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

July 2021

Author .....
Program of Media Arts and Sciences
July 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 July 2021, in partial fulfillment of the
requirements for the degree of
Master of Science in Media Arts and Sciences

Abstract

Tiny trainable instruments is a collection of instruments for media arts, using machine learning techniques and deployed in microcontrollers.

Thesis Supervisor: Tod Machover
Title: Muriel R. Cooper Professor of Music and Media

Acknowledgments

This thesis could not have been possible with the support and care of these wonderful people:

Directly related to thesis: thesis commitee + UROPs Peter Tone, Maxwell Wang

Opera of the Future, Future Sketches

Family and friends

Contents

List of Figures

List of Tables

Chapter 1

Introduction

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

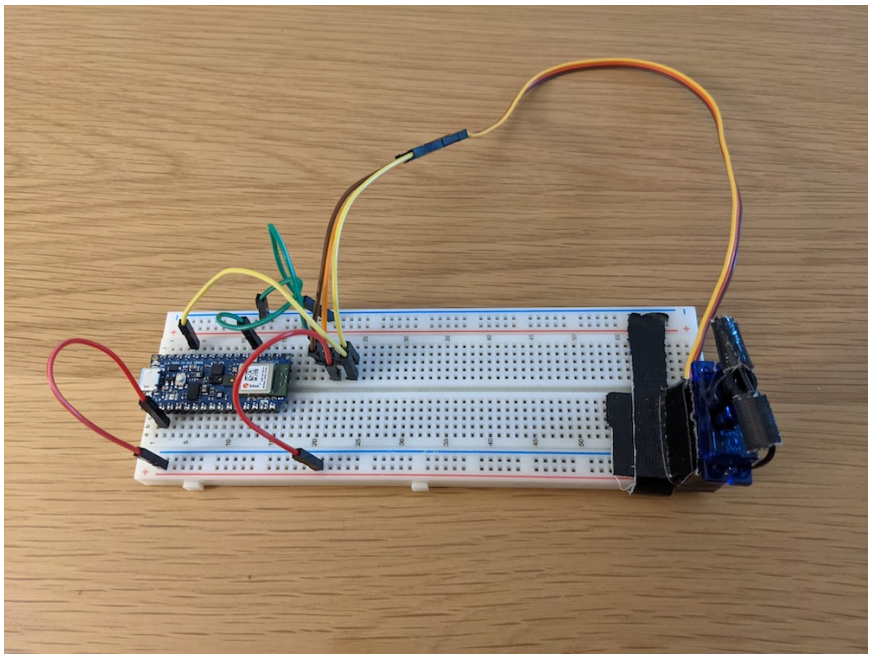

Figure 1-1: Desk at home

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 an 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.

TODO: Explain what is TensorfLow Lite Micro, what does it do, the same for TinyML Foundation

COMMENT: add 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 conditions.

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 in a glossary of terms to go over the basics of machine learning TODO: Add contextual stories to frame each aspect TODO: More pictures TODO: Add how everything is public, even the errors 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

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

2.1 Philosophy

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 Design principles

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

2.3.1 Cheap

2.4 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.

2.5 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.

2.6 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.

2.7 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

2.7.1 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.

2.7.2 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.

2.7.3 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.

2.8 Outputs

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

2.8.1 Buzzer

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

2.8.2 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.

2.8.3 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.

2.8.4 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.

2.9 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 ran 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.

2.10 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/

2.10.1 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.

2.11 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.

2.11.1 Squishies, by Hannah Lienhard

Squishies is Hannah Lienhard's master's thesis, and consists of a novel squishable interfaces for musical expression. We shared discussions about low-level sound design, code reusability, sound art education, 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.

2.11.2 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 want to explain my experience, pitfalls, breakthroughs and what led me to this particular thesis, being open but not bragging, being detailed and praising without boring, being informal while still being academic, being inviting and celebrating, while still being critical.

TODO: draw out a timeline of all of these activities year by year specifically to list out all of the activities that have had an impact on you and the design of this project. Then, go back and pick 5-8 that are most impactful on how you got to where you are now and explain those chronologically to tell a story. This is the story of "why you care about what you're doing your thesis research on and why the reader should care too"

3.1 Microcontrollers

My first exposure to microcontrollers was as an undergraduate student of electrical engineering back home in Chile, around 2006. I learned PIC microcontrollers, and programming using C# on Windows machines, for implementing different projects.

Around the same time, With other classmates we discovered the rising trend of Arduino microcontrollers, which were open source and with a strong hobbyist community sharing examples on the internet. The community behind it was actively sharing code, and it was exhilarating to be able to share code with strangers on the internet. One of the first projects we created was an automatic guitar tuner, where an Arduino board detected the pitch of the string and then moved a motor that moves the tuning gear to match the desired pitch using a PID controller.

I was really impressed with the workflow of the Arduino, its low cost, its compatibility with all operating systems, its documentation, and even more with the passion for sharing of the whole community behind it.

In electrical engineering we were supposed to learn many low-level programming techniques, and Arduino was deemed as too much of a shortcut or high-level programming, so for our capstone undergraduate thesis project we were not allowed to use them, so for thesis I had to implement my robotic project using a cumbersome setup, where I used an Apple machine, with a disk partition to also have Windows installed on it, to write and compile C#, and writing code that was very specific to that particular chip, without a clear way to make it useful to other people, and missing the Arduino workflow.

TODO: comment. there isn't too much of a transition here between your experience as an undergrad and then the community you were searching for around media arts afterwards. I would expand on this. This is a great place to build a story that is connected to the technical material but sets up why you care about your work

Starting in 2010, and particularly after graduating from electrical engineering in 2013 I focused on software, learning 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.

On 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 (TODO: what does this mean?), triggering 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.

TODO: explain what i am doing with the Arduino.

3.2 Machine learning

My first experiment with creative machine learning, was with my NYU ITP classmate Corbin Ordel, 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 gamechanger 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.

Chapter 4

Background

As part of the research that directly informed this thesis, I want to highlight the courses, projects and studies I have undertaken while at MIT Media Lab.

COMMENT ABOUT CLASSES: it may be better to do a lit review of sorts and discuss the material that informs your work here rather than specific classes

4.0.1 Classes

1. Comparative Media Studies, by Sasha Costanza-Chock
2. Recreating The Past, by Zach Lieberman
3. Sound: Past and Future, by Tod Machover

4.0.2 Projects

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.

4.1 Media arts instruments

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.

TODO: maybe add my own personal experience with these instruments.

4.1.1 Bast

The Czech company Bastl Instruments have released the Kastle series of instruments based on the Arduino Uno's chip ATMEGA 328P, and their source code is released on GitHub, with alternate firmware for different behaviors.

Currently they offer the Kastle v1.5, a melodic and drone synthesizer, and the Kastle Drum, for rhythmic work. The only difference between these synthesizers is the firmware and the faceplate, and it is encouraged to write new firmware to modify their behavior. Also they are forgiving instruments, its inputs and outputs are robust enough to allow for mistakes in connections, in an electrical and mechanical way.

This instrument inspired many characteristics present in Tiny Trainable Instruments, including the use of breadboard jumper cables for making connections, their low cost (90 USD), and their portability by use of batteries.

Add an example of how Moog synthesizers in the 1970s were as expensive as a house! And then as a car.

Mention how accessible the different components of the instruments are, how you can get everything from adafruit for not that much money, etc.

Add explanation about how this is not just on GitHub, but also published on the arduino IDE library, and how examples are worked already, and how things are pre-compiled.

Lower barriers include releasing a Bill of Materials, on Adafruit, inspired by the Drawdio project, which can be bought as a kit from Adafruit. TODO: explain why Adafruit is cool and easier to use than Digikey etc.

4.1.2 Critter & Guitari

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

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

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

4.1.3 monome

Aleph: earlier sound computer.

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.

4.1.4 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.

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, they became open source, which is available at 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 appropiating for Tiny Trainable Instruments, such as a drum machine you can talk to, Alexa spinoff.

4.2 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" TODO: add how this project addresses the 4Ps

4.3 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, 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.

TODO: add what does the speed enable you to do on a personal level? why is it important to share that speed with others?

The last spark that led me to this thesis was the release of two 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.

4.4 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 Design Justice Network and the book Design Justice by Sasha Costanza-Chock.

TODO QUESTION: is any of the methodology you use to develop all of this work specifically related to either of these? you might connect specific aspects of the EFF project and Sasha's work that are connected

4.5 Instruments

The table 4.1 is an example of media arts instruments that are scriptable.

Maker	Instrument	Year	Computing	Software
BASTL	microGranny 2	YEAR	microcontroller	Arduino
BASTL	Kastle v1.5	YEAR	microcontroller	Arduino
Critter & Guitari	Organelle	YEAR	computer	Linux?
Critter & Guitari	EYESY	YEAR	computer	Linux?
monome	aleph	YEAR	computer	Linux?
monome	norns	YEAR	computer	Linux
Shbobo	Shnth	YEAR	microcontroller	ARM Cortex
Shbobo	Shtar	YEAR	microcontroller	ARM Cortex

Table 4.1: Table of media arts scriptable instruments

4.5.1 BASTL

BASTL Kastle, two iterations and a spinoff: Kastle, Kastle v1.5, Kastle Drum.

Based on Arduino, GitHub repository with alternate firmware.

Breadboard patching with jumper cables, inputs and outputs robust enough to allow for mistakes in connections.

4.5.2 Critter & Guitari

Organelle computer for sound, scriptable, Linux operating system + Pure Data software.

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

4.5.3 monome

Aleph: sound computer

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

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

4.5.4 Shbobo

Peter Blasser's Shbobo

Shnth and Shtar

Shlisp language and Fish IDE.

github.com/pblasser/shbobo

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

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

The repositories are hosted on GitHub, to promote collaboration, and people can file issues and pull requests.

GitHub repository

Arduino library

PDF zine for explaining, reference as the PDF booklet for monome norns

5.3 Audience engagement

5.4 Workshop

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 a weekend.

On 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

5.5 Multimedia documentation

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

Chapter 6

Conclusions

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

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. Pubslihing 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.

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 inspires more complex

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.

Appendix A

Context

During this thesis I had this particular context:

A.1 Language

My native language is Spanish, and this thesis was written in English.

A.2 Social

This thesis was written during the COVID19 pandemic, during the longest stretch I have had of not visiting my home country Chile, or leaving USA and its east coast, and not working on a shared office, or living with roommates, or seeing family, limiting my physical interactions to mostly a handful of friends and toi the internet.

A.3 Location

Written in Boston MA, New York City NY, Mt Airy MD, between November 2020 and July 2021.

A.4 Software

This thesis document was written using LaTeX and the Microsoft Visual Studio Code editor.

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 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.5 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.

Appendix B

Scripts

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_DESIRED="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 "successs, 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_DESIRED

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

      echo converted "$i" to $EXT_DESIRED

      # 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"
```

```
# 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"

  # check if in directory some files match the extension
  if [ -f $DIR_MEDIA/*.$EXT_ORIGINAL ];

    # if there are matching files then
  then

      echo "successs, there are matching files"

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

```
  else
    echo "no $EXT_ORIGINAL files in $DIR_MEDIA/"

  fi

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 commitee.

```
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 "$f" .tex)
  # delete original docx dile
  rm -f "$PWD/docx/$base.docx"
  # create new docx file with pandoc
  pandoc -s -o "$PWD/docx/$base.docx" "$f"
done
```

Bibliography