinyTrainable Arduino library, as part of the MIT UROP program.

Maxwell Wang is a MIT undergraduate student who was a documentation writer, and hardware and software tester, as part of the MIT UROP program.

The Council for the Arts at MIT provided the generous funding of the workshop materials.

Renata Gaui designed and created the flyers for the workshops.

Bernardita Moraga packaged and distributed the materials for the workshop in Chile.

Appendix B

Scripts

By pressing down a special key
it plays a little melody

Pocket calculator

Kraftwerk, 1981

During the writing of this thesis I developed scripts, which are included in this repository on the folder scripts.

I abstracted them to make them useful for other people and published them with a MIT License at <https://github.com/montoyamoraga/scripts>.

Since they are useful scripts and they are commented, I include them here.

B.1 Formatting code with clang-format

clang-format is a command line tool for formatting code. This script was written to auto format the code from the Arduino/C++ library TinyTrainable.

```
echo "formatting with clang"
find "$PWD/../TinyTrainable/src" "$PWD/../TinyTrainable/examples" -iname
    "*.cpp" -o -iname "*.h" -o -iname "*.ino" | while read f
do
    clang-format -i "$f"
    echo "formatted $f"
done
```

B.2 Converting formats with ffmpeg

ffmpeg is a command line tool for converting audiovisual files between formats. This script was written to convert audio files from .mp3 to .ogg format for training a database for speech recognition.

```
# clear command line
"clear"

# directory name
DIR_MEDIA="media"

# extension of original files
EXT_ORIGINAL="mp3"

# extension of desired files
EXT_DESIRE="ogg"

# announce start running script
echo "start running " $PWD/$0
```

```

# check if files/ exists
if [ -d "$DIR_MEDIA" ];

# if files/ exists then
then

    echo "success, $DIR_MEDIA/ exists"

# check if there are .mp3 files in files/
if [ -f $DIR_MEDIA/*.EXT_ORIGINAL ];

# if there are files with $EXTENSION in directory
then

    echo "success, there are matching $EXT_ORIGINAL files"

# iterate over every matching file in directory
for i in $DIR_MEDIA/*.EXT_ORIGINAL;

# pipe the filename into cut
# -d is delimiter of '.'
# -f is the field number, indexed in 1
# it retrieves the filename without the extension
do name='echo "$i" | cut -d'.' -f1'

    echo convert "$i" $EXT_ORIGINAL to $EXT_DESIRE

# ffmpeg conversion
ffmpeg -i "$i" "${name}.${EXT_DESIRE}"

    echo converted "$i" to $EXT_DESIRE

```

```
# delete original file
rm "$i"

echo deleted "$i"

# finish iteration
done

# if there are no matching files in directory
else
    echo "fail, no $EXT_ORIGINAL files in $DIR_MEDIA/"

# end of if statement for matching files
fi

# if directory does not exist
else
    echo "fail, $DIR_MEDIA/ does not exist"

# end of if statement for existence of directory
fi

# announce finished running script
echo "finished running " $PWD/$0
```

B.3 Deleting metadata with exiftool

exiftool is a command line tool for reading and writing metadata from files. This script was written to delete metadata from images, like GPS coordinates added by

modern smartphones, only keeping the actual image.

```
# clear command line
"clear"

# directory name
# DIR_MEDIA="media"
DIR_MEDIA="$PWD/../thesis/images"

# extension of files
EXT_ORIGINAL="jpg"

# announce start running script
echo "start running " $PWD/$0

echo "looking for files with extension $EXT_ORIGINAL in $DIR_MEDIA/"

# check if directory exists
if [ -d "$DIR_MEDIA/" ];

# if directory exists then
then

# announce directory exists
echo "success, $DIR_MEDIA/ exists"

find "$DIR_MEDIA" -iname "*. $EXT_ORIGINAL" | while read f
do
    exiftool -all= -overwrite_original "$f"
    echo "formatted $f"
done
```

```

else
    # announce directory does not exist
    echo "fail, $DIR_MEDIA/ does not exist"

# end of if statemement for existence of directory
fi

# announce finished running script
echo "finished running " $PWD/$0

```

B.4 Converting formats with pandoc

pandoc is a command line tool for converting between formats. This script was written to convert from .tex files to .docx files, so that each chapter of this thesis document could be uploaded to Google Docs for feedback from the committee.

```

echo "pandoc latex to docx"

cd "$PWD/./thesis"

# iterate through all .tex files in thesis/
find "$PWD" -iname "*.tex" | while read f
do
    # retrieve basename
    base=$(basename "$f" .tex)

    # delete original docx file
    rm -f "$PWD/docx/$base.docx"

    # create new docx file with pandoc
    pandoc -s -o "$PWD/docx/$base.docx" "$f"
done

```

```
# do all the files manually  
pandoc -o "$PWD/docx/aaron-thesis.docx" "$PWD/chap1.tex" "$PWD/chap2.tex"  
"$PWD/chap3.tex" "$PWD/chap4.tex" "$PWD/chap5.tex" "$PWD/chap6.tex"  
"$PWD/appa.tex" "$PWD/appb.tex"
```

Bibliography

- [1] Critter & Guitari. *Kaleidoloop*. [Online; accessed 24-July-2021]. 2015. URL: <https://web.archive.org/web/20150206042159/http://www.critterandguitari.com/collections/instruments/products/kaleidoloop>.
- [2] Critter & Guitari. *Official website*. [Online; accessed 24-July-2021]. 2021. URL: <https://www.critterandguitari.com/>.
- [3] Bastl Instruments. *Official website*. [Online; accessed 24-July-2021]. 2021. URL: <https://www.bastl-instruments.com/>.
- [4] Sam Lavigne. *Training Poses*. [Online; accessed 26-July-2021]. 2018. URL: <https://lav.io/projects/training-poses/>.
- [5] Monome. *Official website*. [Online; accessed 24-July-2021]. 2021. URL: <https://www.monome.org/>.
- [6] Trevor Pinch and Frank Trocco. “Analog days: the Invention and impact Of The Moog synthesizer”. In: First Harvard University Press paperback edition. Harvard University Press, 2004. Chap. 3: Shaping the Synthesizer, p. 68.
- [7] Shbobo. *Official website*. [Online; accessed 24-July-2021]. 2021. URL: <https://www.shbobo.net/>.