Memorandum

TO: Kimberly Lemieux (lemieuxk@camosun.bc.ca)
CC: Justin Curran (jcurran@camosun.ca);
Mel Dundas (dundas@camosun.bc.ca);
Wayne Mayes (Mayes@camosun.bc.ca)
FROM: Jose Figueroa (jose.figfatez@gmail.com);
Temo Guerrero (ccguerrero@gmail.com);
Italo Hernandez (italo.hernandez@ieee.org);
Joel Legassie (joel@jamesbay.org);

DATE: October 27, 2023
SUBJECT: The Conductor: A Progress Report

Revision: 1.0.0
Repository: The Conductor: A Progress Report

Summary

We are making The Conductor, a glove that detects hand gestures and translates them into electronic music. We have faced many challenges over the past few weeks, but through hard work, cooperation, and creative thinking we have kept The Conductor on time and on budget. We are on track to have a working prototype by the end of October, which is exactly on target. This leaves a full six weeks for testing and rework before completion in mid-December.

Introduction

Over the past eight weeks we have worked individually and together in each of our four sub-projects so that we are now ready to bring them together into a working prototype, (see Appendix A).

In the *Hardware* sub-project Temoc and Italo have created solid printed circuit board (PCB) designs and sent them off to the manufacturer. We expect to have them in our hands by October 31. This is a milestone we are proud of because it allows us to move on to the next stage in our development plan, (see Appendix B).

In the *Preparing Data* sub-project, Joel has been optimizing and testing the microcontroller firmware and neural network for fast and reliable data harvesting and processing. We have managed to reduce the total sample processing time by two thirds, giving us response times faster than human perception.

In *Applying Data* Jose has been hard at work building the software that will translate the neural network's predictions of hand position into music. Here we have made a major change to simplify and speed up software development. Instead of building a Virtual Studio Technology (VST) plugin to a Digital Audio Workstation (DAW), we now create our own MIDI messages and send them directly to the DAW.

Simultaneously, Joel and Temoc have started to build and test a graphical user interface that will make The Conductor accessible to non-technical users, (see Figure 2). All of us are collaborating on 3-D printing enclosures to be ready for testing once the circuit boards arrive.

Throughout this activity we have followed plans for communication and *Documentation* outlined in our proposal and group contract. We have maintained our schedule of biweekly meetings and use the WhatsApp group extensively to supplement constant in-person collaboration. This open communication allows us to respond quickly to problems, and is the main reason we are still on time.

Up to now we have developed each of the four sub-projects in its own little silo. Our challenge from this point is to bring them together into a working prototype. We will be in a position to do that once our circuit board arrives and we enter the third phase of our project: testing and rework.

Hardware

In our hardware sub-project, we have completed the design of our printed circuit boards (PCB) and submitted them to JLCPCB for manufacturing and assembly, (see Figure 1). This is a major achievement for our team that unlocks the third and final phase of our project.

Risk and Reward

The Conductor's hardware consists of one main circuit board holding the microcontroller and a MC3416 accelerometer. It will be mounted on the back of the user's hand. The VL53L1X Time of Flight distance sensor, which comes on its own circuit board, will be mounted on the user's palm. There are also three smaller circuit boards holding one MC3416 each that will be mounted on the user's fingers.

Because the MC3416 accelerometers are too small to solder by hand, we are having them assembled by the manufacturer, along with some other very small supporting components[1]. This decision increased the cost and delivery time of our circuit boards, and increased the risk that the circuit boards will not work if we make a mistake in our circuit board design files. However, we determined that the simplicity, and small size of these components make it a worthwhile trade off. To offset the risk we prioritized the completion of circuit board designs in the early weeks of the project. We also have taken great care in the design process, drawing heavily on the expertise of our instructors, through three major revisions of each board.

Our efforts have paid off. Our hardware spending is still well under our original target of $250, (see Appendix C), and we have created circuit boards small enough to mount on the back of the hand and fingers, (see Appendix D). These circuit boards are the foundation of our prototype on which we will build with contributions from in the other three sub-projects.

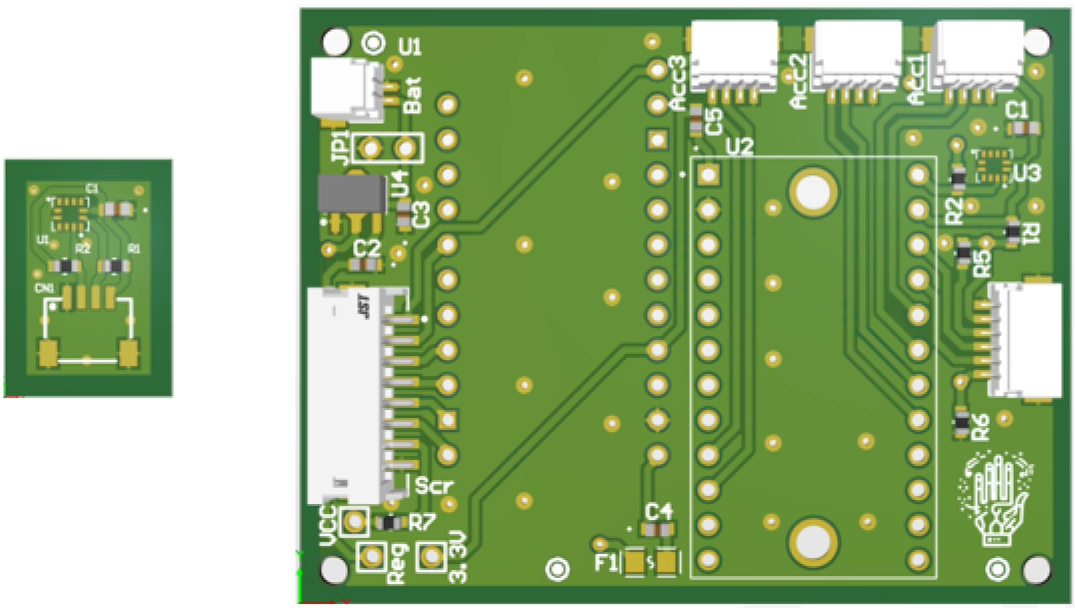


Figure 1: The Conductor's Circuit Board Designs, Version 1.

Preparing Data

We wrote the microcontroller firmware and neural network software to gather and prepare sensor data for The Conductor early on in the project[1, 2, 3]. By testing and troubleshooting it on a breadboard mounted prototype we have developed it to a high level of reliability. In particular, we have spent many hours over the past few weeks optimizing for speed.

A Need for Speed

Sending data over WiFi takes up roughly 90 % of the time required to gather and process data samples. We use a simple TCP socket to format and send data over WiFi from the microcontroller to the user's PC [4]. But, before the socket can send any information it must establish a handshake connection between the two devices.

These handshakes can require over 100 bytes of information to pass back and forth. For comparison a single sample from all five of The Conductor's sensors takes up only 14 bytes. In early versions of the firmware the PC client opened and closed the socket connection every time it requested a sample from the microcontroller. The devices spent most of their time shaking hands and very little talking to each other.. We planned and implemented a code revision that creates a socket when The Conductor is started and keeps it open. Before the change each sample took, on average, 100 milliseconds to transmit and process. Now, The Conductor the average is 30 milliseconds per sample. These numbers are consistent with a real time response rate. We should be able to gather and prepare data between the notes in a musical performance

Once the circuit boards arrive there will be one more major revision to the data preparation software to adapt to sensor data from real hand movements. There will always be bugs to troubleshoot, but we do not expect any major challenges in the final stage of this sub-project, leaving more time to explore exciting ways to apply the data we are collecting.

Applying Data

Once we had fast and accurate data output from the neural network we were ready to apply that data to making music. We have had to make some difficult choices to make it possible, but The Conductor can now manipulate controls within a DAW. We are making music! We are well poised to test and expand on this ability in the remaining six weeks of the project.

Sonification of Data

The Conductor does not have a keyboard, touchpads, or knobs. These traditional controls allow musicians to set the state of the control and get predictable musical output. In contrast, The Conductor collects information about the user's relationship to their surroundings and 'sonifies' it by generating MIDI commands for standard controls within a DAW [7]. It gives musicians an exciting new method of creating musical pieces and performances.

To make this work we have developed and tested a generic MIDI command generator application. It looks for patterns in the stream of predictions that the neural network makes about the user's hand position. The user can program the application to act on those patterns to create arbitrary MIDI messages that also incorporate information provided by the time of flight sensor. Once the circuit boards arrive, we will train the neural network on data collected from a real-user's hand. After that we will be able to test and finalize a set of example controls:

- sine, square, and saw wave modulation, (see Appendix E),
- note-by-note arpeggiation, (see Appendix F),
- play and modify MIDI notes, (see Appendix G).

Concurrently, we have started to build a prototype graphical user interface (GUI). The GUI will allow users to interact with The Conductor on their PC in a number of ways:

- Connect their PC to The Conductor,
- set neural network training parameters,
- train the neural network,
- map hand movements to musical controls or notes
- monitor the flow of data as they use The Conductor to make music.

Difficult Choices

In creating our MiDi generator we used cooperation and open communication to identify and eliminate a major bottleneck in connecting The Conductor to a DAW. In the initial phase of the project we discussed and researched a number of methods to make that connection. We decided to use the open source JUCE platform written in C++ to create a Virtual studio Technology (VST) plugin [6]. The plugin would let users control The Conductor from within their DAW. However, once we started developing our VST application we found it would be too difficult to integrate the TCP socket client and neural network written in Python with JUCE's C++ code base..

We consulted as a team and reluctantly decided to trade the VST's seamless DAW integration for the simpler MiDi generator, (see Appendix H).

The path forward includes fleshing out our MIDI functionality, and testing it in real-time with The Conductor mounted on a user's hand. Our team's hard work and flexibility in the first two phases of the project have positioned us well to incorporate our work in the applying data sub-project into the working prototype once the circuit boards have arrived.

Documentation

Good communication and consistent documentation have kept us on track. Early on in the project we successfully established a shared cloud drive for tracking working files and formal documentation. We organized our firmware and PC software into unified git repositories and established a system to work collaboratively on a complex and growing code base. In the process of designing printed circuit boards we collected and tracked a large amount of valuable documentation:

- Bill of Materials (BOM)
- Electrical schematics
- PCB print documents (gerber files, drill files)
- PCB assembly file (centroid)
- 2-D and 3-D PCB images (.png, .step)
- A user manual for ordering PCBs from JLCPCB [7]

It is a credit to all group members that we have maintained open and consistent communication throughout the first two stages of the project. Our professionalism and collegiality have allowed us to overcome multiple challenges and stick to our schedule.

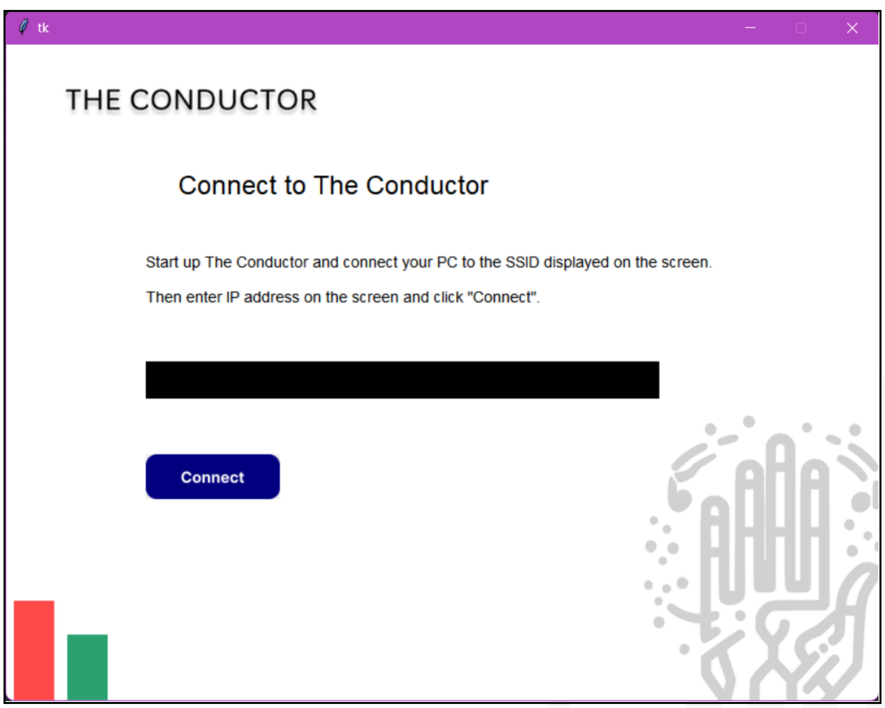


Figure 2: The Conductor's GUI in development..

Conclusion

Since our proposal we have made progress in our sub-project areas and are ready to bring the separate parts of the Conductor together in a working prototype. We are pleased with the progress we have made so far, but there is still much work to do.

Once our circuit boards arrive we will have to quickly determine if they will function properly, in case we have to order a second revision. In addition, we need to flesh out and test our user interface, and we must fit and secure our 3-D printed circuit board enclosures onto a glove and test with the user's hands. Once the prototype is assembled and the user interface is functioning we will do as much testing, troubleshooting and feature development as we have time for, while we begin to develop our website and prepare for the final presentation and symposium. The Conductor team is ready to meet these challenges together and are excited to demonstrate our project to you.

References

[1] H. Kinsley, D. Kukieła. *Neural Networks from Scratch in Python*. Harrison Kinsley; 2020.

[2] B. Krose, P. van der Smagt, *An Introduction to Neural Networks*. English Edition, November. 1996.

[3] R. Shamalik, S. Koli, "Real Time Human Gesture Recognition: Methods, Datasets and Strategies," in *Recent Trends in Intensive Computing*, 2021.

[4] M. Systems, "RFC 9293 Transmission Control Protocol (TCP)," Aug 2022. [Online]. Available: https://www.ietf.org/rfc/rfc9293.html. [Accessed 26 Oct 2023]."RFC 9293 Transmission Control Protocol (TCP)." MTI Systems, Aug. 2022. Accessed: Oct. 26, 2023. [Online]. Available: https://www.ietf.org/rfc/rfc9293.html

[5]"Data Sonification: A New Cosmic Triad of Sound - NASA." Accessed: Oct. 26, 2023. [Online]. Available: https://www.nasa.gov/universe/data-sonification-a-new-cosmic-triad-of-sound/

[6] "Features," JUCE. [Online]. Available: https://juce.com/juce/features/. [Accessed: Sept 11, 2023].

[7] The Conductor, "Instruction manual for submitting a successful PCB design to JLCPCB," 2023. [Online]. Available: https://docs.google.com/document/d/1wHMZg6LsHkX-VwNjcxLu68rwI_WlIJoG/edit.

Appendix A: Sub-project List

Hardware

Co-leads: Temo Guerrero, Italo Hernandez

- Build a wearable device with an embedded microcontroller, printed circuit boards, enclosures, motion sensors, and WiFi transceiver.

Gathering Data

Co-leads: Jose Figueroa, Joel Legassie

- Write software to recognize human gestures based on data gathered by the wearable device.

Applying Data

Co-leads: Jose Figueroa, Italo Hernandez

- Write software to translate gesture data into digital musical signals.
- Build a user interface to allow digital musical signals to interact with digital musical tools.

Documentation

Co-leads: Temo Guerrero, Joel Legassie

- Prepare promotional materials, design documents and other documentation (a website and user manual, or *Circuit Cellar* article).

Appendix B: Schedule

Proof of Concept and Circuit Board Prototype Design (Sept 2023)

- [x] Develop firmware for the ESP32 to collect sensor data and send it to the PC client.
- [x] Develop the neural network on the PC client, train it with 15 gestures and predict gestures in real time.
- [x] Design and fabricate printed circuit boards.
- [x] Plan and design VST software.
- [x] Prepare documentation and marketing materials.

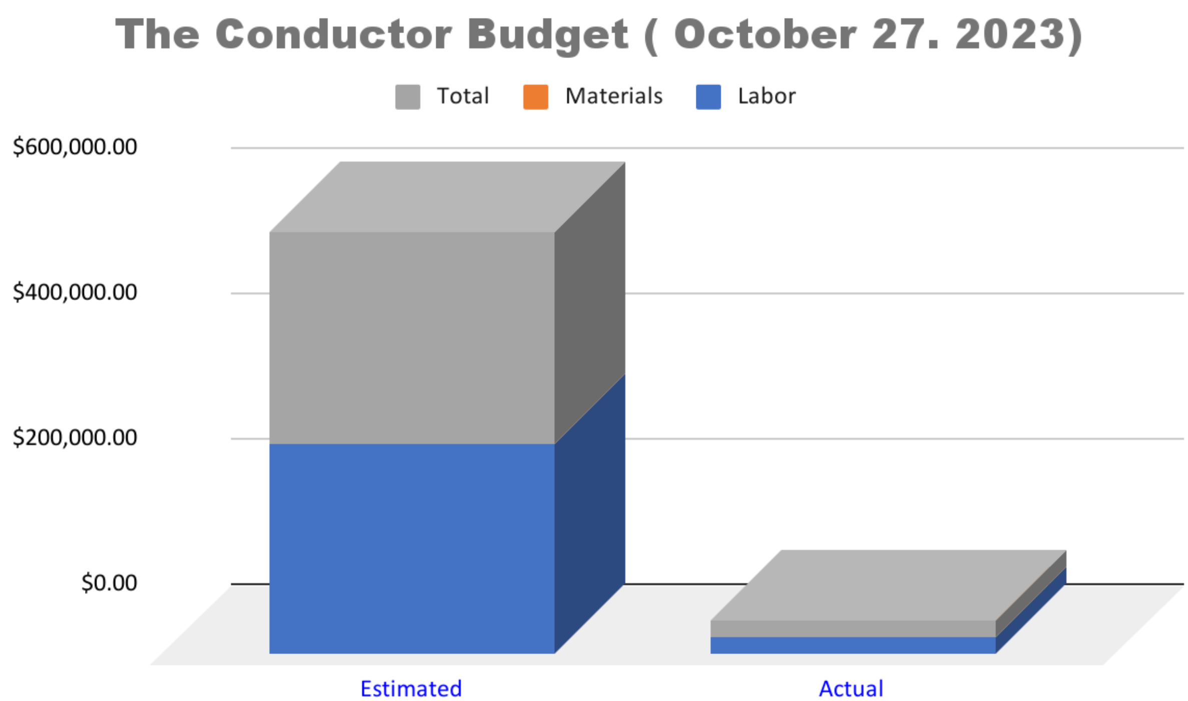
Prototype Construction (Oct 2023)

- [ ] Assemble components on printed circuit boards and test circuit boards.
- [x] Design and print enclosures for circuit boards and mount them on a user's hand.
- [x] Train neural network with data from the hand mounted prototype.
- [x] Prepare documentation and marketing materials.
- [ ] Predict gestures with the hand mounted prototype.
- [ ] Design GUI for windows app. *(In progress)*

Testing and Rework (Nov 2023)

- [ ] Print and test improved enclosures based on testing feedback. *(In progress)*
- [ ] Assemble and test version 2 circuit boards.
- [ ] Rework circuit boards as required.
- [ ] Prepare documentation, website, and marketing materials.
- [ ] Final presentation and symposium demonstrations.

Appendix C: Budget



The Conductor Budget up to October 27			
	Estimated	**Actual**	**Budget deviation**
Labor	$289,920.00	$23,400.00	**-91.93%**
Materials	$394.94	$181.56	**-54.03%**
Total	$290,314.94	$23,581.56	**-91.88%**

Appendix D: Bill of Materials

Table 1. Child Board Accelerometer (3)

Comment	Description	Designator	Footprint	LibRef	Qty	LCSC Part#
0.1uF	SMT Capacitor 0.1 uF	C1	CAP_AMK107BJ475MK-T_TAY	AMK107BJ475MK-T	1	C1591
4 pin connector	4 pin connector	CN1	CONN-SMD_4P-P1.00_SM04B-SRSS-TB-LF-SN	SM04B-SRSS-TB (LF)(SN)	1	C160404
4.7k	SMT Resistor 2.7k	R1,R2	RC0603N_YAG	RC0603JR-072K7L	2	C105428
Acc4	MC3416-P VLGA-12 Attitude Sensor/Gyroscope ROHS	U1	LGA-12_L2.0-W2.0-P0.50-BL	MC3416-P	1	C5250287

Note: The circuit board quote includes five copies of two separate circuit board designs and assembly of the surface mount accelerometers. The cost per unit produced will be considerably lower than this value.

Table 2. Parent Board (1)

Comment	Description	Designator	Footprint	LibRef	Qty	LCSC Part#
SM04B-SRSS	4 pin JST connector	Acc1, Acc2, Acc3	CONN-SMD_4P-P1.00_SM04B-SRSS-TB-LF-SN	SM04B-SRSS-TB(LF)(SN)	3	C160404
2 pin connector	2 pin JST connector	Bat	CONN-SMD_SM02B-SRSS-TB(LF)(SN)	SM02B-SRSS-TB(LF)(SN)	1	C160402
0.1uF	SMT Capacitor 0.1 uF	C1, C4, C5	CAP_AMK107BJ475MK-T_TAY	CL10B104KB8NNNC	3	C1591
1uF	SMT Capacitor 1 uF	C2, C3	CAP_AMK107BJ475MK-T_TAY	CL10B105KQ8NNNC	2	C105524
Resettable Fuse	6V 350mA 100A 750mA 0805 Resettable Fuses	F1	F0805	0805L035YR	1	C2687882
4.7K	SMT Resistor 4.7k	R1, R2	RC0603N_YAG	RC0603JR-074K7L	2	C105428
2.7K	SMT Resistor 2.7k	R5, R6, R7	RC0603N_YAG	RC0603JR-072K7L	3	C127451
8 pin connector	8 pin JST connector	Scr	FP-S8B-ZR-SM4A-TF_LF_SN-MFG	CMP-17439-000670-1	1	C471474
ToF sensor connector	6 pin connector	ToF	CONN-SMD_SM06B-SRSS-TB-LF-SN	SM06B-SRSS-TB(LF)(SN)	1	C160405
Acc4	VLGA-12 Attitude Sensor/Gyroscope	U3	LGA-12_L2.0-W2.0-P0.50-BL	MC3416-P	1	MC3416-P
Regulator	3.3V, 150mA SOT-89-3 LDO Linear Voltage Regulator	U4	SOT89-150P350X160-3N	AP7380-33Y-13	1	C115096

Appendix E: MIDI Control Changes

MIDI Control Changes (abbreviated MIDI CC) are a type of MIDI message that are mapped to parameters inside the DAW such as the amplitude of a delay effect, the cutoff of a filter, or the volume of an instrument channel. They are used to allow users to shape the sound of their instrument rather than what note value is being played [16].

The Conductor uses MIDI CC messages to generate dynamic values to control parameters inside the DAW. In our application we can send a stream of MIDI CC messages which represent several waveforms, essentially creating low frequency oscillators that can modulate the texture of the sounds it controls.

The MIDI CC messages will be controlled by the trained gestures of the glove and the time-of-flight sensor. For instance we can utilize the time-of-flight sensor to change the frequency of a MIDI CC waveform so the effect speeds up or slows down depending on the gesture held and the data received from the time-of-flight sensor. Having multiple gestures controlling MIDI CC messages allows for evolving melodies using the built in effects inside a DAW.

Appendix F: Arpeggiation

Arpeggiators are a popular method of developing musical melodies with electronic instruments. They function by taking in a set of notes from a MIDI input source and sequencing them based on the conditions created in the module[16].

Arpeggiators typically control the order of the sequence of notes (i.e. play the notes for lowest to highest or in a random order) The rate at which the sequence can also be controlled, for instance playing a sequence of whole notes and changing to playing a sequence of eighth notes would speed up the sequence while keeping in time. The conductor will focus on binding a gesture to a parameter inside its arpeggiator so that users can utilize the glove to generate new musical ideas from an external MIDI source.

Appendix G: Syncing Multiple MIDI Controls

Music is essentially organized sound, therefore the conductor requires functionality to start and stop its MIDI effects, and sync them with other instruments. The Conductor uses a global beats per minute clock to organize all the different modulations that are occurring. Users will be able to conduct the flow of MIDI messages through gestures, this is currently being developed in our MIDI application which will control the global rate of the MIDI CC messages and arpeggiation.

Appendix H: Engineering Change Management

Engineering Change Request

ECR Number 01	Issue Summary: Restructuring of MIDI application architecture
Origination Date 2023-10-16	Requested Action Summary: Change architecture to purely python
Originator Jose Figueroa	Urgency: Medium/High
Project :The Conductor, Applying Data	Priority: Medium

Request Details

Issue: Integration of the VST framework with our neural network is requiring a high amount of complexity that is taking away from the main priority of the project.

Requested (or Recommended) Change

Move development of the application to python to simplify the communication between the neural network and the VST software.

Risk if change not completed

The complexity of binding the neural network with c++ framework will cause unnecessary cost compared to writing the code in one language

Risk associated with change

Loss of built in GUI in framework, GUI will need to be developed in python

Engineering Change Response

ECR Number 01	Issue Summary: Restructuring of MIDI application architecture
Origination Date 2023-10-16	Requested Action Summary: Change architecture to purely python
Originator Jose Figueroa	Urgency: Medium/High
Project :The Conductor, Applying Data	Priority: Medium

Request Details

Issue: Integration of the VST framework with our neural network is requiring a high amount of complexity that is taking away from the main priority of the project.

Requested (or Recommended) Change

Move development of the application to python to simplify the communication between the neural network and the VST software.

Risk if change not completed

The complexity of binding the neural network with c++ framework will cause unnecessary cost compared to writing the code in one language

Engineering Change order

ECR Number: 1	Project: MIDI Application
ECO Number: 1	Change Summary: Switched from using JUCE framework (C++) and python to purely python do develop the Conductors PC applications
Owner: Jose Figueroa	Change Date
Approver: Joel Legassie	

Change Description

Initially our plan for developing the PC application for the Conductor used the JUCE framework which is a C++ framework for developing Virtual Software Technologies. The benefits of using JUCE included tight integration with Digital Audio Workstations, higher processing speed when compared to python, and built in GUI objects.

The Issue that caused us to change to using python for the application was the complexity of integrating python and the JUCE C++ framework.

Extensive testing was done to learn how to integrate our neural network which was written with python into the C++ framework. We were able to have a simple python script run inside a C++ program but since we required our application to modify parameters from both the python and C++ side of our application, the time required to develop wrappers for all facets of the application would require more resources that outweighed the benefits of using both languages.

Moving forward the Conductors application will be developed entirely in python which will reduce the complexity of communication data between processes in our application.

Appendix I: Glossary

a.i

artificial intelligence algorithms that are able to mathematically 'learn' to associate properly formatted and accurate inputs with correct outputs with a greater (or lesser) degree of accuracy. A neural network is one form of a.i. It is not to be confused with true Artificial Intelligence which does not exist, yet. Back

C++ Framework

A C++ framework is a set of supporting software that provides extended functionality to the C++ applications. *Return*

DAW

A Digital Audio Workstation (abbreviated as DAW) is a software application that allows users to produce, edit, record, and perform music. *Return*

PCB

A printed circuit board (abbreviated as PCB) is the board with non-conductive material where it can be installed wires, electronic components, traces, and others. *Return*

I2C

Inter-Integrated Circuit (I2C) is a serial communication protocol developed in 1982 by Philips Semiconductors used for connecting microcontrollers with peripherals such as sensors.

TCP Client

TCP (Transmission Control Protocol) is an internet protocol used to transmit information across the internet. *Return*

VST

Virtual Studio Technology is audio plug-in software used for connecting digital audio software with digital audio workstations. *Return*

TCP socket

A TCP socket is a technology that identifies the unique IP address and port of an application for communication between other applications [18]. Return

WhatsApp

An instant messaging service owned by Meta Inc. *Return*

ALTIUM Designer

Altium Designer is a printed circuit board design software package. *Return*

Accelerometer

Accelerometers sense the instantaneous change in velocity using the change in temperature experienced by the sensor. *Return*

Time-of-Flight Sensor

A sensor that detects distance. *Return*

TinyS3

A development board in the ESP32 product line developed by Unexpected Maker. *Return*

MIDI CC

MIDI CC are used for continuous control of musical parameters that do not contain note information. *Return*

MIDI Standard

MIDI (Musical Instrument Digital Interface) is a standard for controlling and communicating with electronic music instruments. *Return*

Python

Python is a high-level general-purpose programming language. *Return*

Sonification

The process of translating data into sounds. Return