



Haptic Piano Instructional Gloves

Neil Guan
Megan Mileski

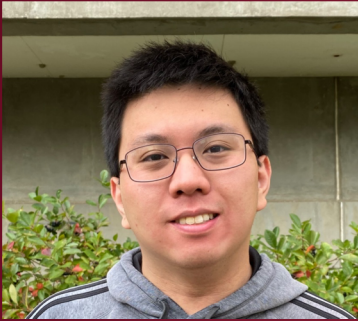
Prepsa Ghimire
Paulina Vu

University of
Massachusetts
Amherst

BE REVOLUTIONARY



Team 11



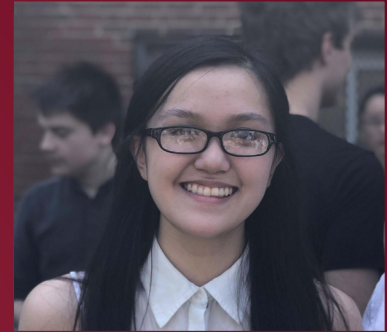
Neil Guan
Electrical Engineer



Megan Mileski
Electrical Engineer



Prepsa Ghimire
Electrical Engineer



Paulina Vu
Computer Engineer



Professor Anderson
Advisor

University of
Massachusetts
Amherst

Problem Statement

Piano is notoriously difficult to learn for several reasons:

- Sheet music does not show the correct fingerings corresponding to which keys to press.
- Sheet music also requires a large amount of memorization and music theory background.
- A great deal of time, effort, and patience is required

Especially for beginners, piano can be a daunting instrument to learn. The mental processing between knowing which keys to press and in what order requires intuition. The memorization, music theory knowledge, and time commitment to learn piano can be discouraging to those wanting to learn the instrument.

Our Solution: Haptic Piano Instructional Glove

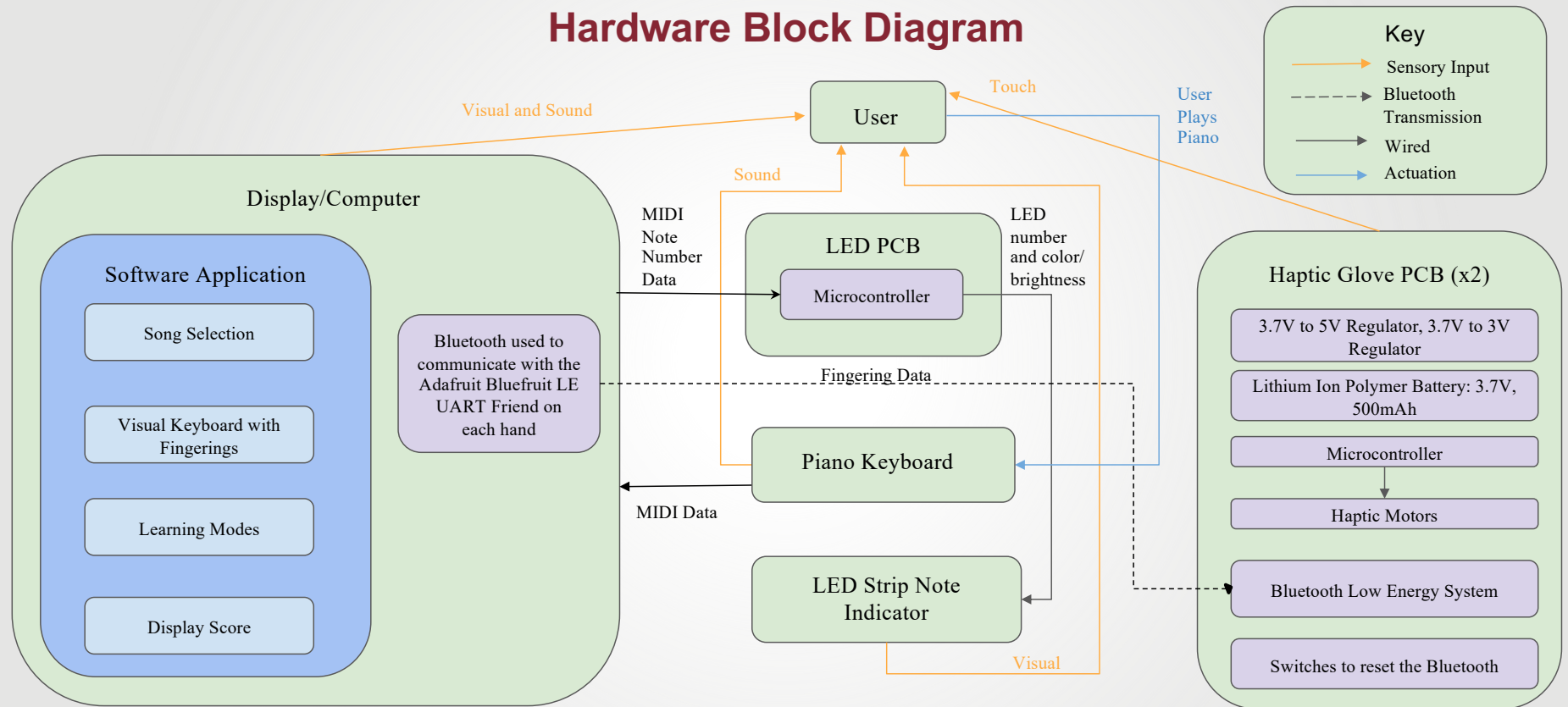


Solution Explanation

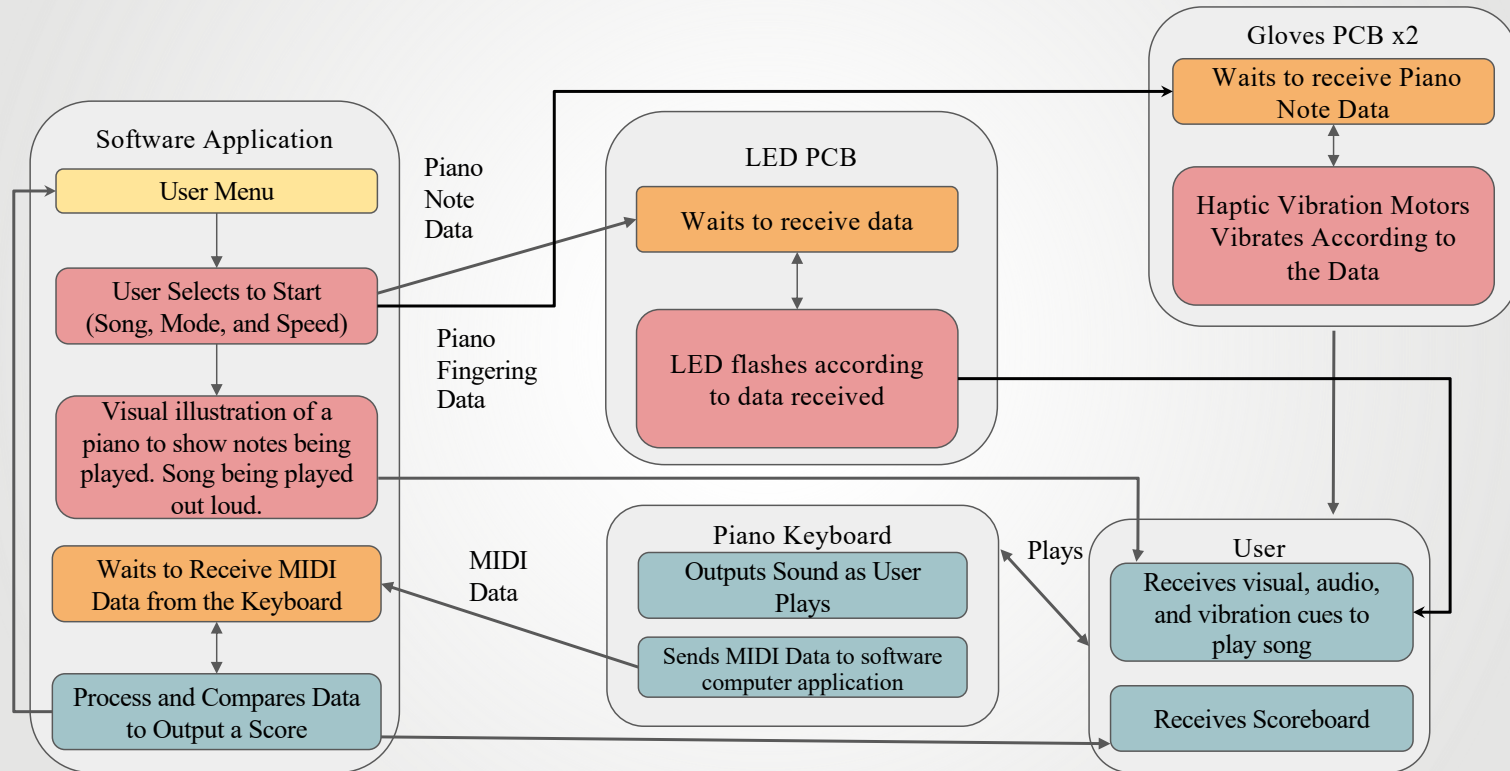
Haptic Piano Instructional Glove addresses issues that come with learning how to the piano by:

- Teaching the pianist to play the piano without sheet music
- Providing the pianist feedback on his/her playing
- Helping the user learn while not actively playing the piano
- Engaging multiple of the pianists' senses in real time whilst playing a song--namely touch, vision and sound for active learning

Hardware Block Diagram








Software Block Diagram



MDR Deliverables Review

For MDR our system will:

1. Wirelessly send 1 chosen song from a laptop to the bluetooth module on each glove 
2. The bluetooth module will feed the signal to the Arduino for each glove 
3. The Arduino will process the signal and power the output pin to the corresponding vibration motor for a time duration equivalent to the length of the note on each glove 
4. There will be a visual display depicting which notes should be played 
5. The user input will be recorded for an accuracy report 

MDR Deliverables Review

For MDR our system will:

1. Wirelessly send 1 chosen song from a laptop to the bluetooth module on each glove ✓
2. The bluetooth module will feed the signal to the Arduino for each glove ✓
3. The Arduino will process the signal and power the output pin to the corresponding vibration motor for a time duration equivalent to the length of the note on each glove ✓
4. There will be a visual display depicting which notes should be played ✓
5. The user input will be recorded for an accuracy report ✓



Automatic Fingering Generation

The automatic fingering generation takes in any MIDI or MusicXML file and generates piano fingerings for each note based on a HMM implementation published in 2020, which is the current state of the art.

The library used is autofingering, sourced from a github repository

The work we did was to copy some of the source code from the main file to obtain data in a workable format and parse the data to be sent to the bluetooth transmitter.

<https://github.com/thegreatkwanghyeon/autofingering>

Nakamura, E., Saito, Y., & Yoshii, K. (2020). Statistical learning and estimation of piano fingering. *Information Sciences*, 517, 68-85.

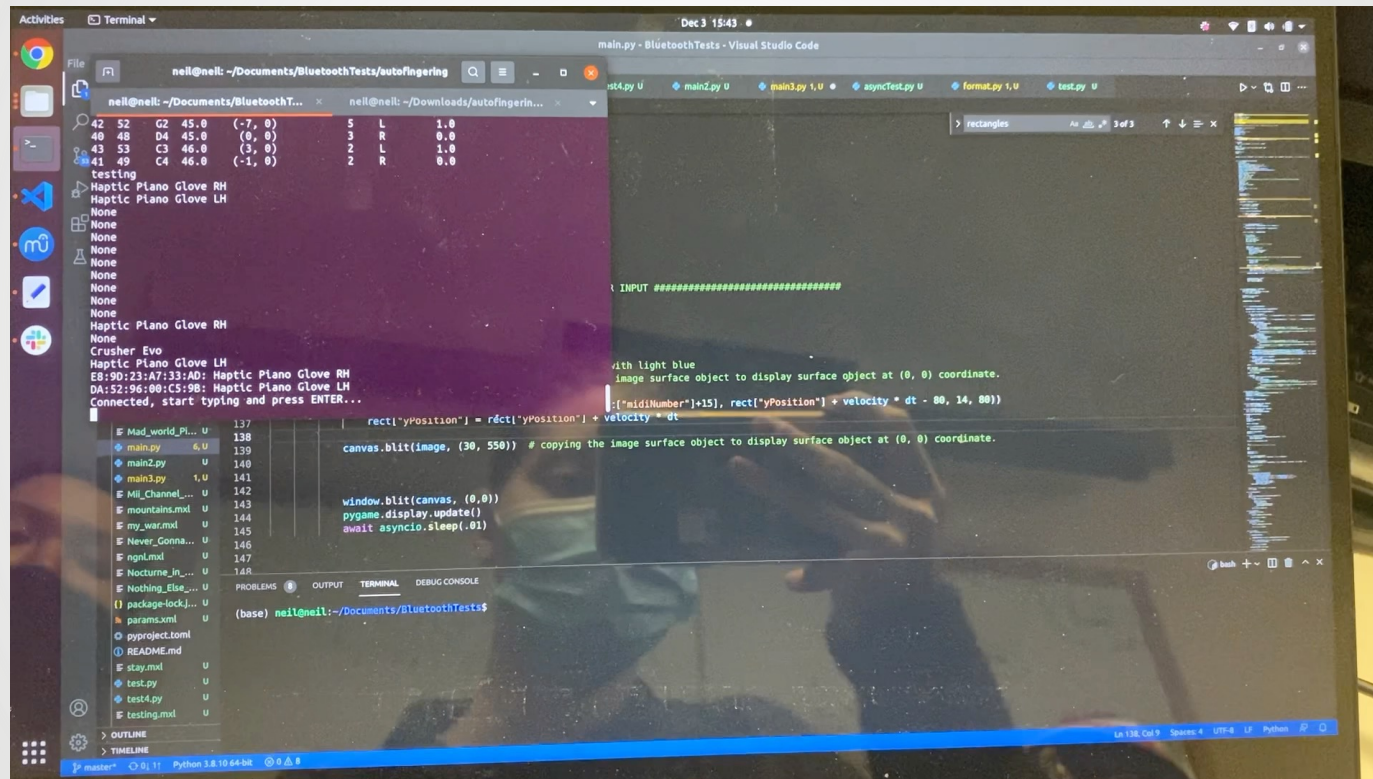
Bluetooth

- The bluetooth uses UART protocol to communicate between a Adafruit Bluetooth LE Friend on each glove with the built in bluetooth receiver on a computer.
- We used a computer for bluetooth because we do not have to invest in additional bluetooth hardware
- We investigated using a Raspberry Pi, but we decided it best to use a computer for now due to processing power concerns.
- The bluetooth device feature for now detects a device that uses the UART Nordic Semiconductor UART protocol UUID and checks the device name; pairing will be advanced in the future. There are currently issues with initial pairing reliability.
- Bluetooth packet receiving and reliability performs relatively well with the latest firmware. We send each chord in real time. It can keep up with the hardest of songs.
- Packets are each one byte, with 5 bits corresponding to which fingers to vibrate and 3 bits corresponding to intensity. The timing information is based on when the bluetooth packets are sent.

LED Subsystem

- Converts the pitch (A0 - C8) to a Note Number (36 - 96) using functions in python
- Note Number converted to HEX in python
- HEX value with the color is sent to Arduino via pyserial in bytes
 - To turn ON the 17th LED on the strip send, “h10202020” to the Arduino’s Serial Monitor
 - To turn OFF the 17th LED on the strip send, “h10000000” to the Arduino’s Serial Monitor
- The colors on the LED represents LEFT or RIGHT hand
 - Right Hand: PINK
 - Left Hand: WHITE

GUI Demo:



```
neil@neil: ~/Documents/BluetoothTests/autofingerin...
42 52 C2 45.0 (-7, 0) 5 L 1.0
40 48 D4 45.0 (0, 0) 3 R 0.0
43 53 C3 46.0 (3, 0) 2 L 1.0
41 49 C4 46.0 (-1, 0) 2 R 0.0

testing
Haptic Plano Glove RH
Haptic Plano Glove LH
None
None
None
None
None
None
None
Haptic Plano Glove RH
None
Crusher Evo
Haptic Plano Glove LH
E8:9D:23:A7:33:AD: Haptic Plano Glove RH
DA:52:96:00:C5:9B: Haptic Plano Glove LH
Connected, start typing and press ENTER...

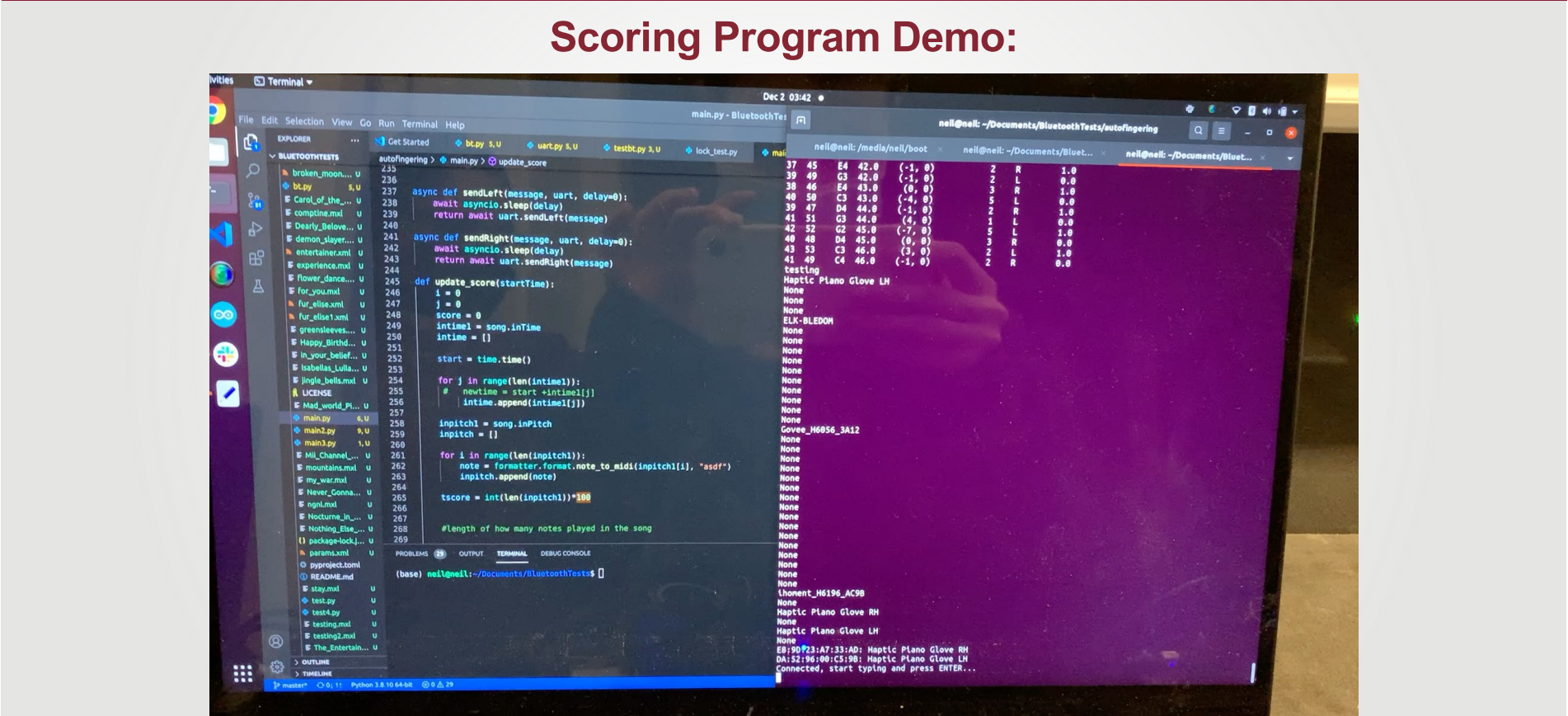
rect["yposition"] = rect["yposition"] + velocity * dt
canvas.blit(image, (30, 550)) # copying the image surface object to display surface object at (0, 0) coordinate.
["midilNumber"]*15], rect["yPosition"] + velocity * dt - 80, 14, 80))

with light blue
image surface object to display surface object at (0, 0) coordinate.
window.blit(canvas, (0,0))
pygame.display.update()
await asyncio.sleep(.01)

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
(base) neil@neil:~/Documents/BluetoothTests$
```

Graphic User Interface

- Displays an 88 key keyboard from Yamaha website
- Rectangles fall down the screen and reach the keyboard image to visually indicate which key(s) should be played
- Created using Pygame and uses the asyncio library to stay synchronized with the LEDs and haptic motors. Ideally runs at 60 FPS
- Dictionary that has all of the rectangle X coordinates that have corresponding MIDI valued keys and every previous rectangle's Y coordinate is stored in a list
- For every game loop the Y coordinate increments for every index in the list. Within each game loop a rectangle is created for list index values that have Y coordinates less than 531.2 which is the top of the keyboard image.



Scoring Program

- Input dictionary generated from MIDI file data, output dictionary from the piano
 - Pitches, Timing
- Output dictionary is continuously being updated
- Uses for loops to determine if user played within range
 - allows for user to be 1 second late or 1 second early (pending change)
 - checks output dictionary if the input not has been played
 - if played, remove from input dictionary
 - will receive 0 if not in time or not correct note
- Outputs a score in terminal
 - Outputs “ Score: 2600/ 8600 ”
 - User correctly pressed 26 notes out of a total of 86 notes
 - System refreshes every second to update score

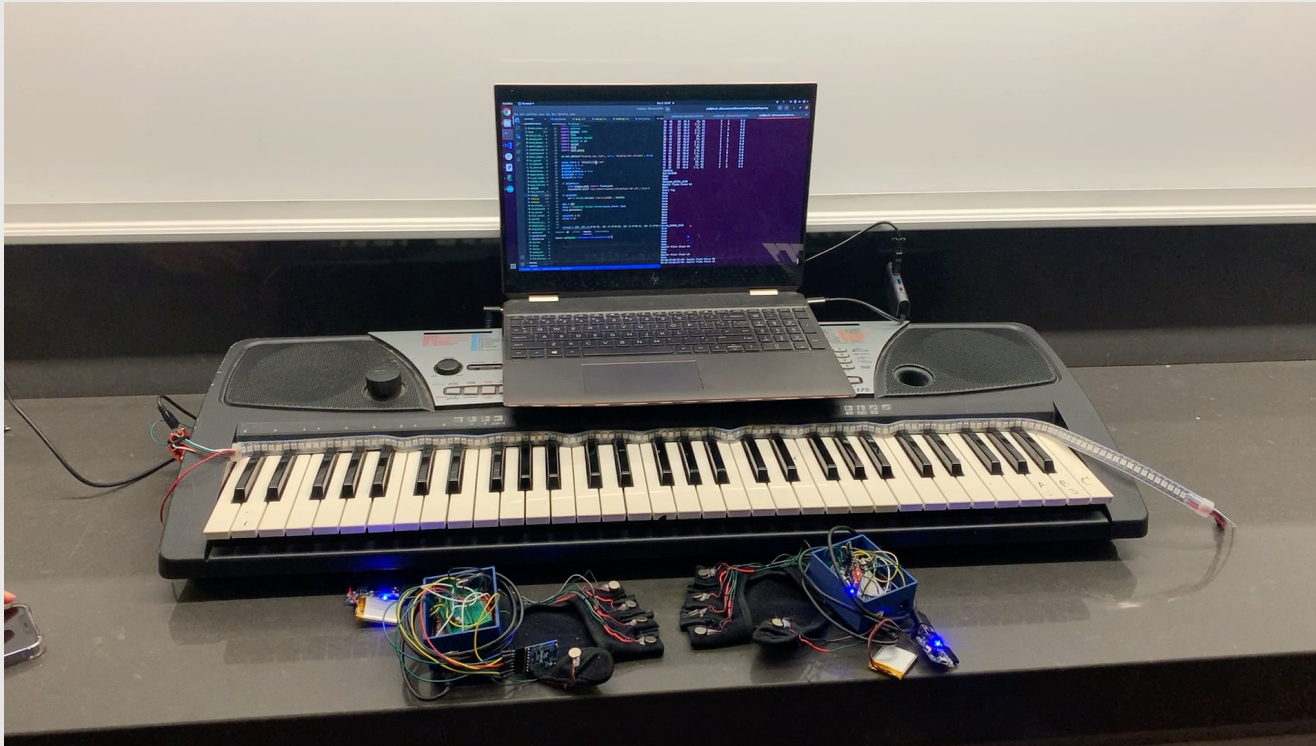
Haptic Motor Gloves

Glove Components:

- A pair of black wrestling gloves with wrist straps
- 3D printed electronic boxes each containing:
 - Arduino Uno
 - Lithium Ion Polymer Battery: 3.7V, 500mAh
 - 3.7V to 5V Regulator, 3.7V to 3V Regulator
 - Adafruit Bluefruit LE UART Friend (Bluetooth Low Energy)
- 10 haptic motors sewed to the gloves



Full System Demo:

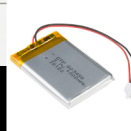
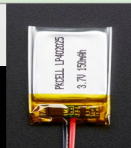


MDR Hardware Components/Modules Used

Component	Justification
Haptic Vibration Actuators	After initially purchasing a couple of the haptic motors to test out, the vibration was determined to be strong enough to be felt through the gloves. The Adafruit actuators were the cheapest reliable ones
Visual Display: 1. WS2812B LED strip (144 Leds) 2. Computer	<ol style="list-style-type: none"> 1. One meter LED strip with 144 LEDs was used because there needed to be enough LEDs for each of the keys on the piano. Here each key has two LEDs (only one turns on). 2. Computer needed in order to analyze the MIDI data gathered from the piano and for GUI
Adafruit Bluefruit LE UART Friend - Bluetooth Low Energy (BLE)	<ul style="list-style-type: none"> • Files freely available online to help us design glove PCB • Uses UART, which is simpler than SPI • Low Latency, firmware upgrades allow for many data packets to be sent in quick succession
Lithium-Ion Polymer Battery	Used to make model portable for non-active learning and used to make system rechargeable.
PowerBoost 1000 Charger - Rechargeable 5V Lipo USB Boost @ 1A	Used to charge the battery and convert 3.7V to 5V for the Arduino and the bluetooth module
Arduinos	Only needed simple processing power to turn on haptic motors and LEDs on the strip.

Battery Choice

	Lithium Polymer	Lithium Ion	Coin Cell Battery	AAAA Battery
Form Factor (size)	Medium 30mm x 35mm 500mAH	Medium 25mm x 35mm 400mAh	Great 20mm diameter 250mAh	Bad 40mm x 10mm x 10mm ~450 mAh
Discharge Rate	Great 500mA continuous	Good 80mA continuous 400mA burst	Unusable 0.2mA continuous	Good Up to 300 mAh, but high current draw reduces capacity
Safety	Medium Dangerous (less risk of leaking electrolytic component)	Most Dangerous	Least Dangerous	Least Dangerous
Rechargeable?	Y	Y	N	N
Price (including charger)	\$7.95	\$25	\$2	\$1
Charge Cycles	Medium 500 (until 60% capacity)	Good 300 (until 80% capacity)	1	1



MDR Hardware Components Costs

Component	Quantity	Cost
MIDI Piano Keyboard	1	\$0.00
Gloves	1	\$0.00
Haptic Vibration Actuators	10 (1 for each finger)	\$21.50
Visual Display: 1. WS2812B LED strip (144 Leds) 2. Computer or Use of Tablet	<ul style="list-style-type: none"> LEDs for each key (61): 1 Computer 	\$0.00
Adafruit Bluefruit LE UART Friend - Bluetooth Low Energy (BLE)	2 (One for each glove)	\$35.00
Lithium-Ion Polymer Battery	2 (One for each glove)	\$15.90
PowerBoost 1000 Charger - Rechargeable 5V Lipo USB Boost @ 1A	2 (One for each glove)	\$39.90
Total		\$112.30

*Have Spent = \$171.07 Total

*Currency Left = \$328.93

MDR Software Components Used/Justifications

Component	Use	Justifications
MusicXML Files (or MIDI) (MuseScore)	Songs ranging from C2 – C6 for the 61 Keys	MusicXML (or MIDI) files are used to gather fingering and piano key data.
Python 3 - Libraries Used: asyncio, mido, auto fingering, pandas, music21, bleak, numpy, serial	Used to coordinate async options, interface with MIDI input, generate fingerings, manipulate data, parse MusicXML/MIDI files, connect to bluetooth, process numbers, and connect to LED strip via serial	Rather than hardwiring the fingerings and the piano notes for songs, libraries are used reads xml (or MIDI) files for any song to give the notes and fingerings needed to play
Bluetooth (Arduino and Python) Uses bleak dependency	<ul style="list-style-type: none"> • Computer: Sends fingering data for the haptic motors • Adafruit Bluefruit LE UART Friend: Receives fingerings sent from the laptop to turn on the Arduino pin corresponding to a finger on each of the hands 	<ul style="list-style-type: none"> • Computer: Due to the processing power required for bluetooth, serial connection scoring, fingering generation, and especially pygame all in parallel, we chose a computer. Plus, computers can connect to up to an unlimited number of devices with no extra hardware. • Adafruit Bluefruit LE UART Friend: Bluetooth Low Energy (BLE) devices durable due to its low energy usage
Arduino (LED) - Libraries Used: Adafruit NeoPixel	Piano note data sent to the Arduino via pyserial from the python code. Python code sends bytes corresponding to the LED number ranging from 0-143 in HEX as well as the color (EX: h01202020)	Able to turn on multiple LEDs at the same time and was used because it was possible to send note data from python to Arduino easily and quickly.
Arduino (Haptic Motors)	Reads bluetooth data and vibrates haptic motors	Fast enough to process data to the haptic motors

System Specifications + Verification

The gloves are fashioned to be secure and doesn't hinder the player's ability to bend his/her joints.	All of the members within the team is able to freely bend their fingers to play the piano, although there is some resistance for larger hands. The system is secure enough to withstand basic hand movements.
The system is designed to be replicable for a piano with different key sizes.	The system is currently built to create a piano with 61 keys. This system can be modified to fit a different number of keys.
The system provides the pianist with haptic feedback indicating which finger the pianist must use within 100 ms.	Some of our members wore the gloves and felt the haptic vibrations turn on to indicate which notes to play. The delay present wasn't noticeable to the pianist therefore negligible.
The system provides the pianist with visual feedback indicating which keys to press within 100 ms.	When playing the piano, our members were visually cued on which notes to press at the indicated time. The delay present was less than 250 ms (human reaction time) therefore not visible to the pianist.

System Specifications + Verification

The system provides the pianist with auditory assistance on his/her playing within 100 ms.	Our members can hear the song selected play while they are practicing the piano. This vocal cue allows for the users to “hear” if they are playing the notes on beat. The vocal cue delay present is not noticeable to the user so it is under 250 ms.
The system is designed to have less than 100 ms of latency between the user and the system.	Our members were not able to physically distinguish a delay with the system while playing the piano. Our system is noted to have a delay of less than 250 ms. Therefore, it doesn’t hinder the user experience.
The gloves have a range of at least 5m.	Our members were able to have the system communicate with the bluetooth from across a room. The system is able to reliably play from a distance of 5 m.
The gloves have a battery life of at least 1 hour.	Our members were able to play a total of 20 songs in one sitting. The system was able to continuously play for roughly 60 minutes.

Custom PCB Specifications

(If Individual Components Printed Separately for very rough maximum estimate)

PCBs Cost

PCBs	Use	Cost
Microprocessor	Used to control the haptic motors and LED strip	\$4.00 (1 unit)
Bluetooth	Needs to be able to receive fingering data from the computer to control the haptic motors	\$4.00 (1 unit)
Power Boost	Needed in order to boost the voltage going into the system from the lithium ion battery used from 3.7 V to 5V.	\$4.00 (1 unit)

Total Costs for the PCBs

Total Cost for PCB on Gloves x2 (1 circuit for each team member)	Total Cost for the PCB for LED strip	Costs for the Components	Total Cost of PCBs for 1 Iteration
$(\$4.00 + \$4.00 + \$4.00) \times 4 = \48.00	$\$4.00 \times 2 = \8.00	\$50	$\$48.00 + \$8.00 + \$50 = \106.00

Estimated Costs for CDR

Component	Quantity	Cost
MIDI Piano Keyboard	1	\$0.00
Gloves	2 Sets (2 different styles for prototyping)	\$0.00
Haptic Vibration Actuators	10 (1 for each finger)	\$0.00
Visual Display: 1. Strip LEDs 2. Computer or Use of Tablet	<ul style="list-style-type: none"> LEDs for each key (63) 1 Computer or Use of a Tablet 	\$0.00
Lithium Ion Polymer Battery	2 (One for each glove)	\$0.00
PCBs (3 iterations)	2 (One for each glove but same circuit) <ul style="list-style-type: none"> Build microprocessor circuit with Bluetooth capabilities with power boost x2 1 (LED PCB) <ul style="list-style-type: none"> Build microprocessor circuit to control LED strip 	$\$106.00$ (each iteration) Estimated Shipping (3 iterations) = \$15.00 $\$106.00 * 3$ (3 iterations for boards) + \$15.00 = \$333.00
3D Printed Case	2 (To attach the PCB to the gloves)	\$0.00
Total		\$333.00

Gantt Chart

[illegible]

Gantt Chart Continued

[illegible]

Team Responsibilities for CDR

Prepsa

- Budget Lead
- KiCad/Altium Lead
- 3D Printing
- Embedded Code for the LED strip in C/C++
- Custom PCB to control LED strip

Neil

- PCB lead
- Custom PCB with Power Boost Design Integration
- Bluetooth System
- LED Matrix
- Team Website

Paulina

- Display Application
- Feedback with user information
- Optimizing Code
- Custom PCB to control Haptic Motors Integration
- Song and speed selection option

Megan

- Team Coordinator
- Embedded Code for the Haptic Motors in C/C++
- Pygame GUI
- Custom PCB Bluetooth Integration Design
- Glove Reconstruction

Citations

K. Huang, E. Y. Do and T. Starner, "PianoTouch: A wearable haptic piano instruction system for passive learning of piano skills," *2008 12th IEEE International Symposium on Wearable Computers*, 2008, pp. 41-44, doi: 10.1109/ISWC.2008.4911582.

C. Seim, T. Estes and T. Starner, "Towards Passive Haptic Learning of piano songs," *2015 IEEE World Haptics Conference (WHC)*, 2015, pp. 445-450, doi: 10.1109/WHC.2015.7177752.

Evening, Aleksander. "Piano LED Visualizer." *YouTube*, YouTube, 9 Apr. 2019, <https://www.youtube.com/watch?v=IZgYViHcXdM>.

<https://github.com/thegreatkwanghyeon/autofingering>

Nakamura, E., Saito, Y., & Yoshii, K. (2020). Statistical learning and estimation of piano fingering. *Information Sciences*, 517, 68-85.

Live Demo!

Questions?