ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

Virtual Band Gloves

Team #34

HENGYU LIU
(hengyu2@illinois.edu)
XUAN TANG
(xuant4@illinois.edu)
HAN CHEN
(hanc7@illinois.edu)
ZHANPENG LI
(zl88@illinois.edu)

TA: Zheyi Hang

Friday 31st May, 2024

Contents

1 Introduction

1.1 Purpose

1.1.1 Problem Statement

The advent of digital technology in music has revolutionized the way we create, perform, and interact with musical instruments. Traditional instruments, while offering rich sound and tactile feedback, often come with limitations related to size, portability, and accessibility. Instruments like the piano or double bass are not only cumbersome to transport but also require significant maintenance and careful handling due to their fragility. This presents a challenge for musicians who wish to practice or perform outside of traditional settings or for those who lack the space to house such instruments. The need for a solution that offers the tactile and auditory richness of traditional instruments, combined with the convenience and portability of digital technology, is evident.

1.1.2 Solution Overview

The Virtual Band Gloves project aims to address these challenges by introducing a novel instrument interface that utilizes cutting-edge audio processing, pressure sensing, and positioning systems to convert electrical pulses into high-quality sound outputs of various instruments. Unlike previous attempts at digital musical interfaces, which often fell short in replicating the depth and nuance of real instrument sounds or the feel of playing them, our solution promises an authentic and dynamic music performance experience. By integrating motion tracking, audio processing, and sensor technology, the Virtual Band Gloves provides a portable and versatile platform for music creation through the analysis of hand movement and pressure sensitivity. This unique purpose positions the Virtual Band Gloves at the forefront of music technology innovation, showcasing a significant advancement over existing musical technologies and applications.

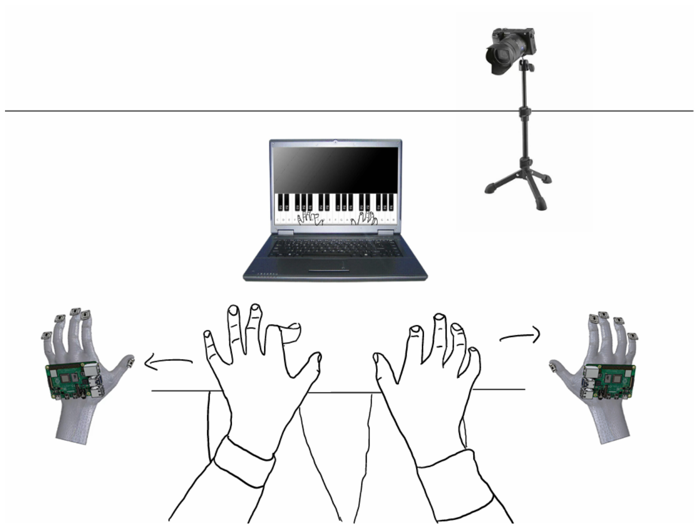


Figure 1: Product in Context[7]

Our virtual band gloves is designed to enable convenient instrument playing, which will redefine how we interact with digital environments. To help visualize what context that Virtual Band Gloves will be put into, we created the Figure 1 to illustrate how our product is used. These gloves boast state-of-the-art 6 Degrees of Freedom (DOF) position sensors on every finger and the back of the hand, enabling unparalleled precision in capturing hand movements and poses. With a Raspberry Pi unit integrated into one glove, our design ensures stable and high bandwidth data transmission to a computer. Complementing this, an industrial-grade camera captures high-resolution videos of hand gestures, providing invaluable additional data for analysis. Our focus extends beyond mere functionality; we're committed to delivering an intuitive user experience. Through sophisticated algorithms, the computer instantly interprets and displays real-time hand poses on a user-friendly interface. But our innovation doesn't stop there. These gloves are also designed to respond dynamically, triggering tailored sounds through integrated speakers based on finger pressure sensor feedback. In essence, our project represents a leap forward in motion capture technology, promising immersive interactions and unlocking new possibilities for digital engagement.

1.2 Functionality

1. **Leveraging object tracking algorithm based on data transferred from sensors to accurately compute the position of hand and motion of fingers.**

   Our project aims to utilize advanced object tracking algorithms that leverage data from sensors to precisely determine the hand's position and the movement of fingers. By integrating sensors into the gloves, we can capture real-time data on hand movements and finger gestures. This functionality is crucial for accurately translating the user's actions into corresponding musical inputs. The precise tracking of hand and finger motions ensures that the virtual instrument responds accurately to the user's commands, providing a seamless and intuitive playing experience.

2. **Using computer vision based algorithms and images transferred from camera to capture the position of hand and motion of fingers.**

   In addition to sensor data, our project incorporates computer vision-based algorithms to analyze images captured by a camera. This enables us to capture detailed information about the position of the hand and the motion of fingers with high accuracy. By combining sensor data with visual information, we can enhance the precision and reliability of our gesture recognition system. This functionality ensures that the virtual instrument can accurately interpret the user's hand movements and translate them into musical outputs, enhancing the overall user experience.

3. **We can produce different sound levels based on corresponding pressure level collected by the pressure sensor at the finger pulps.**

   Our project includes pressure sensors embedded in the gloves, allowing us to measure the pressure exerted by the user's fingers on the virtual instrument's interface. By analyzing the pressure levels detected by these sensors, we can dynamically adjust the sound levels produced by the virtual instrument. This functionality enables users to express themselves musically with greater nuance and control, mimicking the tactile feedback of traditional instruments. By incorporating pressure sensitivity into the virtual instrument interface, we enhance the realism and expressiveness of the playing

4. **We display the position and posture of the hand on the screen in real-time as the player's visual reference.**

   To provide users with visual feedback and guidance while playing the virtual instrument, our project includes a real-time display of the hand's position and posture on the screen. This visual representation serves as a reference for users, helping them to better understand how their hand movements correspond to the musical outputs produced by the virtual instrument. By providing real-time visual feedback, we enhance the user's ability to control and manipulate the virtual instrument, improving their overall playing experience and facilitating learning and mastery of the instrument.

5. **We can switch piano into another instrument and play music and reproduce the timber of different instrument.**

   One of the key features of our project is its versatility in emulating different musical instruments. Users can switch between different instrument modes, such as piano, guitar, or drums, to explore a variety of musical styles and genres. By reproducing the timbre and characteristics of different instruments, our project offers users a rich and diverse musical experience. This functionality allows users to experiment with different sounds and textures, unleashing their creativity and enabling them to create music that reflects their artistic vision. Overall, the ability to switch between instrument modes enhances the versatility and appeal of the virtual instrument, making it a valuable tool for musicians and music enthusiasts alike.

1.3 Subsystem

1.3.1 Block Diagram

The Virtual Band Gloves is consists of 6 subsystems, which are Computer Vision based Position and Motion Detection Subsystem, Sensor based Position and Motion Tracking Subsystem, Pressure Sensor based Sound Level Determination Subsystem, Power Supply, Instrument Virtual Model, and Module Communication and Data Processing Subsystem. The inter-connection are showned in the Block Diagram which is Figure 2.

1.3.2 Subsystem Description

1.3.2.1 Computer Vision-based Position & Motion Detection:

This subsystem utilizes computer vision techniques to accurately track finger positions and movements. It leverages the MediaPipe Hand Landmarker model, which detects

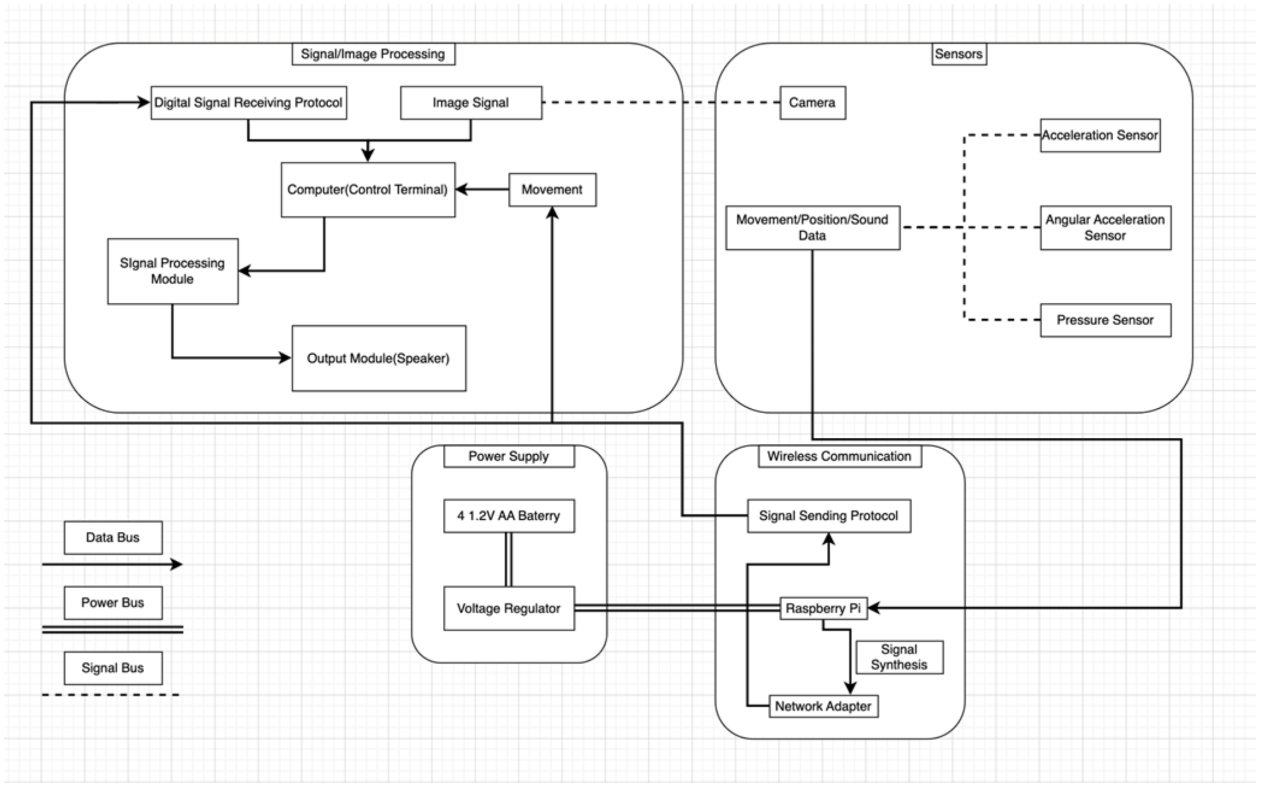


Figure 2: Block Diagram for Virtual Band Gloves[1]

and tracks 21 3D landmarks of the hand in real-time. Additionally, it addresses the challenge of recognizing gloved hands through data augmentation and pre-processing techniques.

1.3.2.2 Sensor-based Position & Motion Tracking:

This subsystem integrates sensors, such as the MPU6050 or JY931, to capture the translational and angular acceleration of the musician's hands. The Raspberry Pi 4B acts as the microcontroller, processing sensor data and facilitating communication with the central computer. The integration of sensors and data preprocessing algorithms ensures accurate and reliable tracking of hand movements.

1.3.2.3 Pressure Sensor-based Sound Level Determination:

Pressure sensors embedded in the gloves measure the pressure exerted by the fingers, allowing for dynamic adjustments to sound levels. The subsystem includes a comprehensive calibration process to accurately correlate pressure levels with audio volume. By integrating pressure data with audio processing algorithms, the system delivers responsive and nuanced musical outputs.

1.3.2.4 Instrument Virtual Model:

This subsystem models the virtual piano interface, providing visual aids for locating keys and ensuring precise fingertip positioning. It incorporates a camera and laser positioning module to assist in tone localization and hand recognition technology for enhanced accuracy. Audio processing algorithms correlate finger positions with key presses, delivering authentic and expressive musical notes.

1.3.2.5 Module Communication and Data Processingthe adve:

This subsystem facilitates communication between sensors, the Raspberry Pi, and the central computer. Dual-protocol communication using HTTP and gRPC ensures data integrity and low latency. Advanced noise filtering and data preprocessing algorithms enhance the reliability of sensor data, while synchronization mechanisms maintain consistency across components.

1.3.2.6 Power Supply:

The power supply subsystem ensures stable and reliable power delivery to all components of the system. It utilizes high-capacity 18650 lithium-ion batteries and provides multiple power output interfaces for peripherals and devices. By supplying consistent power, this subsystem ensures uninterrupted operation of the Virtual Band Gloves.

Interconnecting these subsystems enables the Virtual Band Gloves to accurately capture hand movements, translate them into musical inputs, and deliver an immersive music-playing experience. Through the seamless integration of computer vision, sensor technology, audio processing, and communication protocols, the project achieves its goal of revolutionizing music creation and performance in digital environments.

2 Design

2.1 Computer Vision based Position & Posture Detection Subsystem

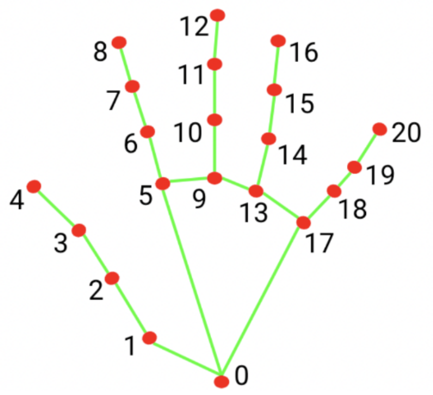
The computer vision system is aiming for accurately tracking the positions of fingers wearing gloves. The system will utilize computer vision techniques and algorithms to achieve real-time tracking with high accuracy and low latency, supporting various features for virtual band performing and further modelling the keyboard keys virtually. The visual part consists of

2.1.1 MediaPipe Hand Landmarker

MediaPipe is an open-source framework developed by Google, designed for building multimodal (audio, video, time series, etc.) applied machine learning pipelines. It offers cross-platform, customizable ML solutions for live and streaming media. One of the notable components within MediaPipe is the Hand Tracking solution, which employs machine learning to perform real-time hand detection and tracking.

The MediaPipe Hand Landmark model detects and tracks 21 3D landmarks of a hand in real-time. This includes the position of the wrist, the knuckles, the fingertips, and the joints in between. The model utilizes a two-part process: a palm detection model that operates on the full image and proposes hand-boundaries, and a hand landmark model that operates within those boundaries to identify and track the hand landmarks.

The hand landmarker model bundle contains a palm detection model and a hand landmarks detection model. The hand landmark model bundle detects the keypoint localization of 21 hand-knuckle coordinates within the detected hand regions.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 3: MediaPipe Hand Landmarks[4]

By getting the landmarks of the palm of the hand, we are able to further calculate distances between hands and the camera and model our hand fingers in specific domains. We can even detect gestures by MediaPipe model to add more features for our virtual band gloves.

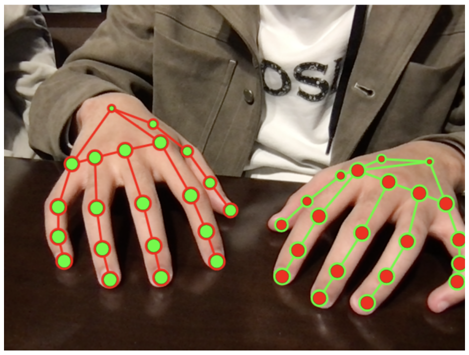


Figure 4: MediaPipe Hand Landmarker Demo[3]

2.1.2 Gloved Hands Detection

To bridge the gap between the appearance of bare hands and gloved hands, you can employ data augmentation techniques on your dataset of gloved hands. This involves artificially expanding your dataset with modified versions of your existing images to simulate the variety the model has been trained on.

Given that the MediaPipe hand tracking model was trained on a diverse dataset of approximately 30,000 real-world images, as well as rendered synthetic hand models superimposed on various backgrounds, adapting it to recognize gloved hands—a scenario significantly different from its training data—poses a notable challenge.

2.1.3 Pre-processing Gloved Hand

Since gloves can have different colors than human skin, adjusting the color space of your training images to include a broader spectrum could help. Color space adjustments involve modifying the input images in ways that alter their color properties, making the differences between various colors less pronounced or adjusting them to resemble the color distribution the model was trained on more closely. This is particularly useful when the model needs to generalize from its original training on bare hands to recognizing gloved hands, which might appear in a wide range of colors and textures.

Implementing these adjustments involves preprocessing your images before they are input into the training or inference process. By adjusting the color space of the image there's some color tunnels that would fit in the MediaPipe Model. All the landmarks is shown with the gloves on.

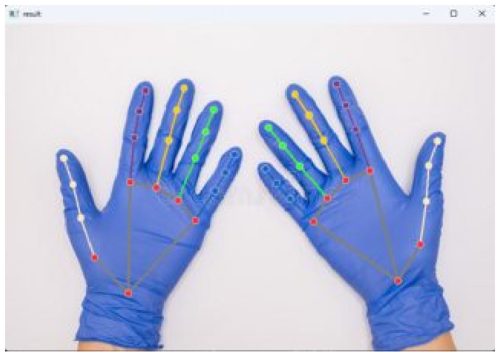


Figure 5: Preprocessing Color Space For Gloved Hands[6]

2.1.4 Perspective Transform

In perspective transformation, we can change the perspective of a given image or video to better understand the required information. In perspective transformation, we need to provide the points on the image in order to gather information by changing the perspective. We also need to provide the points on which we want to display the image. We then take the perspective transform from the two given sets of points and encapsulate it with the original image.

We use cv2.getPerspectiveTransform and cv2.warpPerspective.[2] this is how my code accomplishes this, first by passing a huge piece of A3 paper at a defined angle and using the four corners of the A3 paper as a defined perspective transform datum of four points through which I aim to be able to make the computer's own camera, even with an angular shift, to be able to capture the top view of our hands.

2.1.5 Equations & Simulations

We use perspective transform to change the looking view from the fixed camera view to a top bird-view of our working surface, and in order to draw piano keyboards accordingly in the bird-view warped perspective image window to show corresponding key pressed by hands.

Mathematical Formulation: The perspective transformation can be represented by a 3×3 matrix:

$$\mathbf{M} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & 1 \end{bmatrix}$$

The elements m_{ij} of the matrix are computed such that the matrix transforms the coordinates of four points in the source image to corresponding points in the destination image.

System of Equations: To determine the matrix M, the correspondences between the source points (x_i, y_i) and destination points (x'_i, y'_i) for $i = 1$ to 4 are used to set up a system of linear equations. Each point correspondence must satisfy:

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = \mathbf{M} \cdot \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

2.1.6 Design Alternative

To perform a interactive piano keyboard on our PC screen, we originally used camera view from known laptop lid angle, but this would cause distortion of the keyboard, making it harder for user to detect which key is pressed, now we use an alternative way which is that using perspective transform to perform a top bird view to capture the working surface that our hand is on, making the positioning the piano keys accurate.

2.2 Sensor based Position & Posture Tracking Subsystem

2.2.1 Visualization

This CAD file 6 illustrates the placement of key components within this subsystem. The design showcases the arrangement of 6 Degrees of Freedom (6DOF) position sensors, and Raspberry Pi 4B in relation to the hand.

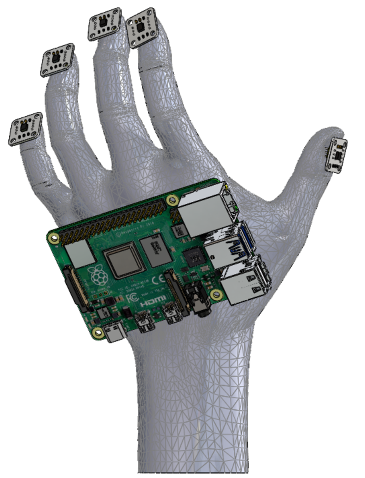


Figure 6: Overall CAD Diagram for Virtual Band[5]

2.2.2 Microcontroller (Raspberry PI 4B)

Input: 5V power supply, 3-axis translational acceleration and 3-axis angular acceleration data, voltage measurement for pressure, video stream from the camera
Output: Pressure sensor and acceleration sensor data

This unit is mainly used to pull the 3-axis translational acceleration and 3-axis angular acceleration data out of JY931 9 degree of freedom sensors, and combine it with the pressure data collected by D0508. These data will be processed within Raspberry PI and sent to the computer through signal synthesis and network. Data transfer and communication will be further discussed within ”Module Communication and Data Transfer” section.

For our project, the Raspberry PI 4B will be powered by a +5V voltage source regulated from two +3.7V 18650 batteries connected in parallel.

2.2.3 6DOF Position and Pose Sensor

Input: Physical translation and orientation of the sensor, 3.3/5V voltage source
Output: Digitalized translational and angular acceleration data

For accurate position and pose sensing, we decide to use JY931. The desirable range scale for translational acceleration would be ±8g, and ±10 rad/s^2, with precision of 0.1g and 0.1 rad/s^2.

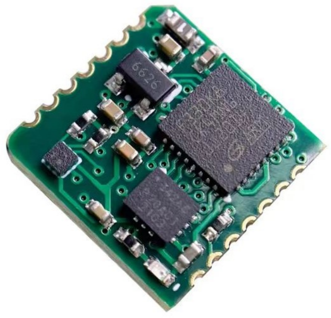


Figure 7: JY931

JY931 iis a compact yet powerful solution for real-time motion sensing and orientation tracking. Equipped with high-precision gyroscopes, accelerometers, and magnetometers, alongside advanced microprocessor technology and Kalman filtering algorithms, it swiftly computes the module's current motion attitude with remarkable accuracy. Employing sophisticated digital filtering techniques, it effectively reduces measurement noise, enhancing precision. With integrated attitude solvers and dynamic Kalman filtering algorithms, it delivers precise attitude measurements even in dynamic environments, boasting an accuracy of 0.2° and exceptional stability surpassing some professional inclinometers. Additionally, its Z-axis heading angle fusion with magnetometer filtering mitigates cumulative errors caused by gyroscope drift, ensuring long-term stable heading angle data output. Featuring built-in voltage regulation circuitry supporting 3.3V/5V operation and compatibility with embedded systems, it offers convenient connectivity options including serial interface support with adjustable baud rates from 4800bps to 921600bps. With a maximum data output rate of 1000Hz and selectable output content and rate ranging from 0.2Hz to 1000Hz, it adapts flexibly to user needs. Crafted with gold-plated stamp hole technology, it seamlessly integrates into user PCB boards.

2.3 Pressure Sensor based Sound Level Determination Subsystem

Input: Physical pressure applied on the finger tip
Output: Pressure sensor and acceleration sensor data

As we have mentioned before, the sound level of the instrument is determined by the data collected by the pressure sensor. The measurable range should be 0g - 5kg with precision of 1g.

This pressure sensor features a slim design with a thickness of 0.3mm, making it ideal for applications requiring space efficiency. It activates with a trigger force of 2kg, signified by a default resistance of less than 1mΩ. With an initial resistance exceeding 10MΩ, it swiftly responds to pressure changes in less than 0.01s. Operating seamlessly across temperatures ranging from -40°C to +85°C, it offers long-lasting performance with a lifespan exceeding one million cycles. This sensor maintains excellent consistency, with resistance variations of +/-3% within individual units and +/-10% across batches under equivalent testing conditions. Additionally, it exhibits minimal hysteresis, with a drift of less than 5% after 24 hours under a static load of 1kg. Furthermore, it demonstrates immunity to electromagnetic interference (EMI) and electrostatic discharge (EDS), ensuring reliable operation in various environmental conditions.

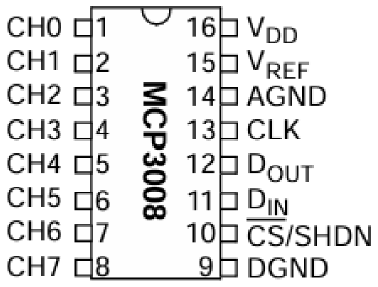


Figure 8: MCP3008 Pinout

The pressure sensor will be integrated into the circuitry with a resistor connected in series. The voltage output from the pressure sensor, reflecting the detected pressure, will be directed to the MCP3008 ADC. This ADC is equipped with eight channels for digitalization, enabling simultaneous conversion of analog signals from multiple sensors. The MCP3008 will facilitate communication between the pressure sensor and the Raspberry Pi microcontroller using the SPI protocol. Through this communication protocol, the Raspberry Pi will receive digitalized pressure data from the ADC, allowing for further processing and analysis within the system. This setup ensures efficient and accurate transmission of pressure information from the sensor to the Raspberry Pi for real-time monitoring and control applications.

2.4 Power Supply

Input: 2 18650 Battery
Output: Pressure sensor and acceleration sensor data

We choose to use 18650 Raspberry Pi dedicated power module as battery source for our project. It is a versatile solution designed to provide stable and reliable power to your Raspberry Pi projects. With its robust features and flexible design, this power module offers an efficient and convenient way to power your Raspberry Pi devices while ensuring optimal performance and extended runtime.

At the heart of this power module are two high-capacity 18650 lithium-ion batteries, each boasting a capacity of 6800mAh. These batteries are configured in parallel, providing a total capacity of 6800mAh for prolonged usage without the need for frequent recharging. This power module is able to deliver a consistent 5V-5.1V/3A output voltage to our Raspberry Pi. This ensures reliable and uninterrupted power supply, essential for maintaining the stability and performance of our projects. This includes the power supply to the Raspberry PI, position and pose sensors, pressure sensors, and a analog-to-digital converter.

Furthermore, the power module boasts three USB-A interfaces capable of delivering a 5V 3A power output each, providing ample power for various peripherals and devices. Additionally, it features a micro-B and a type-C interface, both capable of delivering a 5V 2A power output. This allows us to power additional we have with ease.

2.5 Instrument Virtual Model Subsystem

In this part of the computer vision modeling, we plan to use the positioning and modeling of the camera to assist the positioning of the sensor to the key, so as to obtain more accurate positioning and key differentiation. In addition, we also need to complete the audio processing, including the different audio feedback of the tap and the heavy press, the audio length of the long press and the short press, the pitch of the different keys. This module contributes to the overall design by facilitating accurate key positioning and differentiation. It enhances user experience by providing visual aids for locating keys and ensures precise fingertip positioning. The handshake debugging session improves system reliability by verifying the consistency between pixel movement and actual distances. In this module, we only need to build up the module and set up the basic data for following Below are the interfaces.

Input: Images captured by the camera, Pressure levels from the pressure sensor.
Medium Output: Pixel coordinates of fingertips and key positions, Key positions determined by the Camera
Output: Audio feedback corresponding to key presses.

2.5.1 Camera and Laser Positioning Module

To address the immediate requirement of accurately locating the standard tone, our primary focus will be integrating an auxiliary sensor alongside the existing camera system. By incorporating a small laser module adjacent to the camera, users will be provided with a precise reference point for identifying the key corresponding to the standard tone. This addition enhances user experience by offering a visual cue alongside the auditory feedback.

Simultaneously, leveraging the capabilities of the hand recognition module will facilitate fingertip positioning, further enhancing the accuracy and ease of use of the system. By integrating this functionality into the existing framework, users will have the ability to pinpoint specific notes with precision, thus streamlining the calibration process.

2.5.1.1 Equations & Simulations

We use the accelerator sensors to calculate the distance the hand moves horizontally and vertically. Based on two distances and the piano model, we can easily decide which key is pressing now. We get the equation 1 first. However, If $d_{horizon} > c_{hyperparameter}$, we will need to consider the error caused by distance and calculate the formula below to determine the key. The hyperparameter here is now set to be a fixed number that has more accurate results after several tests manually. Therefore, we get equation **??**

$$\#key = \frac{d_{horizon}}{23.5mm} \tag{1}$$

2.5.1.2 Design Alternative

During the middle of our work, we realized that the accuracy of the accelerator sensors is not quite reliable, which means it can get really high error rate without any fixing mechanism. Therefore, we also work on another method: computer vision. We fixed the angle of the camera, set up the model, and fix each piano key's location. Once we get the pressed signal, we can easily locate the key and play the sound.

2.5.2 Audio Processing Module

In the realm of audio processing, our primary objective revolves around correlating the determined position, derived from sensor data, with the corresponding key audio output. A pivotal challenge lies in effectively integrating pressure levels from the sensor with audio volume levels. Our proposed solution entails a systematic approach:

Pressure-Resistance Curve Calibration:
Initially, we undertake the task of measuring the sensor responses across varying pressure levels, subsequently constructing a pressure-resistance curve. This empirical data serves as the foundation for our calibration process.

Curve Fitting and Inverse Function Utilization:
Employing curve-fitting techniques, we derive a mathematical model that accurately represents the relationship between pressure and resistance. By applying the inverse function to this model, we can efficiently convert voltage readings from the sensor into corresponding pressure values.

Volume Adjustment Based on Pressure:
Leveraging the pressure data obtained, we dynamically adjust the volume of the audio output. This adjustment ensures that the audio volume aligns appropriately with the pressure exerted on the sensor, providing users with an intuitive and responsive auditory experience.

Handling Concurrent Key Presses:
In scenarios where multiple keys are pressed simultaneously, we adopt a provisional approach wherein audio playback for each pressed key occurs concurrently. This interim solution aims to fulfill project requirements efficiently. However, we remain attentive to potential limitations in user experience and explore avenues for improvement.

2.5.2.1 Equations & Simulations

Based on the pressure, we can use the formula below to determine the volume of pressed music note. The force is about 30g to 70g. Based on the Equations below, we can get the numbers we want. $f(voltage)$ is the curve function that we measure the voltage under different Forces and find a curve to fit the data we record. And we get the result as figure 9.

$$Force = f^{-1}(voltage) \tag{2}$$

$$Volume = \min(\max(\frac{Force}{40}, 0.3), 1) \tag{3}$$

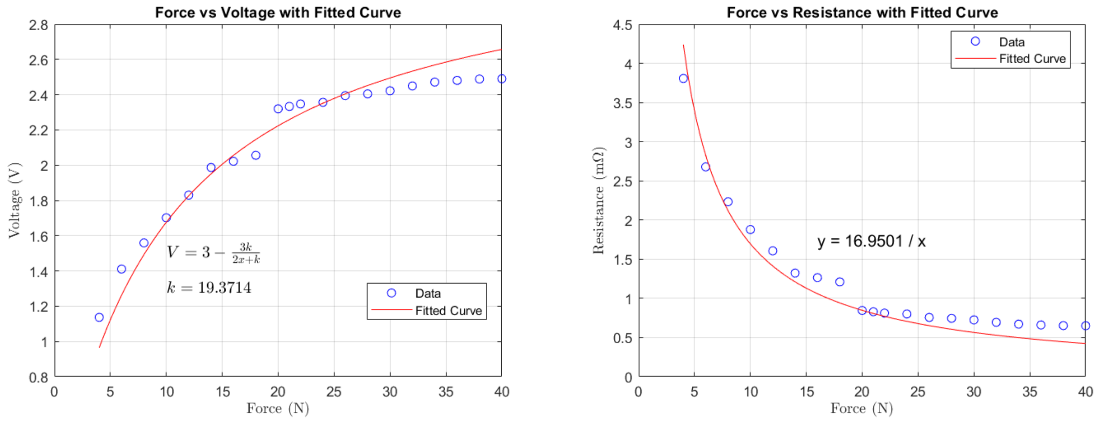


Figure 9: Force vs Voltage curve & Force vs Resistance curve

2.5.2.2 Design Alternative

We initially figured out that the pressure sensor is not able to send back the voltage value but the 0/1 signal whether the voltage is larger than the setting threshold. The following design is that we add an extra voltage monitor to transfer the value back to the computer. In the end, we fix the previous issue and are able to send back the data with pressure sensor, which means we stick to the original design.

2.6 Module Communication and Data Processing

This part mainly addresses how modules communicate with each other and how we process the data.

2.6.1 Sensor Integration

The project is tasked with the integration of 22 sensors distributed across both hands of the musician, comprising 5 pressure sensors and 6 acceleration sensors (JY931) per hand. Each acceleration sensor outputs data in 6 Degrees of Freedom (DOF), with each DOF data being 4 bytes, and operates at a frequency of 256 Hz. The high-frequency operation is pivotal for capturing the nuanced movements of a musician's hands, translating these into precise musical expressions.

2.6.2 Noise Filtering and Data Preprocessing

Given the sensitivity of the sensors and their susceptibility to noise, the subsystem incorporates advanced filtering algorithms executed within the Raspberry Pi 4B modules. These algorithms are designed to eliminate irrelevant or noisy signals, ensuring that only meaningful data is transmitted for further processing. This filtering is particularly crucial for pressure sensors, where even slight pressure variances induced by the glove's material must be accurately accounted for to avoid distortion of the intended musical input. To mitigate the effects of noise and enhance signal clarity, the subsystem employs sophisticated filtering algorithms within the Raspberry Pi 4B modules. These algorithms are tailored to the specific noise profiles and operational frequencies of the pressure and acceleration sensors, leveraging technique. Instead of transmitting continuous sensor data, which includes a significant amount of redundancy and irrelevant information, the system focuses on sending peak data. This approach significantly reduces the amount of data that needs to be transmitted, focusing on those data points that accurately represent the musician's intent.

2.6.3 Equations & Simulations

To reduce the noise, we used the Unscented Kalman Filter. Here are the equations and steps for algorithm implementing.

1. Prediction Step:

   - State Prediction:

$$\mathbf{x}_{k|k-1} = f\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}\right)$$

   - Covariance Prediction:

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T + \mathbf{Q}_k$$

2. Update Step:

   - Compute Sigma Points:

$$\mathcal{X}_k = \left[\mathbf{x}_k, \mathbf{x}_k + \sqrt{(n+\lambda)\mathbf{P}_k}, \mathbf{x}_k - \sqrt{(n+\lambda)\mathbf{P}_k}\right]$$

   - Predict Measurement:

$$\mathbf{z}_k = h\left(\mathcal{X}_k, \mathbf{v}_k\right)$$

   - Compute Covariance of Innovation:

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k$$

   - Compute Cross-Covariance Matrix:

$$\mathbf{C}_k = \frac{1}{2\lambda + n} \sum_{i=0}^{2n} \mathbf{W}_i^c \left(\mathcal{X}_i - \mathbf{x}_{k|k-1}\right) \left(\mathbf{Z}_i - \mathbf{z}_k\right)^T$$

- Compute Kalman Gain:
$$\mathbf{K}_k = \mathbf{C}_k \mathbf{S}_k^{-1}$$
- Update State Estimate:
$$\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + \mathbf{K}_k \left(\mathbf{z}_k - \mathbf{z}_{k|k-1} \right)$$
- Update Covariance Estimate:
$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T$$

2.6.4 Design Alternatives

In the beginning stages of designing a sensor-based motion and posture tracking subsystem for the Virtual Band Gloves project, we selected the MPU6050 sensor. However, during the testing phase after writing the code, we encountered several unavoidable sensor design issues that would be serious and largely affect the accuracy and performance of the whole system.

First, the MPU6050 lacks a magnetic sensor, and therefore cannot obtain accurate angle data directly from the Kalman filter. This limitation leads to inaccurate compensation of the gravitational acceleration, which greatly amplifies the process error generated during the integration process. As a result, the position data obtained is completely unreliable, resulting in the entire system performing significantly differently than expected.

Second, we found that the sampling rate of the MPU6050 was lower than expected. Although the standard specification is 256 Hz, after our saturation testing, the actual sampling rate of the MPU6050 from the manufacturer we purchased was approximately 150 Hz. This discrepancy resulted in missed or delayed acceleration detections, especially during fast motion. As a result, there were errors in the position integration process that affected the reliability of the system.

Third, we found that the zero offsets of the acceleration and angular velocity readings from the MPU6050 were too large, and the acceleration and angular velocity zero offsets were not scaled consistently for the different sensors, resulting in the lack of a uniform low-pass filtering to reduce these errors, and the necessity to design each finger sensor individually, which would have drastically increased our workload. These offset errors accumulated over time and severely impacted the accuracy of the integrated position and angular attitude data. To address these design issues and inconsistencies, we conducted a thorough evaluation of alternative sensor solutions. The JY931 sensor was selected for its advantages over the MPU6050, which it compensates for with a real-world sampling rate of up to 256 Hz, higher accuracy, and the ability to acquire precise angular data.

The adoption of the JY931 sensor has significantly improved the accuracy and performance of our sensor-based motion and posture tracking system. the higher sampling rate and increased accuracy of the JY931 sensor helps to more accurately detect and integrate motion and posture data, allowing our algorithms to more accurately calculate the distance traveled by the fingers, and thus to more accurately determine the notes played by the player!

However, despite these significant improvements, the transition to the JY931 sensor posed a new challenge. We needed to rework our existing SDK to allow the Raspberry Pi to automatically read data from the sensors' registers, and to do this, we carefully reviewed the data and redesigned a telltale SDK suitable for accomplishing our goal of quickly reading the desired data from the registers. In our design, multiple sensors are connected to a single input line through a multiplexer that sequentially selects the output of each sensor. This multiplexing inherently reduces the effective sample rate of each sensor. Although the JY931 can theoretically achieve a sampling rate of 256Hz, in this case the actual sampling rate is reduced.

Reductions in the effective sampling rate can lead to degradation of motion detection resolution, especially during fast motion when high-frequency data sampling is critical. To mitigate these effects, we use a number of different solutions, including optimizing multiplexer switching algorithms and reconstructing missing data points using advanced filtering techniques. We had also devised the use of machine learning algorithms to perform data recovery on data points that might indeed be missing, but this approach would have resulted in a rise in latency, and based on this consideration, we decided to abandon this approach.

In addition, we continuously monitor sensor performance and the impact of multiplexing on data accuracy. Through iterative testing and calibration, we aim to fine-tune the system to minimize any adverse effects of reduced sample rates.

2.6.5 Communication Protocol Design

This subsystem leverages a dual-protocol strategy to optimize data transfer:

HTTP Protocol: Ensures data integrity and is used for intra-network communication. This protocol is particularly suited for environments where network consistency can be maintained, providing a reliable channel for data transfer. gRPC: Facilitates faster data transfer rates, essential for reducing latency in musical performance. By employing gRPC, the subsystem ensures that data packets are efficiently encoded and transmitted across network boundaries, enabling real-time interaction between the sensors, Raspberry Pi modules, and the central laptop. Both protocols operate through four dedicated ports on each device, enhancing data security and ensuring that the data streams remain segregated and protected.

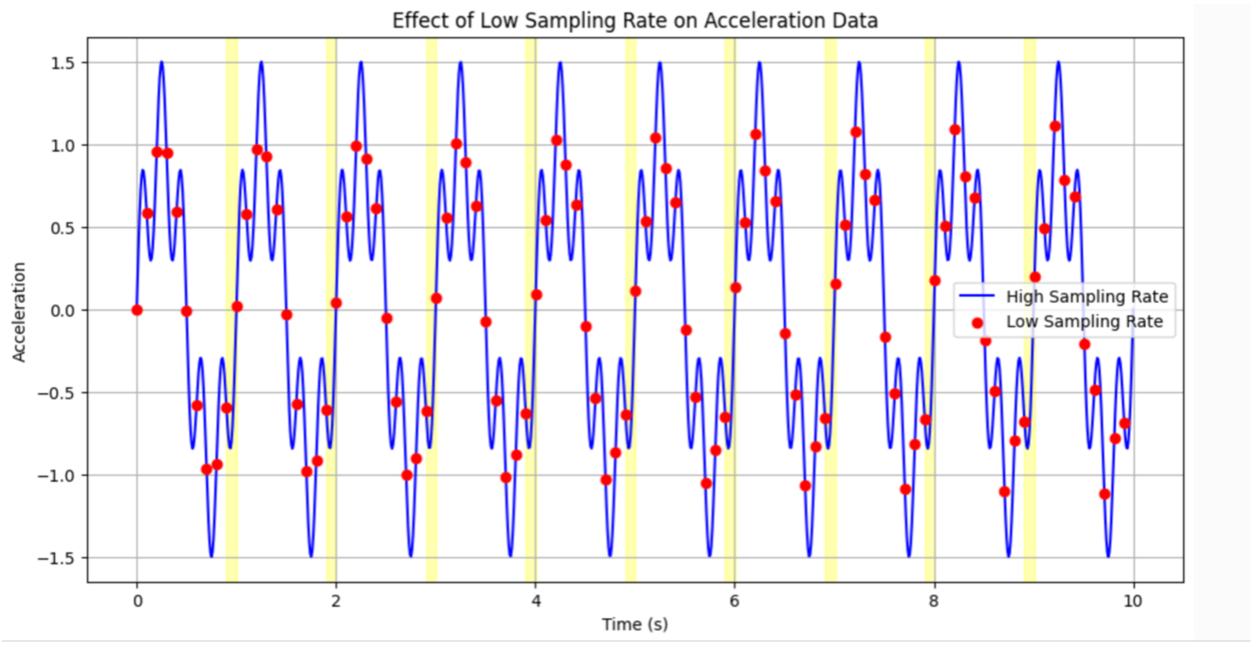


Figure 10: Illustration of Sampling Rate Comparison

2.6.6 Synchronization and Consistency

Addressing the Byzantine Fault Tolerance (BFT) problem, the subsystem integrates a set_time function across all components to mitigate time drift and ensure synchronized operations. This function is invoked every minute, aligning the internal clocks of the sensors, Raspberry Pis, and the central processing unit. By embedding timestamps within the transmitted data packets, the system can correlate data streams with precision, ensuring that simultaneous events across the system are processed coherently.

2.6.7 Sensor and Camera Data Processing

Processing data from multiple sensors and a camera to accurately capture and interpret a musician's movements and inputs involves a sophisticated integration of sensor data, algorithmic decision-making, and error correction methodologies. At the core of this process is the objective to meticulously translate physical actions into precise digital outputs that reflect the intended musical expressions on a virtual piano interface.

The operational logic begins with data acquisition from acceleration sensors attached to each finger and hand of the player. These sensors are critical for determining the relative movement of the fingers from a predefined central zero point. By analyzing this movement data, the system can infer which keys on the virtual piano are being pressed or intended to be pressed by the musician. This inference is based on a pre-established piano model that maps specific spatial coordinates to corresponding piano keys, thus allowing for the accurate translation of physical movements into specific musical notes.

Concurrently, visual data captured by the camera serve a dual purpose: enhancing the accuracy of movement detection and providing a secondary verification mechanism to ensure the reliability of the sensor data. The camera monitors the positioning of the fingers in real-time, offering an external perspective that is particularly valuable in cases where the acceleration sensors might encounter errors or inaccuracies—especially when the musician's hands move rapidly or far from the central zero point.

To reconcile any discrepancies between the data derived from the acceleration sensors and the visual confirmation from the camera, the system employs a weighted harmonic average algorithm. This sophisticated algorithm takes into account both sets of data, prioritizing accuracy by adjusting for potential sensor errors or visual misalignments. By calculating the weighted harmonic average of the two positions, the system effectively harmonizes the sensor and camera data, ensuring that any deviation or error is corrected and that the determined finger position is as accurate as possible.

Once the system has confidently identified the finger positions and detected a corresponding signal from the pressure sensors—indicating not just contact with a virtual key but also the intensity of the press—it proceeds to generate the appropriate musical note. This note is then sustained for the duration of the signal received from the pressure sensors, mirroring the natural behavior of a piano where the sound persists for as long as the key is pressed.

3 Cost and Schedule

3.1 Cost Analysis

- Labor:
  - For each partner in the project, \$20/hour as the standard salary, everyone works at least 6 hours/week
  - Use the formula $(\$20/\text{hour}) \times 2.5 \times (9$ weeks $\times\ 6$ hours/week$)$ = TOTAL
  - Add details about the total labor for all partners here.
- Parts:
  - table listing all parts (description, manufacturer, part number, quantity and cost)
    Every component below is for one hand. If we have extra time after succeeding in finishing the right hand, we will buy another set for the left hand. Every component will double.

Num	Part (for one hand)	Item	Cost
1	GY-521 MPU6050 module 3D Angle sensor 6DOF three-axis accelerometer six-axis gyroscope	1	¥161.24
2	Weite intelligent serial port nine-axis high-precision accelerometer gyroscope module attitude Angle sensor JY931	5	¥142
3	Raspberry PI 4B 8g	1	¥549
4	Shida anti-cut gloves	1	¥35.73
5	PCA9548 PCA9548A 1-to-8 I2C 8-way IIC multiway expansion board Module development board	1	¥5.28
6	Raspberry PI 4B/3B+ 18650 Battery expansion board UPS uninterruptible power supply 5V3A power supply lithium battery module	1	¥107
7	Thin film pressure sensor RP-C FSR402 406 Flexible resistive force and pressure sensitive plantar-robot	5	¥87.5
8	Dupont thread	3	¥29.94
9	Breadboard 400 Tie-Points	3	¥9.72

Table 2: Cost Analysis

- Sum of costs into a grand total:
  - total grand is $1000

3.2 Schedule

Date	Xuan Tang	Hengyu Liu	Han Chen	Zhanpeng Li
3/11	Set up environment for yolo v9 recognition Implemented yolo v9 for capturing hand position and motion information	Started initialization of the raspberry PI 4B and try to set up the development environment.	For capturing the hand position, assuming we have identified the finger position, how to calculate the distance is the problem I try to solve.	Completed purchase of the sensors including 6DOF sensors with different precision (GY-521 and JY931) and microcontrollers (Raspberry PI 4B).

3/18	Complete code analysis and schedule, discuss detailed implementation for finger position detection.	Complete high-level requirement in design document, set up VNC viewer for developing purpose on raspberry pi	Try to get formula to calculate the distance and determine piano key. Finish tolerance analysis and formula calculation in design document.	Complete Visual Aid and Physical Design for the product.
3/25	Collect and annotate datasets for palm and gloved hands detection, evaluate performance of MediaPipe model on gloved hands	Complete data transfer protocol implementation bewteen sensors and rasberry pi	Test code that displays and realizes the function of piano key. Write the protocol from existing code to for later signal processing.	Test and validate force vs pressure sensor curve.
4/1	Testing the performance of color space adjustment on gloved hands	Try to connect the position and pose sensor with the raspberry pi and be able to read the data, try to connect the pressure sensor with the raspberry pi and be able to read the data	Find some other instruments database. Help the team connect all sensor and get the output signals. Discuss the method of treating signals.	Connect pressure sensors to 8-channel I2C multiplexer, Design a connection part for I2C multiplexer if necessary. Assess the need for a connection part based on the setup and requirements
4/8	Evaluate the color space adjustment method performs bad in gloved hands detection, retrain the model using model maker	Complete latency analysis,if the latency go over 70ms, adjust the protocol and data processing process, aiming to decrease latency of concurrency actions.	Connect signal processing parts with notes playing. (being able to play notes when pressing)	Design a connection part compatible with the multiplexer and other components. Prototype the design and test its functionality. .

4/15	Evaluate the re-trained model for the gloved hands detection, test on our virtual band gloves	Integrate data transfered from all fingers, and adjust the transfer bandwidth and protocol	Finish locating module. Finish distance calculation and key determination.	Prepare the position and pose sensors for testing. Develop a testing procedure to assess the functionality and accuracy of the sensors.
4/22	Debug and optimize the landmark detecting model for real-time detection	Integrate camera and sensors for data processing, test audio protocol and noise cancellation method. Address the set_time function to solve the consensus problem	Finish location adjustment and calibration parts.	Conduct tests to evaluate the performance of the sensors under different conditions. Iterate on the design if necessary to ensure compatibility and functionality.
4/29	Complete Vision task for our specific gloved-hand landmark detection	Debug and Analyze test results and identify issues or areas for improvement	Test play accurate keynotes.	Debug and Analyze test results and identify any issues or areas for improvement
5/6	Test and demo, self evaluation and beautify design	Test and demo, self evaluation and beautify design	Test and demo, self evaluation and beautify design	Test and demo, self evaluation and beautify design

Table 3: Schedule of Tasks

4 Requirements and Verification

4.1 Subsystems

4.1.1 Results of Computer Vision based Position & Posture Detection Subsystem

Following the R&V table A, we obtain real-time tracking for blue gloved hands, and all of the hand landmarks are shown correctly, each landmarks coordinates are precise enough to indicate piano key pressed. But under low-light environments, and similar color space backgrounds, the hand landmarking process is not precise.

After implementing the MediaPipe model with color space transformation method, the modified model was able to identify and track blue gloved hand landmarks with an accuracy under many scenarios. This is a significant improvement over the initial trials, when we are hardly able to mark the landmarks of the hands after wearing gloves.

We provide a perspective transformation from the camera image captured from the integrated camera, this gives us the top bird-view of the working surface that our hands are on. This gives a clear view and also reduce the difficulty to visualize the virtual piano key and further implemented features.

4.1.2 Sensor based Position & Posture Tracking Subsystem

Following the R&V table A, regarding translational acceleration measurement, our sensor successfully meets the requirement of ±8g range with a precision of 0.1g. However, the detected displacement accuracy falls slightly short of the specified 20% tolerance. This discrepancy suggests that additional calibration or refinement of our measurement algorithms may be needed to enhance accuracy in detecting translational displacement.

In terms of angular velocity measurement, our sensor meets the requirement of ±10 rad/s range with a precision of 0.1 rad/s. Furthermore, the detected angle accuracy surpasses expectations, achieving an impressive 0.1% accuracy. This indicates that our sensor performs exceptionally well in accurately measuring angular velocity and corresponding angles.

Overall, while our sensor demonstrates strong performance in angular velocity measurement and angle accuracy, there is room for improvement in translational displacement accuracy. Future efforts will focus on refining calibration methods and optimizing algorithms to enhance the sensor's performance in detecting translational displacement within the specified tolerance.

4.1.3 Result of Pressure Sensor based Sound Level Determination Subsystem

Following the R&V table A, regarding pressure measurement within the range of 0g to 5kg with a precision of 1g, our tests indicate that the sensor accurately measures pressure values within this range. By applying known pressure values ranging from 0g to 3kg to the sensor, we have verified the accuracy of the measured pressure values against the expected values. The sensor consistently provides precise measurements, meeting the specified precision requirement of 1g.

Furthermore, in terms of sensor activation upon application of a trigger force of 2kg, our tests confirm that the sensor reliably activates when the trigger force is applied. This activation is indicated by a default resistance of less than 1mΩ, as per the requirement. By applying incremental pressure changes within the measurable range, we have verified the sensor's sensitivity and threshold, observing corresponding output changes as expected.

Overall, the results of our testing demonstrate that the pressure sensor meets all specified requirements with precision and reliability. Its ability to accurately measure pressure values within the specified range, activate at the designated trigger force, and exhibit appropriate sensitivity and threshold make it suitable for its intended application.

4.1.4 Result of Power Supply

Following the R&V table A, based on the specified requirements for the power supply subsystem, our results analysis confirms the following:

1. **Stable Output Voltage:** The power module successfully provides a stable output voltage within the range of 5V to 5.1V. This was verified by using a digital multimeter (DMM) to measure the output voltage of the power module, which fell within the specified range.
2. **Current Delivery Capability:** The voltage output of the power module is capable of delivering a current of up to 3A to the connected devices. To verify this capability, a load equivalent to 3A was applied to the power module. The voltage stability under maximum load conditions was observed and confirmed to be within acceptable limits.

Overall, the power supply subsystem meets both specified requirements, providing a stable output voltage within the required range and demonstrating the capability to deliver the necessary current to connected devices. These results indicate that the power supply subsystem is reliable and suitable for powering the associated electronic components and devices in the system.

4.1.5 Result of Instrument Virtual Model Subsystem

Following the R&V table A, for the **first** one, the piano model is set up precisely and Computer Vision part is able to track all fingertips as Section 2.1.1. For the **second** one, the key determination function is able to find out the right piano key based on $(x_{distance}, y_{distance})$ that the horizontal and vertical movement distance. Also, the program is able to play the sound of the exact piano key. For the **third** part, we move the calibration part to Computer Vision subsystem to set the **C4** note position for the user. For the **fourth** part, we use two methods to determine the key - distance and vision. In the vision part, the pixel location is accurate and this assists the previous method to increase the accuracy. The result here is pretty good. The **last** one, from our project, we got the curve to determine the volume and easily realized from $pygame$ module we used for this section. Overall, we finished and realized these functionalities in high quality.

4.1.6 Result Module Communication and Data Processing

Following the R&V table A, this part contains several key components: Respberry PI 4b, Sensors, camera and ONE computer.

Our communication protocol was built through UDP, and it achieves low latency communication among all important 3 features, sensors, computer and Raspberry Pi. Since the sampling rate of our sensors are not that high(250 Hz), our latency was roughly 2 - 3 ms, which is totally acceptable for playing instruments. Also, considering about the data processing, since we involved advanced filtering algorithm like UKF, which needs to compute heavy load matrix multiplication and inversion, for a given sample point, the time needs to compute the final result of our position is within 800μs, on MacBook Pro with M1 Max chip, and running as Golang program. And the error of position estimation, given the nature of low sampling rate and different zero diffusion, the best precision our UKF achieve is roughly 85%, since we only have one sensor for each finger, and no further prediction for the movement, so the system is completely nonlinear.

5 Conclusion

Our senior design project, the Virtual Band Gloves, offered a novel solution to the limitations posed by traditional musical instruments. By integrating technologies in audio processing, sensor dynamics, and computer vision, we have developed a versatile, portable, and intuitive musical instrument interface.

Throughout the development process, our team successfully designed and implemented several subsystems that work cohesively to translate the movements and pressures of a musician's hands into precise musical outputs. The implementation of the MediaPipe Hand Landmarker with color space transformation for motion detection, along with the incorporation of pressure sensors and advanced data processing techniques, has allowed the gloves to emulate various musical instruments effectively. This capability was verified through rigorous testing, which confirmed the accuracy of the sensors and the responsiveness of the system to the user's gestures.

However, the project was not without its challenges. We encountered difficulties in sensor accuracy and data integration, which required us to explore alternative solutions and refine our approach continually. These challenges were significant learning opportunities that pushed us to innovate and think critically about solving complex engineering problems.

Looking forward, the potential for further development and commercialization of the Virtual Band Gloves is vast. Future work could focus on enhancing the user interface, expanding the range of instruments and sounds available, and improving the portability and durability of the system. Additionally, exploring machine learning algorithms for gesture prediction could refine the responsiveness and user-friendliness of the gloves.

In conclusion, the Virtual Band Gloves project not only meets the design criteria set forth in our initial objectives but also provides a open platform for future innovation in musical performance technology. Our team is proud of what we have achieved and optimistic about the potential impacts of our work on the music and technology communities.

A Requirements & Verification

Here are the Requirement & Verification (R&V) tables of all different subsystems and subsections. Appendix A is designed to assist understanding section 4.

Requirement	**Verification**
Ensure accurate hand landmark positioning	Check whether the landmarker are shown correctly
Since our hands would be gloved and features with multiple equipment, normal mediapipe model could not handle landmarker tasks	make sure gloved hands can be fit in our retrained model
We used perspective transformation to provide a top bird-view for camera capturing	make sure the camera calibration is well implemented to obtain real positioning.

Table 4: R&V Table for this computer vision subsystem

Requirement	**Verification**
The sensor must accurately measure translational acceleration within the range of ±8g with a precision of 0.1g. Consequently, the detected displacement accuracy should be within 20%	Move the sensor along one axis with known displacement and verify the accuracy of the measured values against the expected values.
It should accurately measure angular velocity within the range of ±10 rad/s with a precision of 0.1 rad/s.And the detected angle precision should be within 5%	Rotate the sensor with respect to a fixed axis with known angle and verify the accuracy of the measured values against the expected values.
It must offer programmable interrupts for various functions such as posture recognition, shake detection, image scaling, scrolling, high-G sensing, motion detection, tap sensing, and shake sensing.	Confirm the functionality of programmable interrupts by triggering various interrupt conditions and observing the sensor's response.

Table 5: R&V Table for Position & Posture Sensor

Requirement	**Verification**
The pressure sensor must accurately measure pressures within the range of 0g to 5kg. The precision of pressure measurement should be 1g.	Apply known pressure values within the range of 0g to 5kg to the sensor and verify the accuracy of the measured pressure values against the expected values.
The sensor should activate when a trigger force of 2kg is applied, indicated by a default resistance of less than 1mΩ.	Verify the sensor's sensitivity and threshold by applying incremental pressure changes within the measurable range and observing the corresponding output changes.

Table 6: R&V Table for Pressure Sensor

Requirement	**Verification**
Ensure accurate positioning and modeling of the camera for precise sensor alignment	Check that the tracking points are at the right places.
Provide audio feedback corresponding to different key presses	Verify audio feedback correctness against key presses.
Integrate an auxiliary laser module for precise key localization & Implement hand recognition for enhanced fingertip positioning.	Be able to locate the center point and receive the feedback of calibration.
Conduct handshake debugging session to ensure accurate finger movement tracking.	The program is able to record and compare the distance with the number of pixels on screen.
Utilize mathematical models for pressure-to-volume conversion & Implement concurrent audio playback for multiple key presses.	Validate pressure-to-volume conversion accuracy through controlled pressure exertion and audio volume comparison.

Table 7: R&V Table for Instrument Virtual Model Subsystem

Requirement	**Verification**
The power module must provide a stable output voltage within the range of 5V-5.1V.	Use a digital multimeter (DMM) to measure the output voltage of the power module.
The voltage output should be capable of delivering a current of up to 3A to the connected devices.	Apply a load equivalent to 3A to verify the voltage stability under maximum load conditions.

Table 8: R&V Table for Power Supply

Requirement	**Verification**
Latency must be less than 7ms for real-time performance	A. Perform end-to-end system latency tests using timestamped sensor data. B. Verify latency under maximum load using a network analyzer to eliminate latency by re-write submodules that may cost large latency.
Data transfer bandwidth must accommodate the sensor data rate	A. Calculate minimum required bandwidth: $11 \times 2 \times 6 \times 4 \times 1000 = 528000$ bytes/sec. B. Test with network monitoring tools to ensure bandwidth exceeds this rate. Also make sure PI can afford this large data load in each pin.
Sensor data must be accurate to within a specified threshold, for pressure sensor, can detect 0.02N to 15N, and for acceleration sensors, accuracy must be in 6 digits	A. Calibrate sensors against a known standard. B. Test sensor output by simulating controlled movements and compare against expected data.
Noise filtering must effectively isolate and remove unwanted signals	A. Introduce known noise patterns into sensor data, such as steady noisy caused from gloves. B. Verify filtering algorithms remove them and analyze S2N ratio.
Synchronization across all system components must be maintained within a small margin of error	A. Implement $'set\_time'$ function. B. Verify synchronization with long-duration testing to ensure minimal time drift.

Table 9: R&V Table for Communication and Data Processing Module

References

[1] *Block Diagram for Virtual Band*. Figure 2. Page 3.

[2] GeeksforGeeks. *Perspective Transformation – Python OpenCV*. https://www.geeksforgeeks.org/perspective-transformation-python-opencv/. Accessed: 2024-05-27. 2023.

[3] *MediaPipe Hand Landmarker Demo*. Figure 10. Page 8.

[4] *MediaPipe Hand Landmarks*. Figure 9. https://developers.google.com/mediapipe/solutions/hands.

[5] *Overall CAD Diagram for Virtual Band*. Figure 3. Page 4.

[6] *Preprocessing Color Space For Gloved Hands*. Figure 11. https://stackoverflow.com/questions/76325975/how-to-make-the-hand-detection-of-mediap.

[7] *Product in Context*. Figure 1. Page 2.