The Emperor's New Instrument

Oscar Chen, Yuqi Gong, Karen Song

Department of Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—A system that synthesizes music in real time based on hand movement and gesture. The system uses computer vision to map gestures and hand positions to predefined oscillators and frequencies, wireless gloves to map hand rotations to volume and pitch bending, and a software synthesizer to play music on the laptop speaker. This project allows people to play music with great freedom in musical expressions and at an affordable price.**

***Index Terms*— Computer vision, lighting, music synthesis, real-time playing, sensor, virtual instrument, wearable circuits**

I. Introduction

THE virtual music system that we are building intends to make playing music more affordable and accessible to the general public. As long as the user has a compatible laptop and our music system, they can experience playing different instruments at a low cost anywhere and at any time they want.

Our team is planning to build a virtual music instrument controlled solely by hand movement and gestures. The left and right hand will each take on their own functions and collectively create sound of different volume and pitch. There will also be a screen showing where the user's hand is, and which corresponding note is being played at the moment. The users will also be able to apply music effects like bending and do simple commands such as pause/resume using pre-mapped gestures.

On the hardware and circuit side, there will be an autonomous lighting system to aid video capture, a pair of wireless wearable circuit gloves on users' hands to detect tilt and a central connection module that integrates sensor's data and sends it to users' laptops. On the laptop, there will be a software synthesizer that generates sounds in real time based on the collected data, which allows users to set and use different gestures for commands and music effects.

In addition our team evaluated environmental and economic factors when designing our product: we want to keep the production and maintenance cost low, minimize the energy consumption, and limit possible waste production of the system. Currently, a music system with similar functions (for example, a theremin) costs over $600. Our team intends to cut the system cost development cost to under $400 to make it further affordable.

II. Use-Case Requirements

Before designing the system, our team came up with a list of user requirements and their corresponding metrics (see section IV, design specifications)

The targeted user group of our product are people who are interested in playing music, own a windows/macOS laptop, and are comfortable and capable with setting up basic software.

In terms of human interaction, users will need our system to be fast and accurate. In terms of the speed of sound production, the delay between users changing their hands and the speaker outputting a different sound should not be humanly perceivable.

If the system keeps on misunderstanding the user's input, the user will be frustrated. Therefore another important user requirement is a higher accuracy in gesture detection. The recognition system should be 80% accurate all the time considering the variety in laptop's camera quality and previous gesture recognition research.

Also users will need the pitch and the volume of the output sound to match their expectation. An instrument will be not useful if its sound production can not be reliably controlled. First, any note played by our instrument should appear "in tune" to human ears. Secondly, the volume control needs to be both accurate and intuitive. If the user halves the left hand's pitch angle, the perceived volume must also be halved.

Besides the performance of the instrument, users are also concerned with other factors such as portability, general usability and difficulty of setup. The users need the system to be a reasonable size to carry around, therefore the system should not weigh more than a typical laptop.

Users might prefer to use the musical systems in areas that have dim lighting, the system is therefore equipped with an autonomous LED to ensure the user can enjoy the system in any environment that's brighter than 25 lumens, which is almost any lighting setting. In addition, the batteries on the gloves need to last long enough for the users to enjoy a music session without being interrupted in the middle. Therefore our gloves should be able to function at least 5 hours before they need to be recharged again.

Before enjoying all the music, the users must be able to set up the system on their computers first. Our group ended up deciding that our product will be supported both on Windows and macOS. In order to make the system accessible to the users, the success rate of setting up on either system should be over 90%.

III. Architecture and/or Principle of Operation

A simple block diagram that lists the three subsystems of our project is shown in Fig. 1. For a more detailed block diagram that lists the individual components within a subsystem, please refer to Fig. 5 on page 3. Our system is composed of three major subsystems: wearable gloves, receiving module, and software running on a laptop. The first subsystem, the wearable gloves, consists of two identical circuits mounted on a pair of gloves. Each wearable circuit contains a 3-axis gyroscope IC, an Arduino Nano, a radio transmitter, and a LiPO battery. The wearable gloves will detect the rotation angles of both hands and send them to the receiving module via 2.4GHz radio. The second subsystem, the receiving module, contains a radio receiver, an Arduino Uno, a light sensor, and a LED module. It has two functions. First, it automatically adjusts the brightness of the LED to better light up the user's hands. Secondly, it collects the angle information via radio and forward it to the laptop via USB. The third subsystem, the software running on a laptop, synthesizes sounds based on the video feed of one's right hand and the angle information sent by the receiving module. The software system contains four parts, a demux, a gesture recognition program, a hand tracking program, and a synthesizer. It also collects video feed from the laptop's camera and writes audio streams to the laptop's speaker.

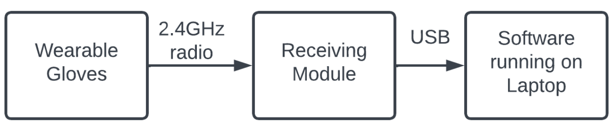


Fig. 1. Subsystem block diagram.

A. Wearable Gloves

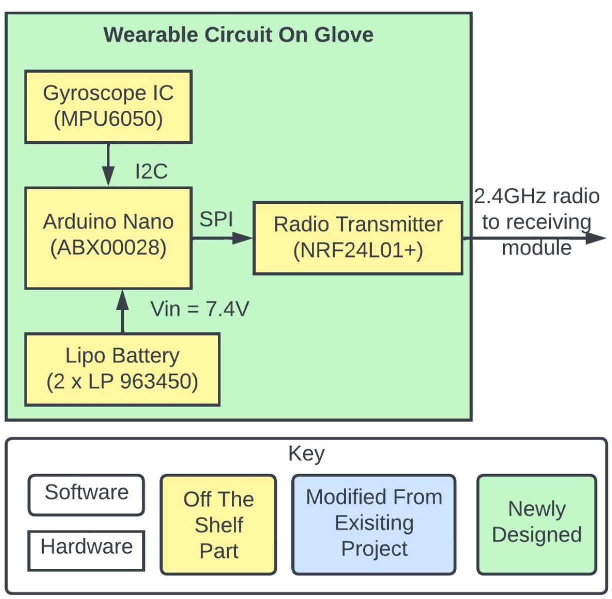


Fig. 2. Block diagram of one glove circuit.

The wearable gloves (see Fig. 2 above) are two identical wearable circuits that collect the 3-axis angular positions of both hands and send it to the receiving module. Since it is a fully isolated module, the circuit on each glove is powered by a rechargeable LiPO battery. A gyroscope IC is used to detect the 3-axis angular positions, a radio IC is used for sending the position data to the receiving module, and an Arduino Nano is used to integrate everything together.

B. Receiving Module

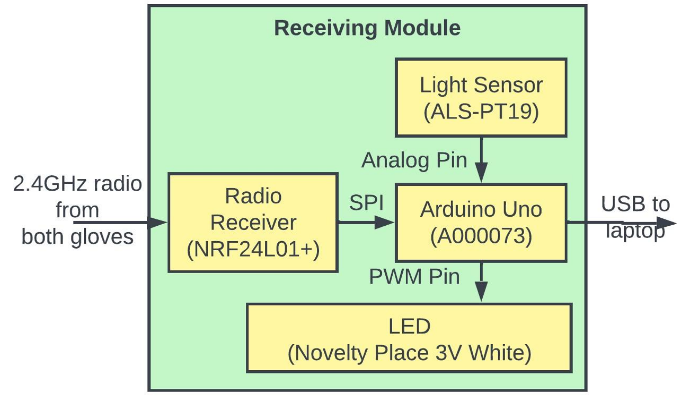


Fig. 3. Block diagram of the receiving module.

The receiving module (see Fig. 3 above) has two functions. First, using a radio receiver and an Arduino Uno, it can collect the data sent by both gloves and forward it to the laptop via an USB cable. Secondly, a light sensor and an adjustable LED enables autonomous light filling to aid the video capture of the user's hand movement and gestures.

C. Software running on laptop

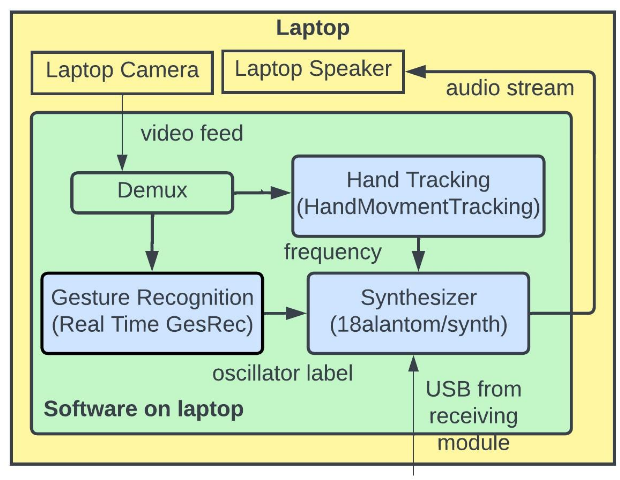


Fig. 4. Block diagram of the software running on laptop.

The software running on the laptop (see Fig. 4 above) is designed to synthesize sound in real time based on video feed from the laptop camera and angular position information from the receiving module. It has four major components, a demux, a gesture recognition program, a hand tracking program, and a synthesizer. The demux allows the hand tracking program and the gesture recognition program to run in a time-multiplexed manner. The gesture recognition allows the user to select an oscillator using gestures. The hand tracking program allows the user to control pitch through hand movement. The synthesizer will first map USB data to volume and pitch bending. Then, with the oscillator and frequency information, it will generate a sound and write to the laptop's speaker.

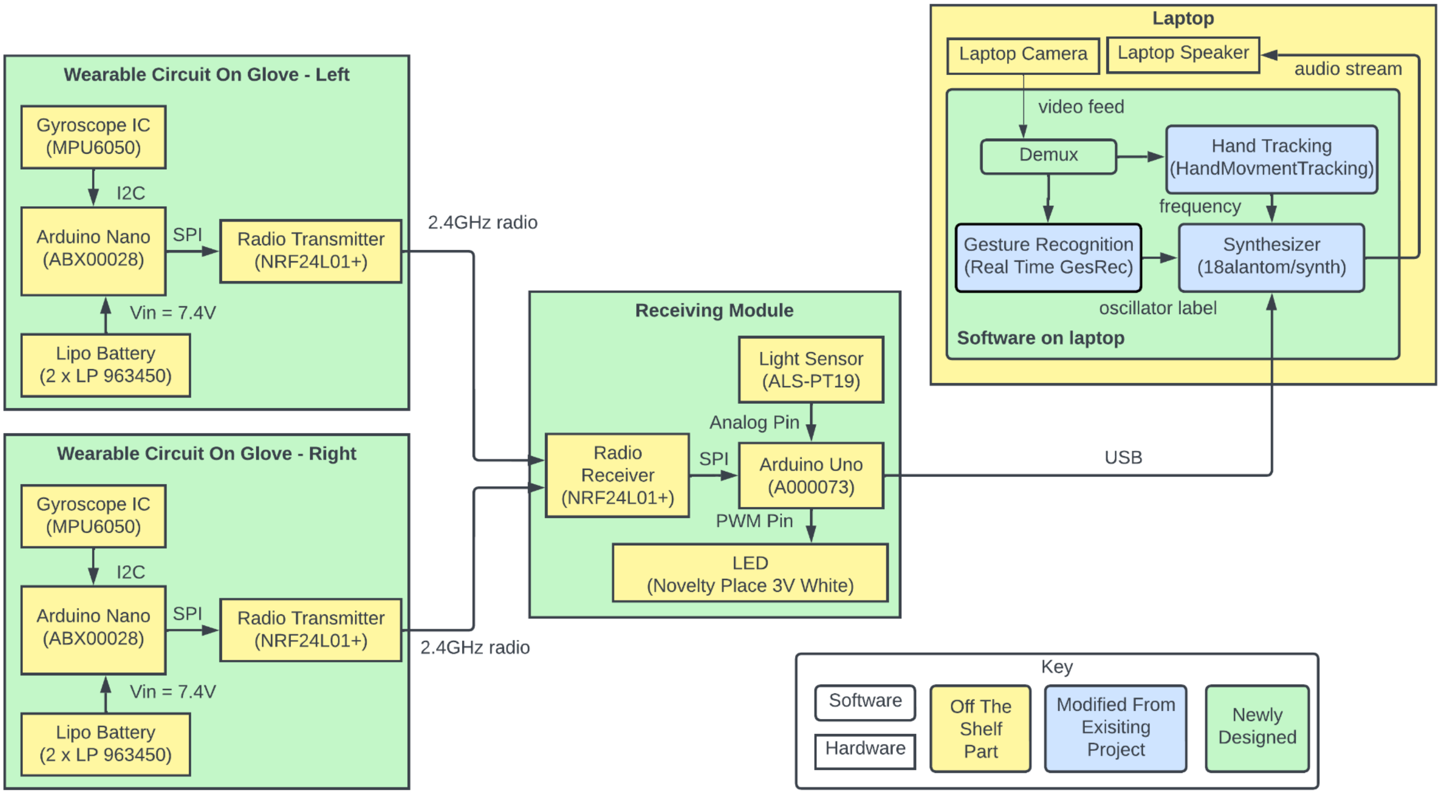


Fig. 5. Block diagram of the software running on laptop.

IV. Design Requirements

In order to ensure there is no perceivable sound production delay, the latency between users moving their hands and a change in the sound output should be less than 10ms. We chose this number based on a past study on hearing aids' latency. The researchers reported that 10ms is the threshold where sound latency becomes noticeable and disturbing.

We also hope that our gesture recognition system will be 80% accurate. This includes all the gesture detection used to select an oscillator before playing as well as to pause the resume playing. Although we want to maximize the accuracy, after learning that a finely tuned CNN usually has a gesture recognition accuracy of 87% and a wearable gyroscope based system has an accuracy of 90%, we decide to scale down our accuracy metric to 80% due to software and hardware limitations.

In order to ensure that our instrument sounds "in tune", the produced sound of a given note should be within 3.6Hz of the desired frequency. We chose this number because this is the reported frequency resolution of the human ear.

Since the user is controlling the volume of the instrument using the pitch angle of his or her left hand, we also want the volume control to be intuitive and accurate. Specifically, halving the pitch angle should halve the volume (or in the decibel world, a decrease of 10 dB). We want our instrument to decrease the sound output by 10dB $\mp$ 0.5dB whenever the pitch angle detected by the gyroscope IC is halved. Because of the difference in laptop speakers, we cannot determine the maximum volume supported by our instrument. However, we need to ensure that no sound is being produced when the pitch angle is 0° (hand is parallel to the ground) .

In terms of portability, since we want our users to be able to carry our instrument around, we expect the total weight of our instrument (excluding the laptop) to be less than 4.8 pounds (the weight of a 16-inch Macbook Pro).

In terms of battery life, we want our gloves to function for at least 5 hours after the LiPO batteries are fully charged. We chose this number because it is recommended to eat one's meal between 4-5 hours apart. Realistically, we expect our user to play for only 1.5-2 hours in one sitting, as this is the typical length of a music rehearsal.

Since it is very possible for our user to perform or practice in a dark environment, we expect our instrument to function in environments as dark as 25 lumens, which corresponds to the brightness of a room lit up by a candle. Realistically, light sources like desk lamps, laptop screens, and ceiling lights can easily bring the environment brightness over 100 lumens.

Finally, to ensure our instrument's accessibility, we expect it to be both Windows and macOS compatible. We will conduct user experiments on both OSs and expect the success rate of initial setups to be over 90%.

V. Design Trade Studies

A. Wireless protocol for wearable gloves

Since we are transmitting angle information wirelessly from the wearable gloves to the receiving module, we mainly considered the latency, power consumption, and cost of three different protocols: WiFi, Bluetooth Low Energy, and Radio. For each protocol, we found the most popular chip that is supported by the Arduino library and compared their specifications (see Table 1 below).

Table I. Comparison between three wireless protocols

Protocol	Chip	Operation current (mA)	Latency(ms)	Unit Cost
WiFi	ESP8266	70	2-100	$7.50
BLE	HC-10	30	6	$10
Radio	NRF24L01+	12	5	$5

In terms of latency, theoretically ESP8266 has the lowest among the three. However, people have reported that the latency is usually not steady and depends on network traffic. In the worst case, the latency can be up to 100ms, which would violate our design requirement. Moreover, since we want to meet the 5-hour battery life requirement, the wireless transmitter needs to draw as little current as possible. ESP8266, being the most power-hungry of the three, is not the optimal choice.

In terms of latency, HC-10 and NRF24L01+ are about the same. However, HC-10's draw about twice as much current during operation. Most importantly, a HC-10 receiver can not listen to multiple HC-10 transmitters at the same time. This means that we need 4 HC-10 in total to collect angle information from both gloves. Not only does this drive up the total cost, it introduces complexity to the receiver module's circuit. As a result, HC-10 is not the optimal choice either.

NRF24L01+ ranks best in all specifications. Most importantly, NRF24L01+ supports building a radio network, where each NRF24L01+ can receive signals from 5 NRF24L01+ transmitters. As a result, we only need three NRF24L01+ (two transmitters, one receiver) to collect data from both hands. In summary, choosing it as our wireless transmitter will help us the most in meeting the 5-hr battery life requirement, the 10ms latency requirement, and keeping the cost of our instrument low.

B. Battery choice for wearable gloves

Since our wearable gloves are not connected to the laptop, all of its components need to be powered by batteries. Since both the radio and gyroscope chip are drawing power from the Arduino Nano, we need to find a suitable battery to power the Arduino. The most obvious choices are dry-cell battery, lithium-ion battery, and LiPO battery. Since we do not want our users to constantly change the batteries on their gloves and create unnecessary chemical waste, we quickly ruled out dry-cell batteries. Dry-cell batteries usually do not support recharging and have a lower capacity compared to LiPO batteries. For example, a typical 9V battery has a capacity of 500 mAh, where a 3.7V LiPo battery has a capacity of 1800mAh.

Between lithium-ion batteries and LiPO batteries, although lithium-ion batteries are cheaper and have a higher power density than LiPO batteries, they are less safe and age faster than LiPO batteries. Specifically, lithium-ion batteries can catch on fire easily if they are punctured. Since the batteries are attached to the gloves, this is an immense safety risk we are not willing to take in exchange for a lower price and a higher power density. However, there is one complication to using LiPO batteries: their output voltage is usually around 3.7V, where the Vin for Arduino Nano needs 7-12V. This means that each glove needs to be powered by two LiPO batteries put in series. Although this seems like it will make the gloves heavier, LiPO batteries usually come in different sizes and weights. As a result, we can easily create a 7.4V Vin with two small LiPO batteries.

C. Synthesizer choice

The major part of sound production lies within the synthesizer, and we need to decide between using a pre-existing synthesizer or building one from scratch. Although using a pre-existing synthesizer will enable users to do more musical effects, it introduces many problems. First, it is hard to find an open-source synthesizer that is both Windows and macOS compatible. After finding one, we still need to read its source code and figure out how to pass in our inputs to its interface. Not only is this a time consuming task, it makes fixing latency problems virtually impossible. As a result, developing our own synthesizer is more preferable. Since both our gesture recognition program and the hand tracking program are written in Python, it is very easy to write the output of these modules to the synthesizer with low overhead. Also, fixing Python library compatibility issues is also easier than fixing compatibility issues of an open-source software written by someone else.

VI. System Implementation

A. Wearable Gloves

Each hand glove consists of two LiPO batteries, an arduino nano, a gyroscope (MPU6050) and a radio transmitter (NRF24L01+). The two LiPO batteries will be in series and connected directly to the arduino and provide a voltage at about 7.4 Volts. The arduino will be the power supply for MPU6050 and NRF24L01+. MPU6050 is used to measure the angular position and NRF24L01+ is used to send the angle to the computer.

When the user raises his/her left hand, the gyroscope will measure the pitch angle and the transmitter will send the data to the computer. The volume will be increased from 0 to 100 percent as the pitch angle increases from 0 degrees to 90 degrees.

When the users' right hand is raised, the angle will also be transmitted to the computer by NRF24L01+ and it will act as a pitch bend. The pitch will be raised or lowered smoothly based on the angle. If the angle is negative, the pitch will be lowered according to the absolute value of the angle. Otherwise, the pitch will be increased. The angle of positive 90 degrees will bend 2 semitones up and the angle of negative 90 degrees will bend 2 semitones down.

The component layout of the wearable circuit glove is shown in figure 6.

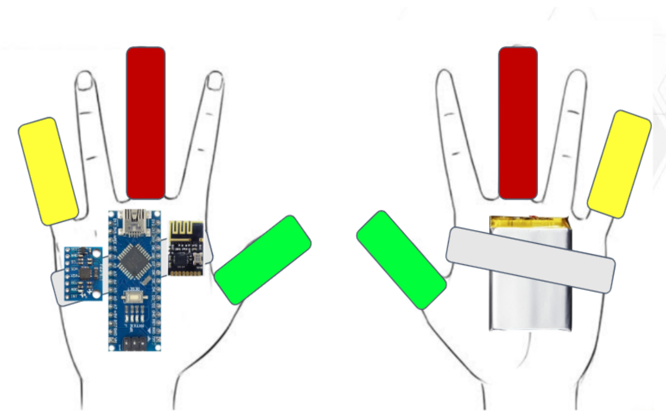


Fig. 6. Diagram of the dorsal side of the left hand glove (left) and the palm side of the left hand glove (left). Right hand glove is identical

B. Receiving Module

The receiving module is composed of two subsystems: the receiving component and the autonomous lighting system, connected on an arduino uno.

The receiver component consists of a NRF24L01+ radio receiver, which can listen to at most 5 different NRF24L01+ radio transmitters. It will capture the information from the wearable circuit gloves, and send the information to the arduino uno which will later transmit the data through a USB port to the laptop and into the synthesizer.

The autonomous lighting system contains a ALS-PT19 light sensor located near the user's hands, that will collect data on the lighting level near the CV's observational target, and transmit the data to the arduino uno through the analog pin. The arduino will then send a command to an LED (NoveltyPlace 3V white) after analyzing whether extra lighting to the system will be needed or not.

C. Software running on laptop

The software component of the system consists of mainly three parts, the hand moving tracking model (based on *HandMovingTracking*), the gesture recognition model (based on *Real-time GesRec*), and the synthesizer (based on *18alantom/synth*). All of these components are developed based on altering preexisting code bases to meet our design requirements. These models are carefully selected after our team browsed github and examined dozens of similar models.

For the gesture recognition model we will only be using a subsection of the gestures provided in the dataset since there are not as many commands and music effects that our music system needs. This subset of gestures will contain gestures that are relatively unique and easy to be detected, such as holding a fist or pointing a thumb up. Figure 7 below shows how *Real-time GesRec* generates a classification output for each individual gesture inside its database.

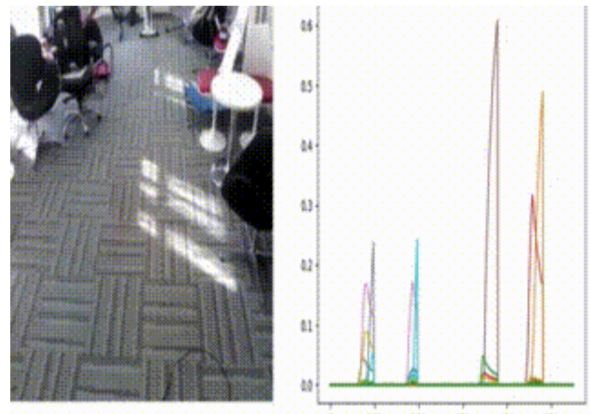


Fig. 7. Real-time GesRec shadowing classification scores (right) of each gesture presented on the video (left)

The hand tracking models will mainly be used to identify which of the thirty-two note areas the majority of the user's pitch controlling hand (right hand) is lying in (depicted in Figure 8). First of all it will identify which hand in the video feed is the user's right hand, then classify which area on the screen does the hand fall into. This screen division information will also be shown on the laptop during the user's session on the video feed.

Fig. 8. Screen area splitting for different pitches on a typical 16:9 laptop screen (16 * 2 sections)

The synthesizer will first load a pre-defined oscillator class based on the gesture detected by the gesture recognition program. When the user is playing, the hand tracking program will pass in the note frequency that corresponds to the user's hand position. The USB cable from the receiving module will pass in the volume and pitch bend information. Using the note frequency and the pitch bend information, the synthesizer will generate a numpy array that represents the oscillator oscillating at the corresponding frequency, multiply the array by the volume, and write to the audio stream buffer by buffer. When the user changes to a different note, the synthesizer will gradually “glide” to the new note by gradually changing the frequency. The glide duration can be changed before playing for a different sound effect.

VII. Test, Verification and Validation

A. *Tests for Gesture Recognition Accuracy.*

To test for the system’s gesture recognition accuracy, our team will input a random sequence of all gestures and run the recognition test for all of them to get the recognition accuracy. The recognition accuracy test is considered successfully passed if the recognition test accuracy is greater than or equal to 80%. This test will be repeated multiple times for a more accurate recognition result and to better analyze reasons behind the failed test cases.

If the recognition test results are not ideal, our group will continue to revise the system by examining the failed recognition cases and identify what could be the cause of the failure. If the results are too far from the intended goal, our team will consider switching to a different model.

B. *Tests for Sound Production Latency*

To test for the system’s sound production latency, our team will use a slow-motion camera to measure the time elapsed between the gesture or movement input and the sound production output. The sound production latency test is considered passed if the time between the input trigger and the sound output is less than 10 ms.

If the sound production latency test results are not ideal, our group will consider swapping to faster software or hardware components.

C. *Tests for Pitch Production Accuracy*

To test for the pitch accuracy of the system’s sound output, our team will use a spectroid (audio spectrum analyzer) to test the output music pitch accuracy. The sound production pitch accuracy test is considered passed if the pitch of the output sound is within +-3.6Hz of the desired pitch.

If the pitch production accuracy test results are not ideal, our group will first retest it on multiple different computers to ensure this is not a problem specific to a certain laptop / model, and decide which part of the sound production software needs to be revised.

D. *Tests for Volume Production Accuracy*

To test for the volume accuracy of the system’s sound output, our team will use a decibel meter to test the output music volume accuracy. The sound production volume accuracy test is considered passed if halving the tilt angle reduces the sound by 10dB, with an error range of +- 0.5dB. The predetermined volume level will vary depending on the built in speaker of the computer. However, we will test that a 0° pitch angle means no sound is being produced.

If the volume production accuracy test results are not ideal, our group will first retest it on multiple different computers to ensure this is not a problem specific to a certain laptop / model, and decide which part of the sound production software needs to be revised. Worse case we will consider adding an external speaker to the system。

E. *Tests for Overall System Weight*

To measure the overall weight of the system, our team will put the entire system together on a scale and record the weight. If the weight of the system is less than 4.8 pounds, our system has passed the weight/size test. If the entire system is overweight, our group will consider using a smaller LiPO battery instead.

F. *Tests for Lighting System*

To measure the effectiveness of the lighting system, our team will use a light meter / light intensity app to measure the brightness of the room while running the system (particularly the hand tracking and gesture recognition function) in a dark environment This test is successful if we can obtain over 80% accuracy in any environment with brightness above 25 lumens.

If this test fails, our team will use more LEDs / use a brighter LED to better light the room.

G. *Tests for Battery Life*

To measure the amount of time the gloves will work before it runs out of power, our team will perform a Battery-life test in order to determine the gloves’ battery life. In this test we will record the time for the lipo batteries to fall below 7V (nano’s Vin threshold) using a timer and a voltmeter.

If this measured time is less than 5 hours, our team will consider switching to larger batteries, add more batteries or revise the system such that it drains less battery.

H. *Tests for Plug and Playability*

To test the plug and playability of the system, our team will ask multiple participants to set up our developed system on macOS and windows following the documented instructions, and measure the success rate of setting up the system. Our team will try to recruit participants with different laptop models to ensure compatibility.

If a setup process fails, our team will try to fix it, add this solution to our installation workflow and update the documentation’s debugging section.

VIII. Project Management

A. *Schedule*

The entire project spans a little more than 2 months. Our team divided the project into three smaller sub-parts: design, development and testing, and final integration.

Specific schedule is shown in Fig 9.

B. Team Member Responsibilities

Based on the different areas of concentration in their studies, team members are assigned different tasks. There are several overlapping tasks in order to speed up the process and make integration later on easier. The basic task assignment is listed below:

- Oscar is working on the synthesizer and arduino design and implementation,
- Yuqi is working on the wearable circuit and arduino design and implementation
- Karen is working on designing and developing the hand monitor system.

Team member responsibilities / task breakdown could also be found in figure 9.

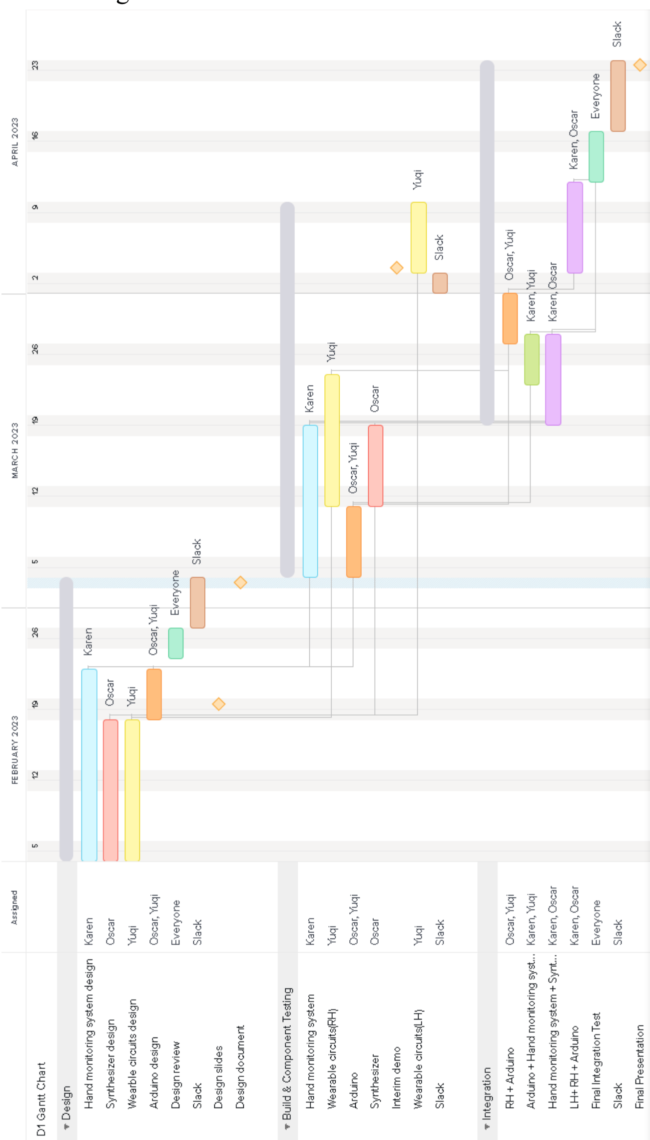


Fig. 9. Team schedule with milestones and team responsibilities

C. Bill of Materials and Budget

The total assigned budget for this project is $600, the goal of our group is to limit the developmental cost under $400. A more detailed list of the purchased items along with a description (including the model number, manufacturer, and cost) can be found in table 2 on the last page.

D. Risk Mitigation Plans

The most significant risk originates from the hand gesture and movement detection system. As a result, our team splits the video feed so that our program will either perform gesture recognition or hand tracking at once. Splitting the input into functionalities allows us to access more pretrained models and hand tracking programs, which greatly reduces the development time and increases detection accuracy.

Another risk lies within our hardware. Because the quality of ICs (especially the tilt sensor and the radio chip) is not guaranteed, our team will start component testing immediately after our chips arrive. If there is a defect, we will order another one or from another seller within one day. We will also try to test if these components meet our design requirements, and purchase additional components/ different models if needed.

IX. Related Work

The source of inspiration for the designing of this project is from a youtube video posted by youtuber musician Grégoire Blanc. In this video he plays the song Clair de Lune on a Theremin – a very special musical instrument invented by a Russian scientist Leon Theremin in the 1920s. In this video, Blanc moves his hands in front of a device, then music of different volume and pitch is created.

The Theremin system uses two antennas to generate an electro-magnetic field that can create a tone. It's the only musical instrument that has no strings, keys or pipes, and is played without touching the actual object. There are multiple videos explaining how the theremin works and how to play it. In one of these videos, the channel owner interviewed one of the best Theremin musicians, Carolina Eyck, who also provides a demo.

The Theremin gave our team the inspiration to create a touchless musical system that can be played by simply moving your hands in the air. Instead of magnetic fields, our team decided to go along with a different approach, a more modern approach to sound production.

X. Summary

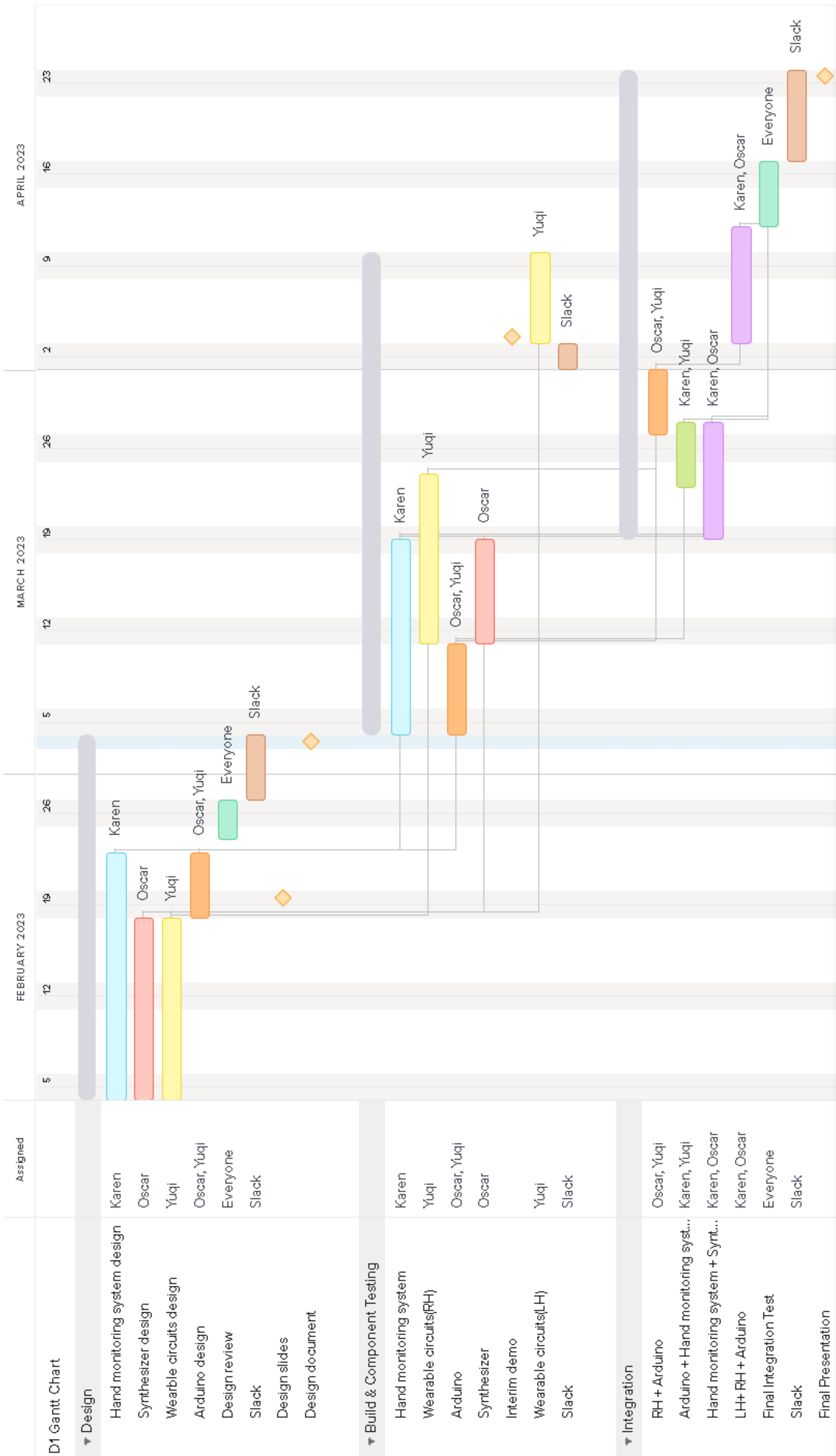
The goal of our musical system is to use signals to generate music, and to control the music generated by hand movement and different gestures, detected and recognized by the combination of CV and wearable circuitry. The system will allow music to be more accessible to anyone who is interested, but also reduces the commitment level to stick to a particular instrument. The goal of the system is to promote music and increase the happiness level of people who are already interested in this area. There are technical challenges in both the hardware and software system, but all of them can be mitigated through early testing and back up planning.

Glossary of Acronyms

CV - Computer Vision
LED - Light-emitting Diode
LiPO battery - Lithium-polymer Battery
USB - Universal Serial Bus

References

[1] Gunduz, A et al.(2019) Real-time-GesRec[Source code] https://github.com/ahmetgunduz/Real-time-GesRec
[2] Takahashi, K (2020) hand-gesture-recognition-using-mediapipe[Source code]https://github.com/Kazuhito00/hand-gesture-recognition-using-mediapipe
[3] Bahadur, A (2017) HandMovementTracking[Source code]https://github.com/akshaybahadur21/HandMovementTracking
[4] Handtracking.io(2021)Yoha[Source code] https://github.com/handtracking-io/yoha
[5] Surge Synth Team(2018) Surge[Source code] https://surge-synthesizer.github.io/
[6] Alan, Ruiz, T (2021) Synthe[Source code] https://github.com/18alantom/synth
[7] "Debussy - « Clair De Lune » on the Theremin." Youtube, uploaded by GréGoire Blanc, 22 Dec. 2013, www.youtube.com/watch?v=PjnaciNT-wQ&ab_channel=Gr%C3%A9goireBlanc.
[8] "Theremin - The World's Strangest Instrument Explained By @Carolinaeyckvideos." Youtube, uploaded by DW Euromaxx, 3 Feb. 2021, www.youtube.com/watch?v=SsQloi46AQU&ab_channel=DWEuromaxx.
[9] "Steve Vai Guitar Lesson - Bending Notes - Alien Guitar Secrets: Passion & Warfare." Youtube, uploaded by TrueFire, 14 Jul. 2016, www.youtube.com/watch?v=uBINGNIFnGI&ab_channel=TrueFire.


D1 Gantt Chart
Assigned
FEBRUARY 2023
MARCH 2023
APRIL 2023
5
12
19
26
5
12
19
26
2
9
16
23
Design
Hand monitoring system design
Karen
Synthesizer design
Oscar
Wearble circuits design
Yuqi
Arduino design
Oscar, Yuqi
Design review
Everyone
Slack
Slack
Design slides
Design document
Build & Component Testing
Hand monitoring system
Karen
Wearable circuits(RH)
Yuqi
Arduino
Oscar, Yuqi
Synthesizer
Oscar
Interim demo
Wearable circuits(LH)
Yuqi
Slack
Slack
Integration
RH + Arduino
Oscar, Yuqi
Arduino + Hand monitoring syst...
Karen, Yuqi
Hand monitoring system + Synt...
Karen, Oscar
LH+ RH + Arduino
Karen, Oscar
Final Integration Test
Everyone
Slack
Slack
Final Presentation

Item	Quantity	Per Cost	Total Cost
LiPO 963450 Battery	5	$16.99	$84.95
MPU6050 Gyroscope	3	$3.33	$9.99
Arduino Nano	3	$12.75	$38.25
Radio Transmitter NRF24L01+	10	$1.20	$11.99
Light Sensor	1	$2.50	$2.50
LED White	100	$0.06	$5.95
Total Cost	**---**	**---**	**$153.63**

TABLE II. BILL OF MATERIALS