Dexterity: A Low-cost, Remotely Controlled, Humanoid Robotic Arm

An Undergraduate Capstone Project

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Jacob Hall

Spring 2025

Dexterity

Robotic Armature for Hazardous Materials Manipulation Operated via Haptic Interface Glove

Alex Schaefer, Bhargav Moosani, Jackson Lamb, Jacob Hall, Max Titov

Statement of Work

Max Titov

I did the high level design at the start of the semester to lay out the groundwork for the project. To do this I made several sketches and diagrams to explain the general PCB layout and the electrical systems of both the robotic arm and the control glove. I chose the ESP32 microcontroller we are using and did initial testing of it and figured out how to use the debugger of PlatformIO with our ESP32.

Then I moved on to the mechanical assembly of the robotic arm based on the Dexhand open source project. I 3D printed every part and post processed them by sanding the support scarring off. This part took much longer than I anticipated, even though there was very good documentation of the build process. As I assembled the hand, I wrote control software for the finger movement based and controlled it with the prototype glove that I designed over the summer. More specifically, I wrote the HallEffectSensors, FingerTracking, FingerControl, and EmaxServoControl libraries.

Also, I designed our second PCB for the forearm after discovering that the ribbon cable was flipped in the original design because Jackson was out for a medical reason.

After finishing the mechanical assembly, I switched to mechanical design of the control glove. I went through 4 iterations of the design, 3D printing at each step to test how it all worked. Then I assembled the mechanical of it and Alex and I assembled the electrical of the glove. Finally, I designed the robotic arm stand and mounting system.

Alex Schaefer

Throughout the course of the semester, I spent the majority of it working on tasks related to the embedded code. I started off by trying to learn the costs and benefits of each coding environment for the ESP32, before coming to the conclusion with the group to use PlatformIO. Next, I worked on creating libraries for the wireless ESPNOW communication, testing its reliability, and determining how to improve it. Once I had libraries set up for our specific ESPNOW usage, I started work on FeeTech serial servo control. I determined which libraries worked best for our serial servo control and I utilized the FeeTech FD software for servo programming and debugging (FeeTech servos allow users to set identification and startup internal variable values). Once I had a good understanding of the serial servos, I moved on to building the high level embedded code.

When starting the process of building the main embedded code project, I first learned how to implement multithreading on the ESP32. I next worked with Jackson on integrating timer-based ISRs within multiple cores. Once I had a basic outline for both, I began structuring the architecture of the full project. I focused on creating several project files that are loosely coupled and have a single, specific functionality. I then worked on the data flow within the project. By the end of the high level development, my goal was to have each team member be able to plug the functions that they had been working on into just two functions in one file: one function that runs at setup and one function that runs at a 50 Hz rate (managed by the ISR). My goal was for each person to integrate into a separate file such that issues would be easy to track and git conflicts would be minimized. For the most part, I succeeded in creating such an interface. Finally, I spent the rest of my time helping my team members integrate their code into the main PlatformIO project and test the full pipeline functionality. This includes the creation of the wrist control library by combining Jacob's knowledge of the IMUs and my knowledge of the serial servos and software structure. I also spend some time helping Max assemble the control glove.

Jackson Lamb

My primary responsibility during the semester was designing and testing the two PCBs used on the control glove. This began by laying out the circuit schematics in KiCad according to Max's rough sketches. Then, I forward annotated my schematics into the KiCad PCB and started laying out components and routing connections on our PCBs. This was by no means a linear process. There were several times I was required to reconfigure the schematics and adjust the PCB layout and routing accordingly. For example, we realized not all of the pins on our microcontroller could be freely used as we initially expected after I had already done a draft layout and routing. As part of the PCB design process, I needed to find symbols and footprints for the components we were going to use, so I ended up selecting some of our parts. These included the I2C multiplexer used to drive the haptic buzzers and the linear voltage regulator. My final involvement with the PCBs was performing a connectivity ("beep") test on them.

As a secondary responsibility, I contributed to our embedded codebase by writing two libraries and contributing to two others. The two libraries I wrote, GloveControlPanel and HandControlPanel, provide simple APIs for interacting with the buttons, switches, and LEDs connected to each microcontroller operating our control glove and robotic hand. I reformatted Max's extant code for reading data from the control glove's Hall effect sensors into a library. Finally, I assisted Bhargav with the library for driving our haptic buzzers.

Bhargav Moosani

My primary responsibility for this Capstone project was to design and test the haptic feedback system, which provides tactile feedback between the robotic armature and the control glove. Initially, the plan was to implement an electromagnetic brake system to provide variable resistance to the control glove, gradually restricting its movement as the robotic arm interacted with the environment. After a couple weeks of testing, we realized that this was not feasible (or at least out of scope to implement), as the finger tracking system of our control glove involves using Hall-effect sensors to track finger movement. The presence of an external electromagnetic field would interfere with these measurements, and thus I had to seek an alternative method of implementing haptic feedback in our system.

The alternative haptic feedback solution was to use linear resonant actuators (LRA) on the control glove to provide tactile feedback. The LRAs were integrated wirelessly with force-sensing resistors (FSR) on the robotic arm via ESP32 microcontrollers. This began with developing the LRAs to simulate variable pressure through vibrations. Once I found a progression of haptic settings that were realistic in regards to sense of touch, I integrated one FSR with an LRA to ensure that the haptic settings were being driven properly by the FSR. These tests led to the realization that the FSR's upper pressure limit of 2 kgs was not sufficient, so after doing some research I found different FSRs that had 5 kg capacity which would be plenty for our implementation. I then worked with Alex to integrate the haptic system wirelessly using two ESP32s, which involved integrating the ESPNOW library with the Haptic and ForceSensor libraries that I wrote along with Jackson. Lastly, I had to integrate the five LRAs with the I2C multiplexer, which would complete development of the haptic feedback system.

Jacob Hall

During the initial project conception, it was clear that the hall-effect sensor mechanism for finger tracking would work well. However, a different method of capturing the wrist orientation was desired. I proposed utilizing two IMUs, one on the hand, and one on the forearm of the glove to calculate the orientation of the forearm in space and the wrist orientation relative to the forearm. I searched for viable IMU chips and found the ones used in our design early in the semester. After receiving the IMU chips, I found that existing code libraries for the chips only supported one IMU on an I2C bus at a time. Available GPIO pinouts on the microcontroller were sparse with the number of objectives for the glove (such as resistive feedback) competing for pins at the time, so using two I2C buses was not an option. I found an example of bitbanging to communicate with the IMUs on the same I2C bus, so I implemented a code library for communicating with the IMUs using this method. I also researched quaternion mathematics and conversions to euler angles and implemented the functions necessary to convert the raw quaternion outputs of both IMUs into a single euler angle sequence for the wrist orientation. In addition, I researched how to calibrate the IMUs and wrote a calibration sequence run during startup of the control glove. On the robotic arm side, I reviewed existing DexHand robotic arm project code and adapted the wrist orientation-to-servo command code to work with our project. I also assisted with soldering the PCB board.

Table of Contents

Table of Figures

Abstract

This project aims to develop a haptic finger-tracking glove that controls a robotic hand, enabling safe manipulation of hazardous materials while maintaining human-like dexterity. The system consists of a control glove that tracks the user’s finger movement, coupled with inertial measurement units (IMUs) to monitor hand and forearm orientation. This motion-tracking system drives a robotic armature, which is based on the open-source Dexhand project [1]. We will integrate pressure sensors on each robotic fingertip that activate linear resonant actuators (LRA) on the corresponding fingertip of the control glove. The LRAs will provide vibrational tactile feedback proportional to the amount of pressure applied to simulate the sensation of object manipulation. This haptic feedback, combined with the precise motion tracking, allows for intuitive and accurate control of the robotic hand in hazardous environments.

Background

The increasing need for safe and cost-effective solutions in laboratories handling hazardous materials has driven our team to develop a haptic finger and forearm-tracking glove system that controls a robotic hand. This system is designed to enable researchers to interact with toxic substances while maintaining human-like dexterity and without the need for expensive fume hoods or overly complex robotic setups. The core innovation of our project lies in creating a more affordable and user-friendly alternative compared to existing commercial solutions, making this technology accessible to a wider audience of researchers and laboratories.

Prior Work

Commercial products such as HaptX [2], SenseGlove [3], and Weart [4] have set the standard in the haptic feedback and remote manipulation market, offering highly immersive and precise systems. These technologies allow users to interact with virtual environments or robotic systems while experiencing realistic tactile feedback. However, these products are often prohibitively expensive and primarily serve industrial applications, limiting their accessibility to smaller research labs and educational institutions.

In the open-source space, projects like LucasVR [5] and Nepyope's VR Glove [6] have demonstrated innovative solutions for hand-tracking and haptic feedback. These projects offer promising foundations, especially in terms of making the technology more accessible. However, these products are not tailored to handle hazardous materials in real-world application, as their primary use cases lie within virtual reality environments.

Our project differentiates itself in two key ways: **cost** and **target audience**. By leveraging open-source designs and affordable components, we aim to drastically reduce the overall cost of the system, making it viable for smaller research institutions and universities. Our system also targets laboratories that require safe manipulation of hazardous materials. Unlike existing virtual reality or industrial solutions, our glove allows researchers to remotely control a robotic hand to handle toxic substances with haptic feedback, improving both safety and ease of use without the need for costly equipment or infrastructure.

Coursework Preparation

Our coursework at the University of Virginia has provided a strong foundation for this project. Several key courses have directly prepared us for the technical challenges involved:

- **Intro to Embedded Computing Systems** has given us a solid understanding of microcontrollers and real-time systems, both critical for the kinetic of the glove and robotic arm as well as driving the haptic feedback system to simulate tactile sensation.
- **Intro to Control Systems** and **Signals & Systems** laid the groundwork for developing the precise control algorithms necessary for converting user hand movements into corresponding robotic arm movements.
- **Electronics** provided us with the skills to design and integrate the necessary hardware components, such as sensors and actuators, into a compact and efficient system.
- **Computer-Aided Design** has allowed us to design and/or modify existing models of both the glove and robotic arm components in a way that is both ergonomic and easy to manufacture, ensuring that the final product is as user-friendly as it is functional.

By applying the principles and techniques we have learned in these courses, we are well-equipped to develop a system that not only tracks hand movements accurately but also provides real-time, tactile feedback to the user. This combination of coursework and innovative design will allow us to bring a cost-effective, high-functioning tool to laboratories where safety and precision are paramount.

Project Description

Performance Objectives

The broad performance objective of this project is to design and implement a control glove for the DexHand, open-source robotic arm. Our main objective is to keep the full system affordable and accessible without excluding functionality. Our system needs to be low latency while providing sufficient feedback to give the user an intuitive sense of the forces encountered by the robotic arm. Thus, the latency between forces encountered and buzzing on the control glove will be within two 50 Hz cycles, or 0.04 seconds. Similarly, the position data will be transferred to the robotic arm within 0.04 seconds. However, due to the inherent delay of servo motor speed, the robotic arm cannot move to the received position within 0.04 seconds. Instead, the arm must be moving in the proper direction within 0.04 seconds. By implementing these features, labs will be able to utilize low-cost robotics for general purpose handling of hazardous materials without sacrificing functionality.

How it Works and Technical Details

Overall Description

As can be seen in Figure 1 below, the Dexterity system consists of a user-worn glove consisting of a suite of sensors with a haptic interface and a robotic arm controlled by the glove. At a high level, the hall-effect sensor signals from each of the fingers and the forearm are processed into movement commands sent to the robotic arm. Upon receiving these commands, the robotic arm fingers and forearm are manipulated to match that of the glove as close as possible. The robotic arm has a suite of touch sensors, whose data is processed into haptic commands sent back to the glove. The glove processes these haptic commands and restricts movement of the user’s fingers.

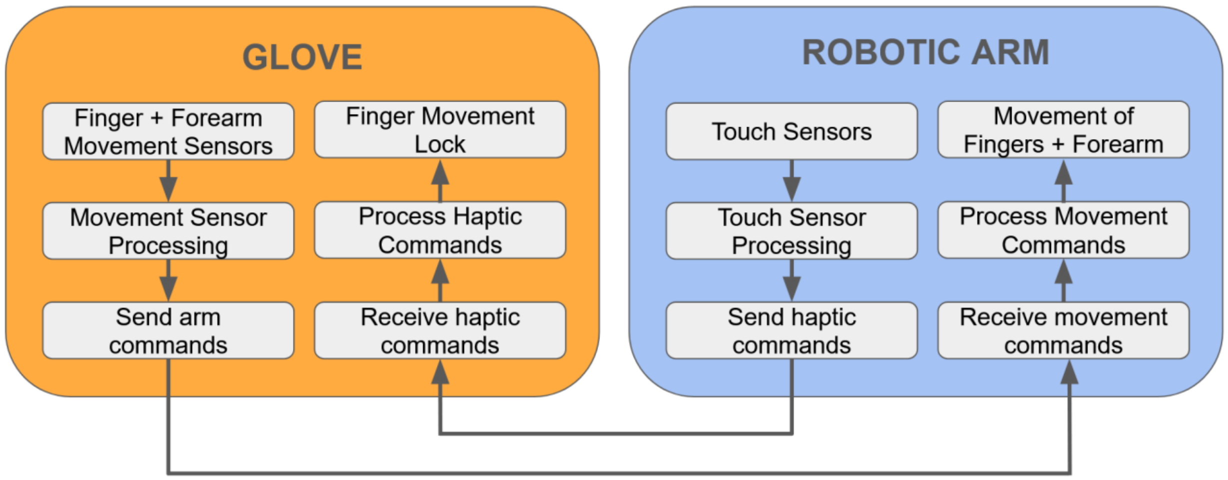


Figure 1: Complete System-Level Diagram

Robotic Arm Mechanical Design

The robotic arm is based on the open-source DexHand project, with modifications to increase its functionality. The arm includes:

- **Hand and Fingers**: The robotic hand features 16 degrees of freedom distributed among five fingers, with 3 DOF for each finger and 4 DOF for the thumb. Finger joints are designed to replicate human-like dexterity.
- **Forearm and Wrist**: The forearm rotates about its axis, and the wrist features two degrees of freedom for flexion/extension and radial/ulnar deviation.
- **Arm Base**: A custom-designed elbow base with one degree of freedom, securely attached to a flat surface (e.g., table or rolling cart) to provide stability.

The hand, fingers, wrist, and forearm are shown in Figure 2. All finger and wrist parts are 3D-printed using stereolithography (SLA) technology [7] with ABS-like resin for high precision and durability. SLA printing ensures smooth finishes and tighter tolerances, critical for the mechanical coupling of parts. The shell of the forearm and the arm base are 3D-printing using fused deposition modeling (FDM) technology [8] using PETG filament.

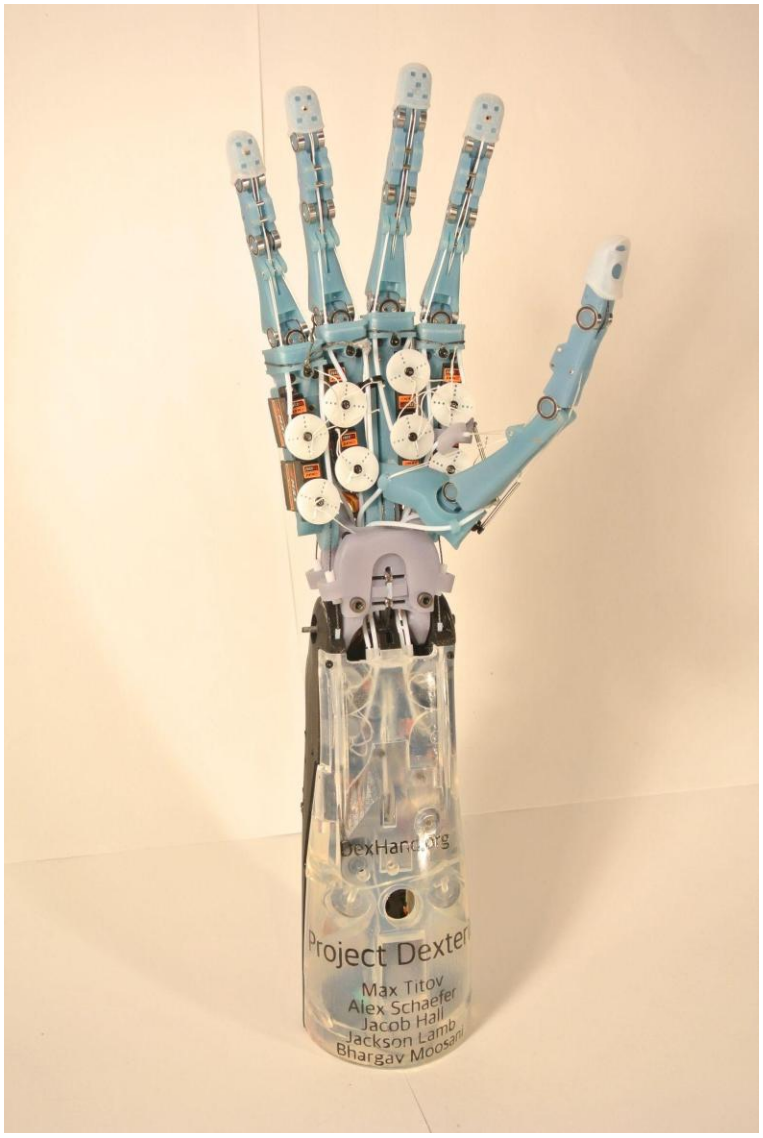


Figure 2: Complete Robotic Arm

The robotic system employs two types of motors mounted into designed slots within the 3D-printed frames, secured using screws:

- **FeeTech Serial Servos**: Used for wrist and forearm rotation, these servos allow precise positional control through serial communication. They also feature overcurrent control
- **PWM-Controlled Servos**: Used for individual finger joints, these motors provide fast and but less accurate angular movement.

Joints and linkages are constructed with:

- **Metal Rods**: To ensure durability under stress.
- **Flexible Tendons**: Nylon cords are routed through pulleys to mimic human tendons, transmitting motor torque to finger joints.

- **Bearings and Bushings**: Provide smooth joint movement and reduce wear on rotating parts.

Pressure sensors are embedded in the fingertips underneath the silicone sleeves using double sided tape. They provide real-time feedback on the force exerted during object manipulation. According to [9], males have an average pinch strength of 8.3kg and females have an average of 6.3kg.We chose pressure sensors with the highest max pressure value that still could fit on the fingertips of the robotic arm. The ones we chose have a range of 30g to 5kg.

Robotic Arm Electrical Design

This robotic arm control system is built around an ESP32S3 DevkitC microcontroller. A 6V 10A power supply directly powers the servos and uses a 3.3V low dropout regulator (LDO) to power the ESP32. The IMUs send data over I2C to the ESP32. The ESP32 uses an analog to digital converter (ADC) to read data from the pressure sensors which are set up in a voltage divider configuration. The digital servos are powered through PWM channels of the ESP32. Finally, the serial servos are controlled using a Feetech TLL Linker, which converts serial data from the ESP32 to half-duplex serial data that the servos can read. A diagram of this system is shown in Figure 3.

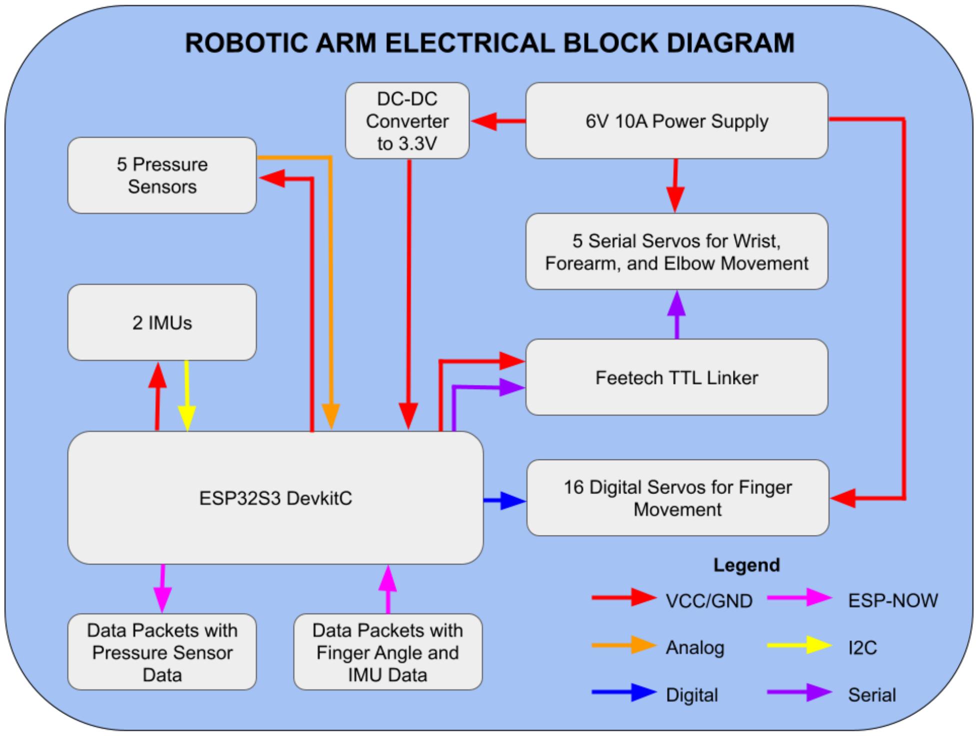

ROBOTIC ARM ELECTRICAL BLOCK DIAGRAM
DC-DC Converter to 3.3V
6V 10A Power Supply
5 Pressure Sensors
5 Serial Servos for Wrist, Forearm, and Elbow Movement
2 IMUs
Feetech TTL Linker
ESP32S3 DevkitC
16 Digital Servos for Finger Movement
Legend
VCC/GND
ESP-NOW
Analog
I2C
Digital
Serial
Data Packets with Pressure Sensor Data
Data Packets with Finger Angle and IMU Data

Figure 3: Robotic Arm Control Flow

Control Glove Mechanical

The control glove consists of 53 3D-printed parts. They are SLA 3d printed out of ABS-like resin. Joints that rotate are connected using 2mm pins that are cut to length. Parts that are fixed to each other are connected using M2 bolts and threaded inserts. The two photos below show the full mechanical design of the glove.

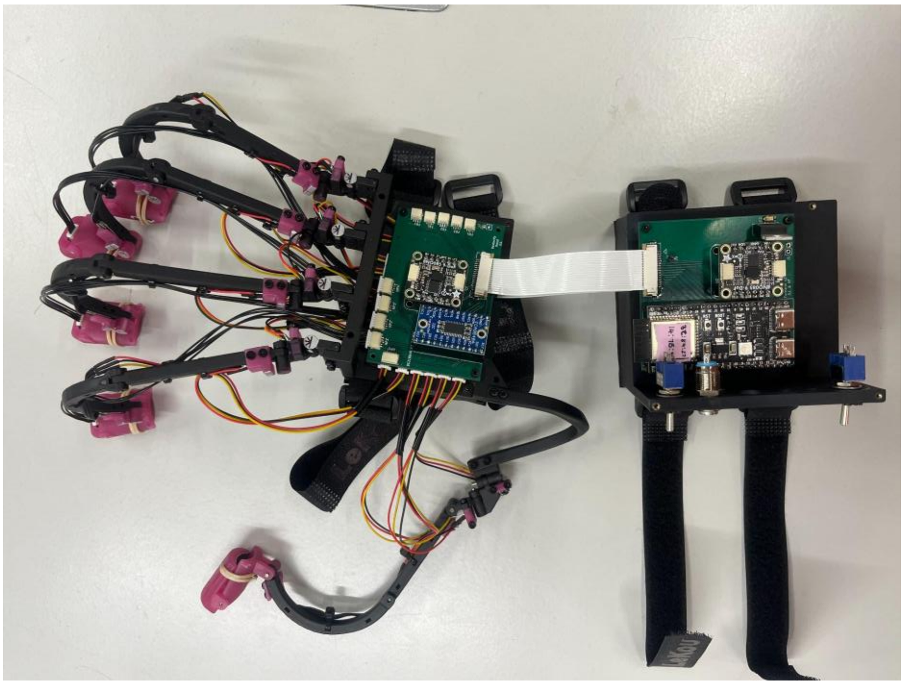


Figure 4: Top down view of the control glove.

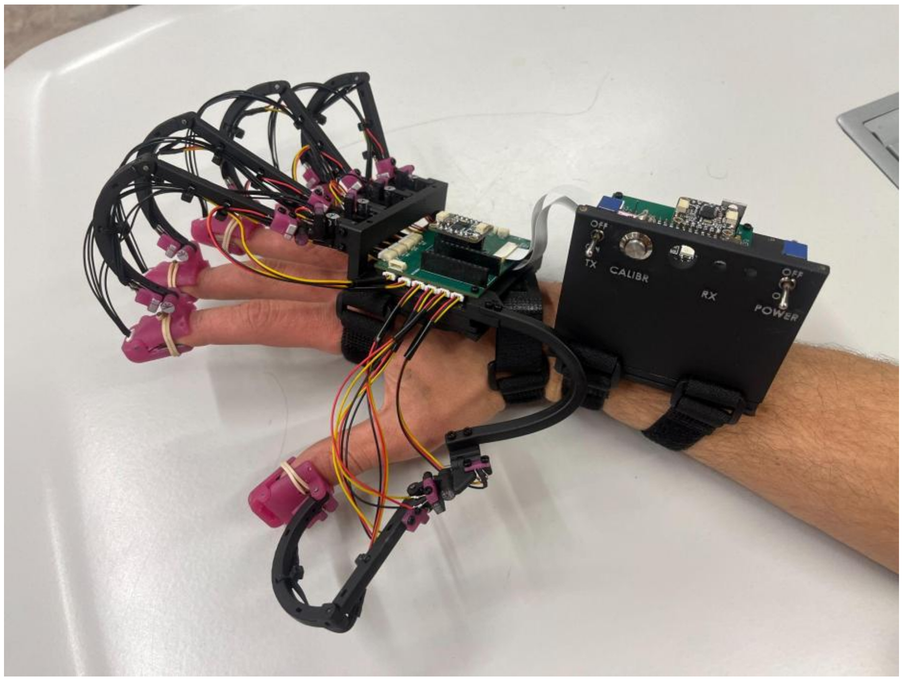


Figure 5: Side view of the control glove on a user

There are two enclosures as described below:

Hand enclosure

- Holds the hand PCB
- Has attachment points for the finger and thumb mechanical assemblies
- Has two straps to securely attach the enclosure to the back of the user's hand.
- Utilizes a piece of foam between the part and the users hand for comfort

Forearm enclosure

- Holds the forearm PCB
- Has two straps to securely attach the enclosure to the back of the user's forearm.
- Utilizes a piece of foam between the part and the users forearm for comfort
- Has a control panel on the side with 2 switches, 2 buttons, and 2 LEDs.

The tracking system for the glove consists of joints equipped with Hall-effect sensors and corresponding magnets. Each joint is designed to accurately capture the rotational movement of the user's fingers. An overview of the finger assembly is detailed below.

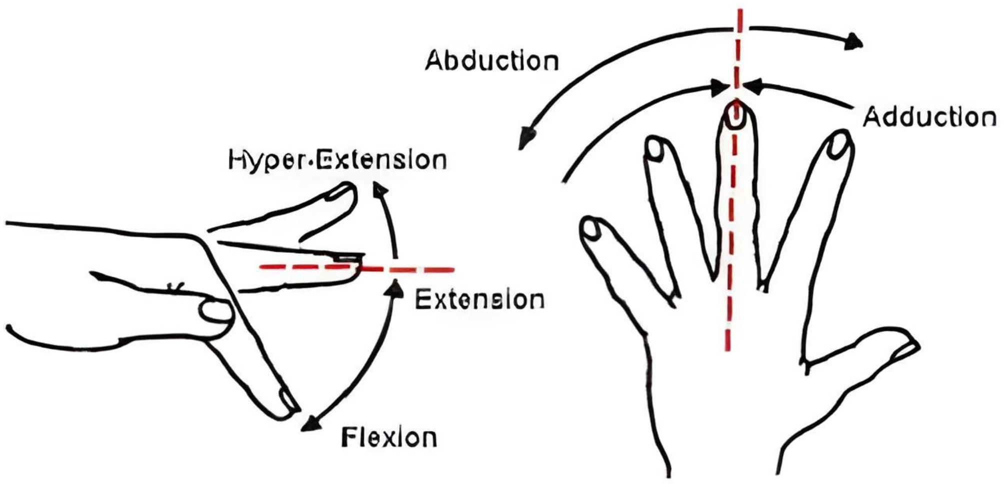


Figure 6: Demonstration of Measured Degrees of Freedom [10]

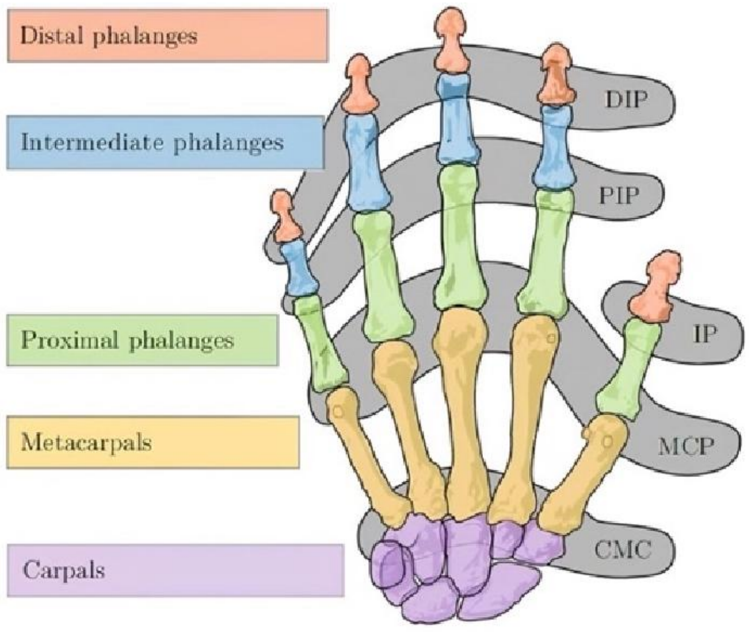


Figure 7: Definitions of finger joints [11]

Finger Assemblies

- Each finger assembly is made up of three main joints (PIP flexion, MCP flexion, and MCP abduction) to closely replicate human finger motion.
- Each finger has a finger attachment mechanism that clamps onto the finger using a rubber band. There is a LRA buzzer positioned on the tip of each finger.
- The joints are connected using 2mm steel pins and feature smooth rotational movement supported by plastic bushings to minimize wear and tear.

- Hall-effect sensors are mounted near each joint, and small neodymium magnets are attached to the moving parts. As the joints rotate, the relative position of the magnets to the sensors changes, producing a voltage output proportional to the angle of rotation.
- The thumb assembly is designed with an additional degree of freedom to mimic the opposable motion of a human thumb.

Control Glove Electrical

For sensing motion of the user's hand and forearm, the glove consists of 16 hall-effect sensors [12] and two IMUs. Hall-effect sensors have an output voltage correlated with the strength of the magnetic field at its location. By positioning a small magnet near each of the hall-effect sensors such that the dipole of the magnet rotates as the joint is, we can capture an output voltage correlated with the degree to which that joint has moved. The IMUs are used to capture acceleration and angular velocity data from the wrist and forearm. A diagram of the electrical connections is shown in Figure 8.

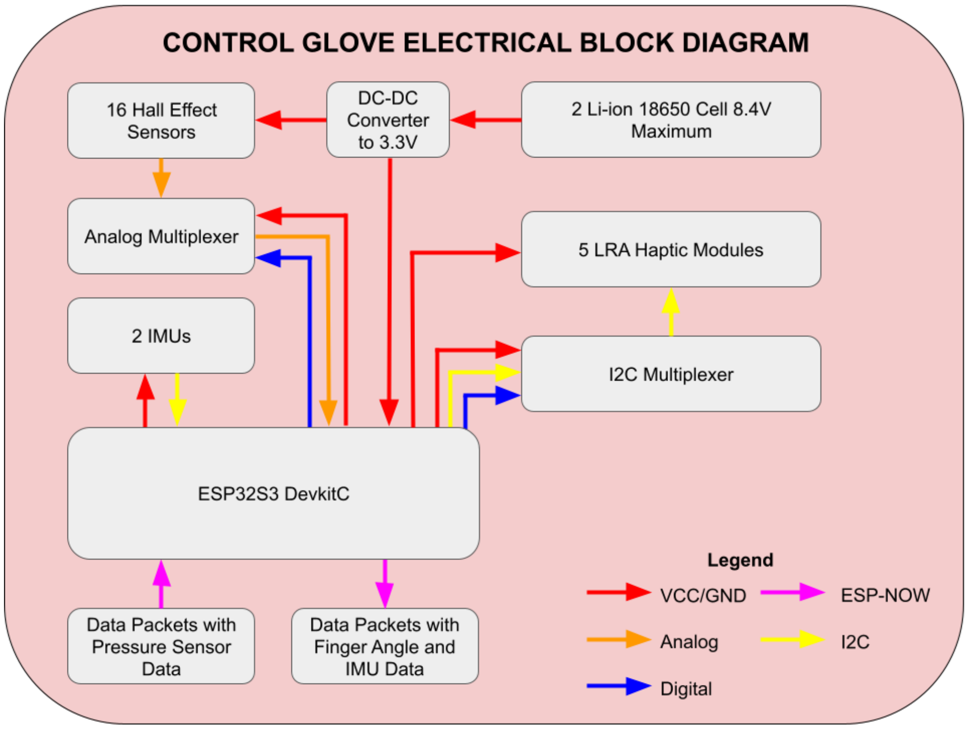


Figure 8: Control Glove Electrical Block Diagram

The electrical system consists of 2 PCBs, named the Hand PCB and the Forearm PCB. They are detailed below.

Hand PCB

- 16 connectors for the hall effect sensors wired up to a mux
- IMU
- I2C Mux thats wired up to connectors for the LRA haptic modules
  - The LRA haptic modules have fixed I2C addresses, so we have to use a mux to select between them.
- Ribbon cable that connects to the other PCB

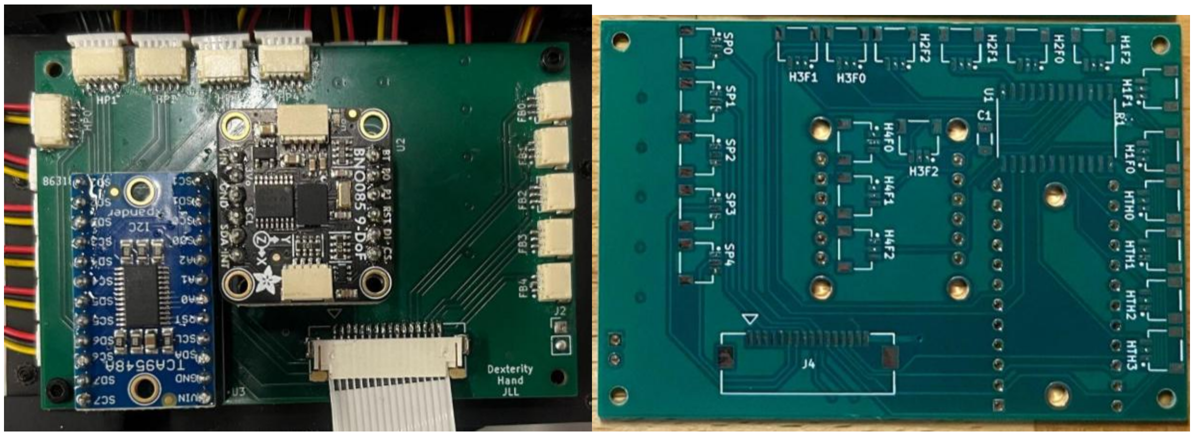


Figure 9: Top (left) and Bottom (right) of Hand PCBs

Forearm PCB

- ESP32S3 DevKitC-1 microcontroller
- IMU
- 3.3V LDO regulator that connects to two 18650 LiPo batteries
- Ribbon cable that connects to the other PCB

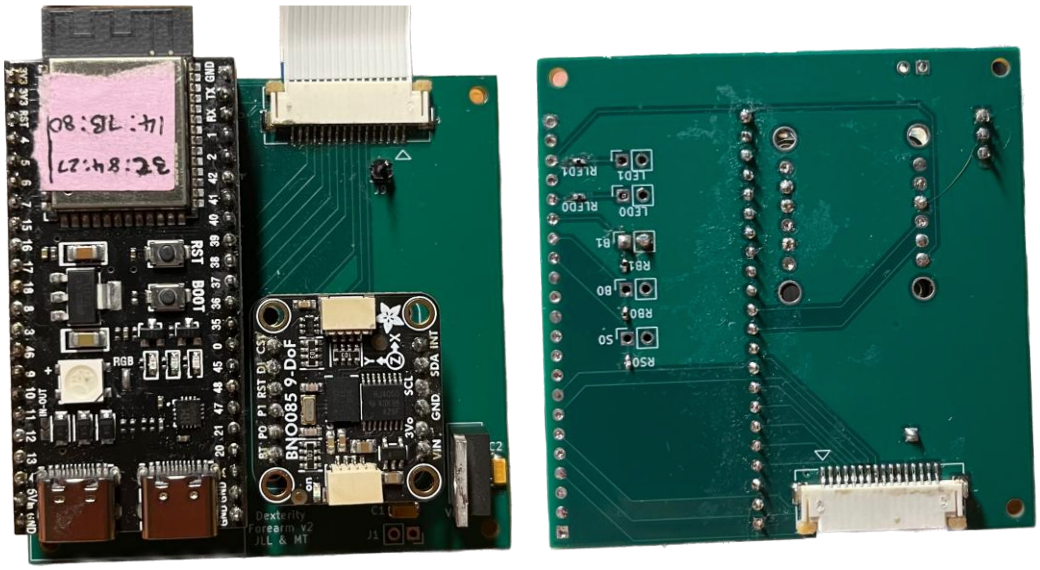


Figure 10: Top (left) and Bottom (right) of Forearm PCBs

The two PCBs are connected via a ribbon cable, which has power, hall-effect sensor selection lines, data lines for the hall-effect sensors, I2C[11] line for communication with the IMU, and another I2C line for controlling the LRA haptic modules. I2C is a communication protocol with the advantage of requiring only two lines for communication between an arbitrary number of devices.

Finger Tracking & Control

Each finger has 3 hall-effect sensors and the thumb has 4. These are used to measure the degree with which the finger is adducted, abducted, and flexed.. For each finger, one sensor is placed just over the knuckle joint whose output voltage is highly correlated with the degree of abduction and abduction. The upper and lower joints on the fingers each have a sensor whose output is correlated with their degree of flexion. The digital multiplexor on the back of the hand is used to select which hall-effect sensor to read data from using minimal selection lines. To aid in visualization of this aspect of the glove, a glove prototype consisting only of the microcontroller in a small form factor, all hall-effect sensors, and the multiplexor is shown in Figure 11 below.

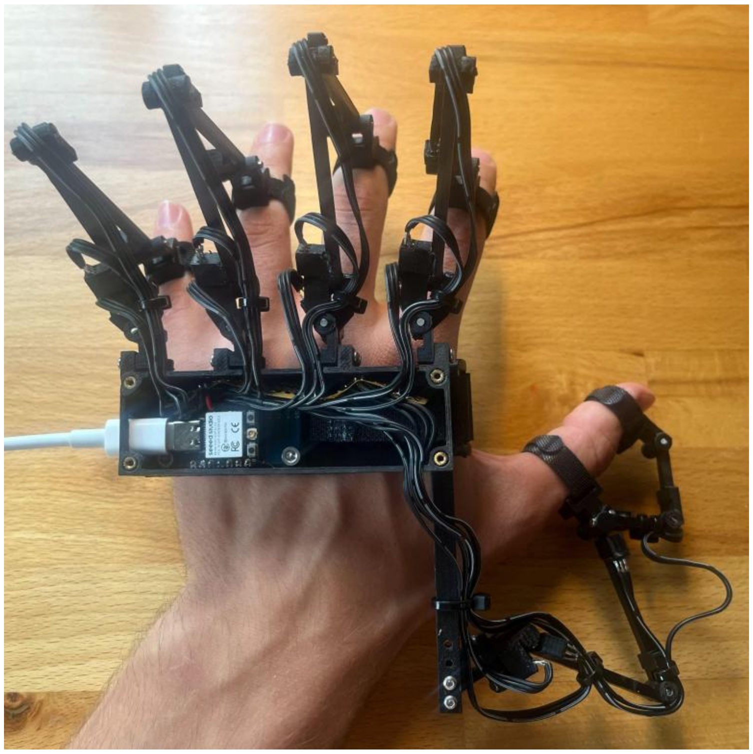


Figure 11: Control Glove Prototype

In the control glove, we read raw hall effect sensor values using an ADC. These values are not linear in relation to the angle the sensor is to the magnet. In order to linearize the data, we will take raw readings at several joint angles and create a polynomial fit. An example of the sensor readings to angle conversion data from the prototype glove is shown in Table 1 below.

Table 1: Example Hall-Effect Sensor Readings to Angle Conversion Data

Raw Value	Angle of Joint
2040	30
2110	60
2220	90
2370	120
2500	150

Using a polynomial fit calculator, we created the following polynomial for a joint on the pinkie, where y is the angle of the joint and x is the raw value, for example:

$$y = -0.000204869x^2 + 1.18023 - 1522.07$$

After doing the above steps on all 16 joints, the control glove produces linearized data that corresponds to the actual angle of each joint. These angles are then sent wirelessly to the robotic arm using the ESPNOW wireless communication protocol. When the angles are received at the robotic arm they are converted to commands for each of the finger servos.

Wrist Tracking & Control

The rotation of the wrist on the robotic arm has 2 degrees of freedom, specifically pitch and yaw. These rotation angles are open-loop controlled by two differential servos. Commands to the servos are converted from a Euler angle differential sent from the glove controller. This set of Euler angles consisting of yaw, pitch, and roll angles are calculated from the difference in orientation from the glove forearm PCB to the glove hand PCB.

To measure the orientation of the PCBs, IMUs (Inertial Measurement Units) with 9 degrees of freedom, which is simply a MEMS (micro-electromechanical) 3 axis accelerometer, 3 axis gyroscope, and 3 axis magnetometer combined into one chip, are placed on the hand and forearm PCBs for the control glove. Orientation can be represented as a quaternion, DCM (direction cosine matrix), or an euler angle sequence [13]. Quaternions consist of a real part and 3 imaginary parts (yikes!) and are much easier to perform calculations such as multiplication and addition on, but difficult to interpret qualitatively. Euler angles are easy to interpret, but difficult to perform calculations on. Typically, Euler angles are presented as a sequence of roll, pitch, and yaw angles, which can be seen in Figure 12 below. There are 12 possible sequences of Euler angles components.

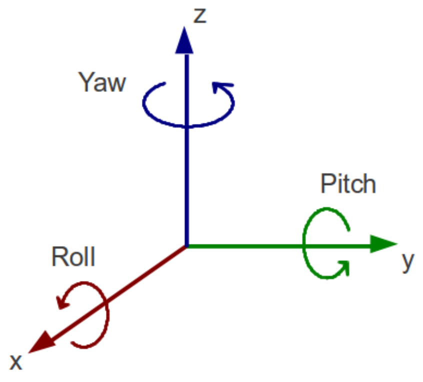


Figure 12: Euler Angles [14]

While none of the three MEMS sensors in the IMUs can provide estimates for the orientation in all three Euler angles reliably, the data from all of the sensors can be fused together to make orientation estimates reliable via a process called sensor fusion. Assuming that impulses (changes in acceleration) are kept relatively low, the gravitational vector due to the Earth can be extracted from the 3-axis acceleration measurements easily [15]. From this 3-dimensional, gravitational vector, which is relative to the IMU reference axes, we can calculate both a pitch and roll angle. A similar concept can be applied to the magnetometer. With the magnetometer, we can detect Earth's magnetic field, assuming there are no significant magnetic sources nearby the sensor, and extract the yaw angle (like a compass) [16]. The gyroscope provides angular rate data, which can be integrated into changes in angles over a specific time interval [17]. This can supplement the orientation estimates from the accelerometer and magnetometer. The gyroscope cannot provide an accurate estimate of the change in angle after a reasonable amount of time (tens of seconds to a few minutes for hobby grade sensors) because each axis has a constant, unremovable bias value. After integrating this constant value, the changes in angle estimates have an error that is equal to the constant times the time interval.

The CEVO/Hillcrest Labs BNO085 9DOF IMUs [18] were chosen because it has a 32-bit ARM® Cortex™-M0+ microcontroller on-chip that performs the sensor fusion briefly described above and outputs its orientation represented as a quaternion. These IMUs have many different possible orientation outputs, referred to as reports, which utilize variations of the sensor fusion algorithm optimized for different purposes such as AR/VR, gaming, etc. The report we chose was the "Rotation Vector" report, which provides the most accurate orientation estimate available for the sensor, according to the sensor's datasheet. The IMU supports several communication protocols including I2C (Inter-Integrated Circuit), SPI (Serial Peripheral Interface), and UART (Universal Asynchronous Receiver Transmit). The I2C communication protocol was chosen because of its simplicity both in terms of required physical wires/traces and code library support. Both IMUs are polled for their reports at 50 Hz using a timer-based interrupt service routine (ISR). The maximum rate the sensors can be polled is dependent on the specific report, but our chosen report could be polled at a much higher rate on the order of a few hundred hertz. 50 Hz updates to the orientation were chosen because updates at a higher rate than about 20 Hz are imperceptible.

To begin communicating with IMUs, it's necessary to pull the reset pin low on each to ready them for initialization over I2C. For simplicity, the reset pins for both IMUs can be electronically connected to the same GPIO pin on the ESP32. The Arduino-based I2C code library called TwoWire was utilized for I2C communication. A TwoWire object, representing an I2C bus with specific SDA (serial data line) pin, SCL (serial clock line) pin, and clock rate are initialized after resetting the IMUs. After an I2C connection is set up, the desired reports are requested from each IMU. This is done by sending a specific set of bytes including the report request ID defined by the datasheet of the IMUs. After reports are requested, it sometimes takes a few seconds for the reports to be non-zero. Thus, we wait until non-zero reports are received before processing.

After non-zero quaternion reports are being received by the control glove ESP32 from both IMUs, we begin processing the orientation data and sending commands to the robotic arm. We cannot send the IMU reported orientation of the hand directly to the robotic arm because the orientation is relative to the coordinate frame established by the earth gravity and magnetic field vectors. The wrist orientation needs to be relative to the forearm, so what we want to calculate is a different quaternion between the forearm and hand quaternions. To calculate that difference between quaternions, you can use the following formula:

$$q_{diff} = q_2 q_1^*$$

This calculates the rotation quaternion from the q1 to q2. Multiplication of quaternions is a lengthy calculation with nuances that won't be described here, but is easily found online [19]. After calculating the difference quaternion, we can convert the orientation into an Euler angles representation using a yaw-pitch-roll calculation sequence [20]. After this wrist rotation is calculated, the three euler angles are sent wirelessly to the robotic arm.

At the robotic arm, the wrist rotation data received and the servo motors commanded. In order to have a high range of motion and increased torque, FeeTech Serial Servos were chosen instead of utilizing the same servos as the fingers. These servos are commanded by a serial data line with a 16 bit (0 to 1023, inclusive) input. The joint kinematics of the wrist are such that they are controlled by two of these servos in a differential configuration. Mathematically, servo angle for the left and right pitch servos are defined below:

Left-Pitch Servo Angle = pitch + yaw
Right-Pitch Servo Angle = yaw - pitch

Thus, as wrist rotation commands are received, the above calculation can be performed using floating point math and mapped to a 16-bit integer value. To perform this mapping, a float angle to integer conversion constant, that is simply 1023 divided by 360 degrees, is multiplied by the servo angle and casted to a 16 bit integer. After this the center position of each respective servo is added to the command. To send the commands to the servos, the SCServo code library from the DexHand project [21] was used. We used a specific function, WritePos(), in this library that allowed us to set the speed of motion as well. We were able to tune this speed qualitatively to balance the quickness in response to a command and the jitter of the motion.

Haptic Feedback

The haptic feedback was implemented using a system consisting of Tiny Circuits LRA (linear resonant actuator) drivers [22] on the control glove and force-sensitive resistors (FSR) on the robotic arm. On a basic level, the FSRs on the fingertips of the robotic arm drive tactile feedback to the corresponding LRA on the control glove, allowing the glove to sense and feel the robotic arm interacting with its surrounding environment. The LRAs consist of an metal pad connected to an Texas Instruments DRV2605 chip via an I2C bus. The DRV 2605 chip stores 118 vibrational settings with varying sensations and frequencies. For the purposes of our project, five settings were chosen to simulate applying variable pressure as an object: the more pressure the robotic arm applies on an object, the greater the strength and frequency of vibration. The LRA haptic settings are driven by an ESP32 microcontroller.

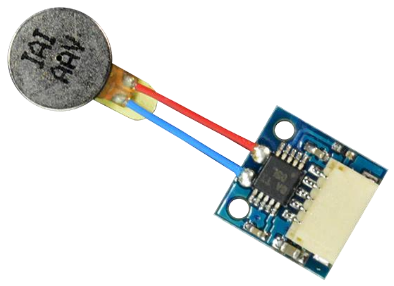


Figure 13: Tiny Circuits linear resonant actuator [22]

The FSR is a variable resistor that changes its electrical resistance in response to an applied force or pressure. When pressure is applied to the surface of the sensor, the material’s resistance decreases, allowing more current to pass through. The change in resistance is proportional to the force applied, and is used to drive the haptic settings on the LRA. Shown below is an image of the FSR as well as a schematic of the FSR configuration:

Figure 14: Force-sensitive resistor [23]

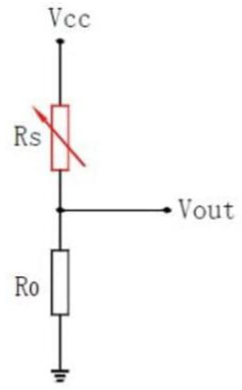


Figure 15: FSR Voltage Divider Schematic

The five FSRs (one for each fingertip) are connected to an ESP32's ADC channel, in which the values are stored into a 12-bit register. Thus, there are 4,096 digital pressure values available to associate to the five haptic settings. The fixed resistor (R_0 in Figure 15) in the voltage divider controls how much of the voltage range (and thus pressure range) is used by the ADC: ideally, with a 3.3V supply voltage from ESP32, we would like to utilize all 3.3V of the divider. A 10kΩ resistor as the fixed resistor gives us the best "resolution" for maximizing the pressure range that is used.

Looking at the system as a whole, the FSRs and LRAs communicate wirelessly as they are driven separately using the ESP32s on the robotic arm and control glove, respectively. This is achieved with the antennas that are available on our microcontrollers. Once the FSR reading is processed by the ADC channel, the microcontroller sends the data packets wirelessly to the microcontroller on the control glove. The control glove code is designed to separate the received ADC values (0 to 4095) into five segmented ranges, each associating to a one of the haptic settings. The five LRAs are connected to an I2C mux, which selects the corresponding LRA to activate (with the selected haptic setting) when the associated FSR on the robotic arm reads pressure.

Embedded Software Design

To connect all subsystems into one coherent system, we utilized PlatformIO as our project manager. By choosing PlatformIO, we could compile all code described above into separate libraries that could be referenced anywhere within a given project, while still utilizing the user-friendly Arduino framework. Within PlatformIO, we created two projects: one on the haptic glove ESP32 and one on the robotic arm ESP32. Figure 13 shows the high level overview of the two projects.

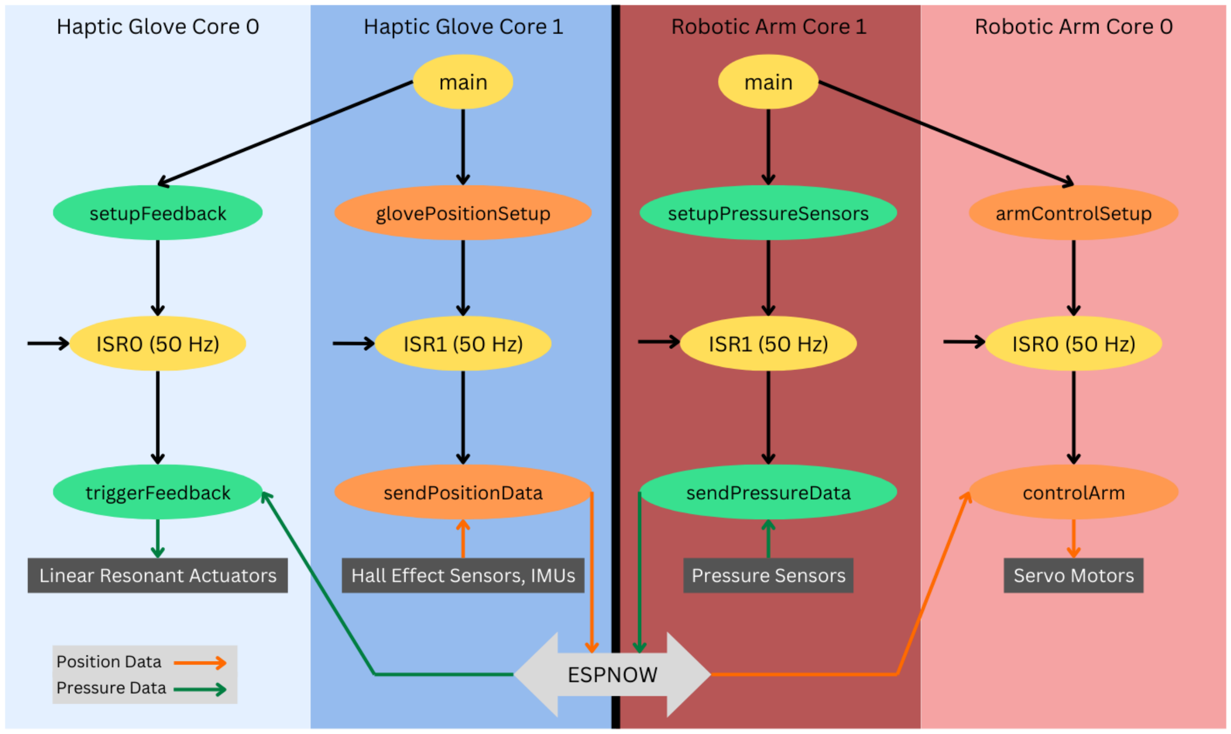


Figure 16: Flowchart of Embedded Software

In each ESP32, the main method is started in Core 1 by default. We use the main method to kick off the processes of setupFeedback and glovePositionSetup on the haptic glove side and setupPressureSensors and armControlSetup on the robotic arm side. In order to make efficient use of the ESP32 resources and ensure timely processing of data, we utilized both available cores. We used the FreeRTOS libraries to manage ESP resources, splitting up resources by overall functionality. The pipeline of pressure sensor data to LRA commands all executes within one core of each ESP32, while the pipeline of Hall-effect sensor data and IMU data to servo motor commands execute within the others. We initialize the necessary peripherals and communication methods in each setup function. We also use the setup functions to register the associated data processing actions to a timer-based interrupt service routine (ISR) - which are also managed by FreeRTOS libraries. By using FreeRTOS for management of both ISRs and tasks on different cores, a consistent interface is present for ensuring that no segments of code are stalling or using up unnecessary resources. ISRs are used to trigger data processing at a 50

Hz rate for each respective process on each core. The function sendPressureData reads the pressure sensors, categorizes the reading into one of five buckets, and sends the bucket number to the haptic glove ESP. In turn, the triggerFeedback function retrieves the data sent and commands the LRAs based on the respective bucket received. Similarly, the sendPositionData function reads data from the Hall Effect sensors and IMUs and sends the data to the robotic arm ESP. In turn, the controlArm function retrieves the data sent and translates the position data into servo motor commands. We use the libraries esp_now.h, WiFi.h, and esp_wifi.h to generate the sending and receiving functions that utilize the ESPNOW protocols. Instead of calling the triggerFeedback and controlArm functions when data is received, we instead write the data to a struct to be referenced by these methods. This is done such that the LRAs and servos are controlled at a 50 Hz rate regardless of whether or not packets are dropped.

System Testing Plans

1. **PCB Testing**
    a. Perform fly wire testing for both the hand and forearm PCB prior to populating the boards. This is simply testing to make sure that the known test points/traces that should be connected electrically have no resistance between them.
    b. With the ribbon cable installed between both the hand and forearm PCBs, perform fly wire testing for connection points across the ribbon cables.
    c. With the forearm and hand PCBs connected via the ribbon cables and the forearm PCB powered on, verify that hand PCB has power as well.
    d. After populating the boards, verify that the microcontroller, forearm IMU, hand IMU, and I2C multiplexor have power.
2. **Haptic Feedback**
    a. Haptic Setting Selection/Cycling
        i. Selected five out of the available 118 haptic settings to provide realistic, progressing feedback for variable pressure. To ensure it was realistic, I programmed a loop that cycled through the settings, progressing from lowest to highest pressure.
    b. Local Integration of Pressure Sensor with Single LRA
        i. Configured an FSR with the LRA using the same ESP32
            1. Constructed a voltage divider circuit with the FSR and a 10kΩ resistor
            2. Developed LRA pressure segmenting code, which associated five different pressure ranges with the ADC values from the FSR.
    c. Wireless Integration of Pressure Sensor with Single LRA
        i. Worked with Alex and Jackson to integrate ESPNOW library with the Haptic and ForceSensor libraries

ii. Configured both ESP32s (robotic arm and control glove) to drive the haptic settings wirelessly
iii. Verified that data packets were being sent and received reliably from both ESP32s

d. Integrate Wireless Haptic System with I2C MUX

i. Configure all five FSRs with the robotic arm's ESP32
ii. Then, configure all five LRA with the I2C mux, ensuring that the LRAs are being driven by their corresponding FSRs

e. Complete Haptic System Test

i. Attach pressure sensors to robotic arm and test that the haptic feedback system is working in its entirety as intended

3. Finger Tracking Tests

a. Calibration and Initial Functionality

i. Check that the VCC pin is at 3.3V
ii. Calibrate sensors by mapping raw outputs to known angles (e.g., 0°, 90°, 150°) using polynomial fitting.
iii. Test smooth and continuous output during full-range joint motion at a sampling rate of 50 Hz.

b. Data conversion and range of motion calibration test

i. Test that the finger range of motion calibration works by having multiple people try the glove on and see if we can calibrate to various hand sizes
ii. Ensure that the min and max values match what we expect the robotic arm to receive based on min and max servo values.

c. Individual Servo Configuration

i. Connect digital servos to the ESP32 PWM channel
ii. Test movement to predefined positions, ensuring accuracy within 1% tolerance.
iii. Verify the full range of motion and record center positions for calibration.
iv. Check for anomalies such as stalling or overshooting during operation.

d. Servo Integration with ESP32

i. Connect the URT-1 board to the ESP32 and control the servos using PWM signals.
ii. Test communication reliability and responsiveness between the ESP32 and servos.
iii. Calibrate neutral positions for use in the robotic finger assembly.

e. 3. Full Finger Movement Pipeline

i. Assemble the servo-driven finger mechanisms and ensure proper tendon tension for free joint movement.
ii. Test flexion and extension movements to match expected ranges.

iii. Verify accurate mapping of finger tracking data to servo positions in real time.
iv. Conduct live tests with the control glove to ensure smooth and responsive finger mirroring.
v. Confirm simultaneous servo movements operate reliably without latency or power issues.

4. **Wrist Control**
    a. **IMU Angle Differential**
        i. Verify that IMUs are initialized and calibrated according to desired specifications
        ii. Set hand IMU to known pitch and yaw angles relative to the forearm IMU and verify that differential rotation calculations are correct
    b. **Angle Data Wireless Communication**
        i. Test sending real-time, angular orientation data at 50 Hz from the glove ESP32 to the robotic arm ESP32.
        ii. Verify that data is received by the robotic arm with very low drop rates and floating point precision is maintained
    c. **FeeTech Servos**
        i. Connect each servo to the FeeTech URT-1 board through one long daisy chain, and connect that board to a computer via a micro-usb cable. Using the FeeTech FD software, search for connected servos. Select each servo's unique ID number and verify that the command for driving each servo to a specific position with a given speed and with a given timeout result in matching data in the graph provided by the FD software with a 1% tolerance
        ii. Connect the URT-1 board to an ESP32, and use the DexHand SCServo [21] functions to send each servo to a given position. Then, connect the URT-1 board to the computer running the FD software and verify that the position of each servo is within 5 ticks (0.5%) of the specified position
    d. **Servo Kinematics**
        i. During integration of the wrist servos in the mechanical assembly, verify that both servos center positions are at half their maximum range and that the cable tendons are tensioned
        ii. After integration, sending command angles to the servos to verify that the desired range of motion is reachable. Note the limits of the range of motion, specifically the angle of each servo.
        iii. Set the commanded pitch and yaw angles to known values and verify that the servo angle equations produce the same pitch and yaw angles on the robotic arm when used to command the servos.
    e. **Full Wrist Control Pipeline**

        i. Test the full pipeline of received IMU data, calculating the wrist angles, sending & receiving them wirelessly, and calculating & sending the servo commands.
        ii. Set the pitch and yaw angles of the hand IMU relative the forearm IMU to known values and verify that the robotic arm wrist rotates to those angles.

5. **Embedded Software Integration**
    a. **ESPNOW**
        i. Verify that >90% of ESPNOW packets are received at a 50 Hz sending rate after connection is established between two ESP32s by comparing the number of packets sent by one ESP to the number of packets received by the target ESP
        ii. Verify that the data sent by ESPNOW on one device is equal to the data received by the other
    b. **Timer-Based ISRs**
        i. Verify that the timer-based ISR causes the appropriate function to start in the appropriate core (ISR1 callback in core 1, ISR0 callback in core 0)
        ii. Verify that the appropriate function starts at the expected 50 Hz rate. Do so by setting an integer equal to 0 and then incrementing and printing the value every callback. Use a stopwatch to time until the printed integer is 200, and ensure that the measured time is between 3.8 and 4.2 seconds (5% measurement error)
    c. **Core Usage**
        i. Print the current core running each function one by one. Ensure that all functions called directly or indirectly from setupFeedback, triggerFeedback, armControlSetup, and controlArm are running in core 0 and all functions called directly or indirectly from glovePositionSetup, sendPositionData, setupPressureSensors, and sendPressureData are running in core 1

<u>**Physical Constraints**</u>

Because we are building upon the open-source Dexhand project, we already have a helpful guide for constructing the robotic hand assembly. This substantially reduces the difficulty of implementing the robotic hand, arguably the least familiar and most intimidating part of this project for us ECE majors. With this obstacle removed, our project faces only limited physical constraints.

Part Availability

The vast majority of the parts required for our project are either off-the-shelf components or 3D-printed. Our PCBs are the only exception to this general rule, but they can be readily ordered for manufacturing. Therefore, lack of part availability should not constrain our project.

Manufacturability

Beyond a healthy amount of troubleshooting, assembling our project once the parts were gathered was fairly straightforward. However, most of the 3D-printed parts and PCBs required at least a couple days to become available to use. Luckily this was a shorter turnaround than expected, and some of this time burden may be further alleviated by printing parts or ordering PCBs for manufacture in parallel. We definitely recommend carefully planning the manufacturing process to ensure that all required parts are obtained with time to spare for remaking parts if necessary.

Cost Constraints

In the absence of external funding, our maximum budget is $1,000. This consists of $500 from the ECE department in addition to $100 from each of our five group members. At the time of writing, we do not anticipate and would rather avoid spending this maximum budget, but we are ultimately willing to do so if it is required to complete our project. Thus, cost should not constrain our project.

While this is true, the servo motors used to move the robotic hand are the single most expensive component of our project. Purchasing or eschewing servos supporting a wider range of motion for the hand would make the difference between staying within the department budget and dipping into our own pockets.

Resources and Equipment

Outside of the components we purchase, we will require a 3D printer to produce parts for our robotic hand assembly, a soldering iron for assembling our PCBs, and lab equipment such as an oscilloscope to test our circuits and communications between the hand and glove. All of this equipment may be accessed in the NI Lab. Additionally, one of our group members, Max Titov, has a resin 3D printer in his workshop that was used to create most of the components for the robotic arm and all the 3D printed components of the control glove. Having a person printer means that other teams using the 3D printers in the NI Lab should not be a significant concern for us.

Software Tools

We will use KiCad for circuit schematic and PCB design, Visual Studio Code coupled with the PlatformIO extension for embedded code development, and SolidWorks for any required computer-aided design work. All of these software tools are either freely available for all users or free for all of our group members by virtue of being students. KiCad and PlatformIO in Visual Studio Code thankfully had very small learning curves that allowed us to jump in and start working quickly. However, it took us a while to learn their quirks.

Comments on a Potential Production Version

A potential production version of our project would likely bear a strong similarity to the prototype because a major factor differentiating our project from similar products is its comparably low cost. Thus, the most salient physical constraints of a production version would result from increasing the scale of production rather than changing the device itself. Producing our prototype at a commercial scale could decrease costs in some respects but increase them in others. As an illustration, purchasing components in bulk would drive down the per-unit cost of production, but license fees may need to be paid for using certain software tools for a commercial rather than purely academic purpose. Additionally, we would need access to a larger 3D-printing capacity to produce a sufficiently large number of units.

Societal Impact

The relevant stakeholders in our project can be divided into two groups: potential users and those affected by said use. Potential users include scientists managing hazardous materials in a laboratory setting. They ostensibly aim to maximize safety for themselves, members of their team, and the public.

Entities affected by the project's use may include the public at large, businesses, and governments. All these groups are generally concerned with the dangers posed by new technology, such as misuse and environmental impacts. The public at large wants to be safe from danger, businesses want to protect their employees, limit their liability, and maintain a good image, and governments strive to regulate technologies and protect their citizens.

Safety

Although our robotic arm promises to improve the safety of those handling hazardous materials, there is nonetheless a major safety concern worth considering: communication interruptions. These could arise intentionally or accidentally.

In the first case, a bad actor could intercept data between the glove and hand, convincing the latter that they are the glove. By doing so, the attacker could steal control of the hand from the user and issue actions ironically resulting in a safety hazard. For example, the attacker could trigger a premature or unwanted explosive reaction. Ideally, data transfer between the glove and hand would be mediated by a communication protocol featuring message encryption and signing. Such a protocol would ensure that third parties could neither see the contents of messages nor forge messages from the glove.

In the second case, messages between the hand and glove could be corrupted by interference. This could cause the hand to act in unexpected ways, such as freezing in the middle of pouring a volatile chemical into a reaction mixture and producing an unstable solution.

Sending a checksum along with each message and comparing each message against its associated checksum upon receipt would allow the glove and hand to detect and reject corrupted messages.

While communication interruptions are problematic, it should be noted that our robot arm is, by its very nature, intended to be operated over a distance sufficient to protect the user from the hazards posed by the materials being handled. For instance, a user handling explosives with our arm would be using it to avoid working within the blast radius anyway. Therefore, communication interruptions are likely as much of an issue of wasting materials or damaging property as they are of user safety.

A somewhat less acute safety concern is the force the glove applies to the user’s hand to provide haptic feedback. Care must be taken to ensure that such force is only as strong as it must be to provide feedback in order to prevent finger discomfort and injuries.

Unintended Uses

Despite our project’s intended use as a method for handling hazardous materials, we ultimately cannot control how our product is used once it ends up in customers’ hands. One concerning manner in which our product could be misused is the performance of medical or other procedures on humans and animals. Because we are not designing our robotic arm for the humane treatment of living things, injury could result from the device, for example, inadvertently applying too much force to a fragile limb.

Additionally, our project could conceivably be used to create hazardous materials for acts of terrorism instead of disposing of them. The robotic arm could empower bad actors to construct explosives, synthesize toxic chemicals, or extract bioweapons by making such activities safer.

These misuses will likely draw attention from the public at large who will abhor or fear them as well as governments who, feeling pressure from constituents, seek to regulate them. Hazardous materials whose creation was mediated by our robotic arm pose a national security threat for governments because such materials may be used to commit acts of terror.

Environmental Impact

Because our project largely uses off-the-shelf electrical components, its environmental impact is similar to other groups’ projects. However, our robotic arm distinctly relies upon a large number of 3D printed parts. The primary environmental concerns with these parts are waste and difficulty of recycling. Firstly, printing parts that cannot support themselves necessitates the printing of support structures, which will be discarded as waste once printing is complete. Secondly, the plastics used for 3D-printed parts, such as ABS and SLA, may either be rarely recycled municipality or not recyclable at all [24].

External Standards

We integrated several standards over the course of our project. The first set of standards that came into play related to PCB design. We found that JLCPCB requires many of the IPC standards to be met in order to pass the automated design for manufacturing (DFM) tests [25]. This includes IPC-2221 [26], which defines a wide range of PCB guidelines such as materials selection, board size, board shape, thermal management, component placement, trace width, space width, and via design. IPC-2222 [27] is also included, and further extends the criteria for rigid PCBs by providing more guidelines for materials, board configuration, terminal attachments, holes, and spacing. Additionally, many integrated circuits (ICs) that connect to the PCB follow the footprint recommendations from IPC-7351 [28]. As a result, we followed the same standard to mount the ICs. By following the IPC standards, we were able to create boards that could be manufactured by JLCPCB with our expected behaviors, while following common practices to interface with other ICs.

We also considered standards unrelated to PCBs. Since an integral component of our project is wireless communication, we ensured that we chose a microcontroller that complies with IEEE 802.11 standards for local area networks. Since the ESPNOW protocol is built off of IEEE 802.11, we have a pre-existing understanding of its underlying hardware and device addressing. Finally, we ensured that our enclosures for our PCBs satisfy the criteria for NEMA Type 1 [29]. By doing so, we are protecting those internal components from sizable external materials as well as unintended damage from the user.

In future works, there are several other standards that we would like to consider. For example, ISO 9241-940 [30] provides methods to establish benchmarks, establish requirements, and identify problems with haptic interfaces. However, we are still in the early stages of haptic development, and the interfaces described in this standard are more complicated than the LRA interface we chose. Additionally, ISO 10218-1 [31] lays out guidelines for safe design of industrial robots and requirements for protective measures to eliminate or greatly reduce hazards associated with the robots. While we did build a robotic arm, we made minimal changes to the design put forth by DexHand. In future works, we will evaluate how the DexHand robotics compare to this standard.

Intellectual Property Issues

To assess the patentability of our project, it is necessary to enumerate the essential components of our project and compare them against prior art as described in patents. Our project is a physical product with which the user controls a robotic hand remotely by leveraging movement tracking and haptic feedback. This yields the following four key elements:

A. robotic hand
B. remote control
C. movement tracking
D. haptic feedback

The patentability of an invention may not only be brought into question by the existence of a previous invention sharing all relevant characteristics of the present invention (i.e. a novelty or 102 rejection). Rather, an invention may also be deemed nonpatentable by arguing that it would have been obvious to one of ordinary skill in the art to create the present invention by combining elements of other inventions (i.e. an obviousness or 103 rejection). Therefore, all that is required to nullify the patentability of our project, in principle, is a series of prior art references that fit together.

Goupil et al. (US Pat. No. 10,809,804 B2) [32] discloses in independent claim 1 a "human-computer interface system comprising a fingertip assembly configured to apply a pressure to the finger of a user; said fingertip assembly including a finger panel comprising a plurality of tactors, including a tactor configured to apply a pressure to a palmar surface of the finger." This addresses the haptic signals provided by the haptic glove. Additionally, dependent claim 2 discloses the human-computer interface system of claim 1 further comprising a "back coupled to the fingertip assembly and a positional tracker coupled to the back configured to communicate a position in three-dimensional space of a hand." This dependent claim adds the element of the control glove tracking the position of the user's hand in space.

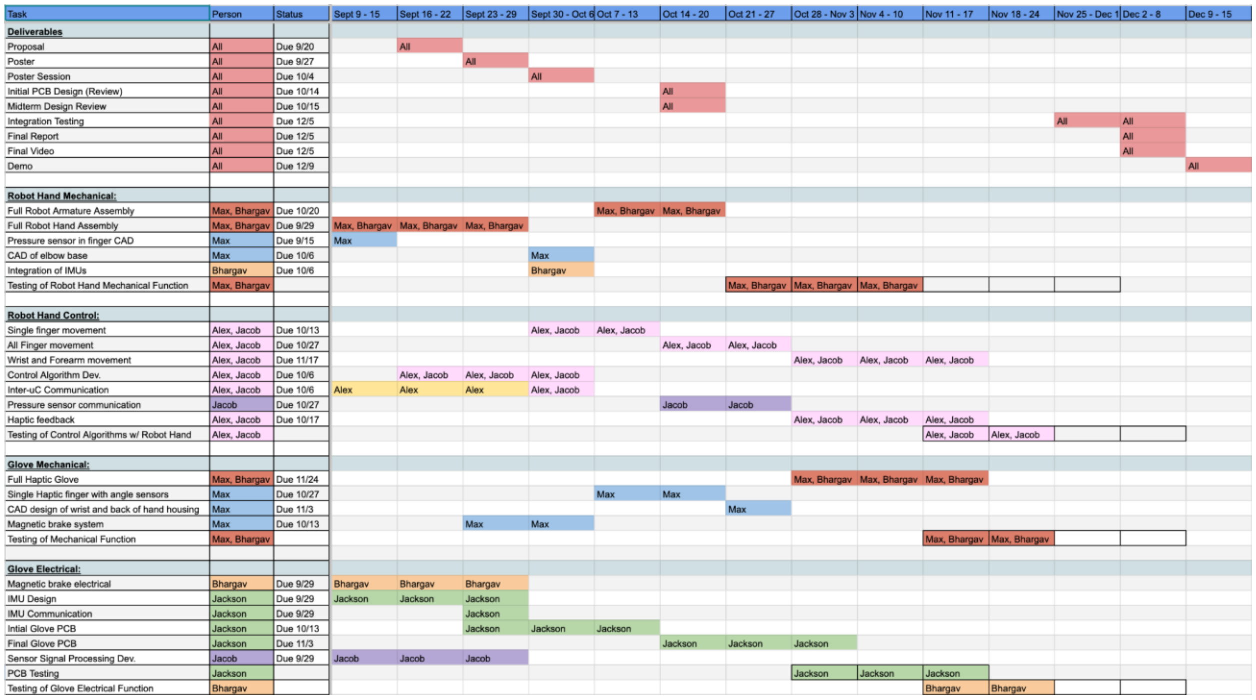
Hamdi et al. (US Pub. No. 20170144312A1) [33] discloses in independent claim 5 a "robotic grasping tool and controller with tactile feedback comprising a support; a plurality of robotic fingers mounted on the support, each of the robotic fingers having…a plurality of sensor modules…each of the sensor modules including a pressure sensor…" These limitations mirror our project's ability to provide haptic feedback based on what a robotic hand senses. The claim goes on to disclose a "plurality of control sheaths, each of the control sheath having a sheath housing adapted for receiving a corresponding finger of a user's hand; at least one joint angle sensor mounted in the sheath housing…such that movement of at least one joint of the finger of the user's hand is detected by the at least one joint angle sensor, and in response, the plurality of servomotors drive and control angular movement of the plurality of segments of the corresponding robotic finger; a plurality of tactile feedback modules mounted in the sheath housing, each of the tactile feedback modules being in communication with a corresponding one of the plurality of sensor modules." Again, these limitations drive home the idea of remotely controlling a robotic hand such that the user's movements are tracked and receiving feedback from sensors.

Levesque et al. (US Pat. No. US10613627B2) [34] disclose in independent claim 1 a system comprising in part a "a wireless receiver [and] a haptic output device configured to receive the haptic signal and generate a haptic effect to the user." Although Levesque et al. use an eye-gaze sensor as the input mechanism, the use of wireless signals to send haptic signals could reasonably be combined with the tracking and haptic feedback of Goupil and Hamdi as well as the robotic hand from Hamdi to synthesize a robotic hand remotely controlled with a haptic feedback glove tracking user movement.

From this perspective, our project is not patentable because it is obvious in view of prior art. However, it is worth making two notes. Firstly, patents are not the only publications which may be cited as prior art. For example, the DexHand project upon which most of ours is built may be sufficient to disqualify our project as patentable. Secondly, this analysis was extremely brief and bare bones. A patentability analysis could easily be as long as this entire report, and it is possible that a good patent agent or attorney could find one or more key elements that distinguish our project from similar pieces. That said, it preliminarily appears that our project would not be patentable.

Timeline

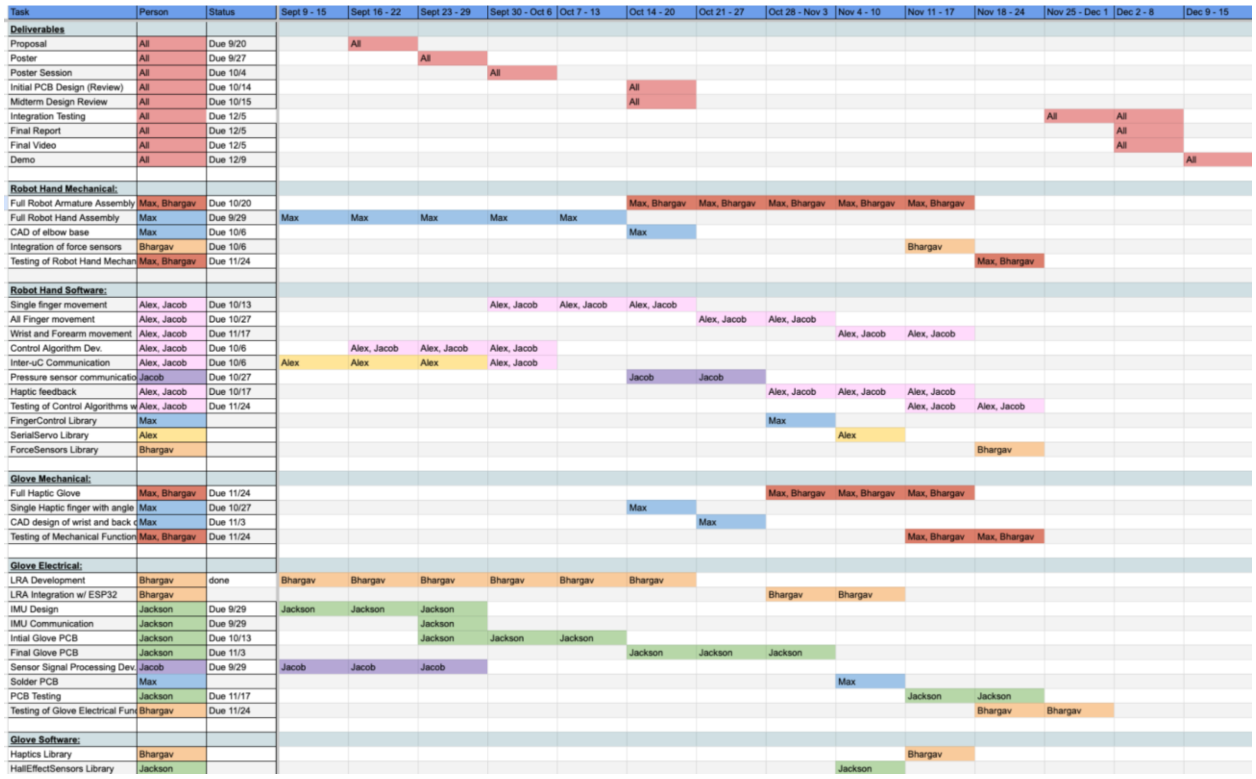
When planning the project in its initial stages, we created a Gantt chart that associated all of the required tasks for our project to a timeline. Shown below in Figure 17 is our initial Gantt chart: above showcases the Gantt Chart we will be following to keep us on track to meet our project goals:



Task	Person	Status	Sept 9 - 15	Sept 16 - 22	Sept 23 - 29	Sept 30 - Oct 6	Oct 7 - 13	Oct 14 - 20	Oct 21 - 27	Oct 28 - Nov 3	Nov 4 - 10	Nov 11 - 17	Nov 18 - 24	Nov 25 - Dec 1	Dec 2 - 8	Dec 9 - 15
Deliverables																
Proposal	All	Due 9/20		All												
Poster	All	Due 9/27			All											
Poster Session	All	Due 10/4				All										
Initial PCB Design (Review)	All	Due 10/14						All								
Midterm Design Review	All	Due 10/15						All								
Integration Testing	All	Due 12/5												All	All	
Final Report	All	Due 12/5													All	
Final Video	All	Due 12/5													All	
Demo	All	Due 12/9														All
Robot Hand Mechanical:																
Full Robot Armature Assembly	Max, Bhargav	Due 10/20					Max, Bhargav	Max, Bhargav								
Full Robot Hand Assembly	Max, Bhargav	Due 9/29	Max, Bhargav	Max, Bhargav	Max, Bhargav											
Pressure sensor in finger CAD	Max	Due 9/15	Max													
CAD of elbow base	Max	Due 10/6				Max										
Integration of IMUs	Bhargav	Due 10/6				Bhargav										
Testing of Robot Hand Mechanical Function	Max, Bhargav								Max, Bhargav	Max, Bhargav	Max, Bhargav					
Robot Hand Control:																
Single finger movement	Alex, Jacob	Due 10/13				Alex, Jacob	Alex, Jacob									
All Finger movement	Alex, Jacob	Due 10/27						Alex, Jacob	Alex, Jacob							
Wrist and Forearm movement	Alex, Jacob	Due 11/17								Alex, Jacob	Alex, Jacob	Alex, Jacob				
Control Algorithm Dev.	Alex, Jacob	Due 10/6		Alex, Jacob	Alex, Jacob	Alex, Jacob										
Inter-uC Communication	Alex, Jacob	Due 10/6	Alex	Alex	Alex	Alex, Jacob										
Pressure sensor communication	Jacob	Due 10/27						Jacob	Jacob							
Haptic feedback	Alex, Jacob	Due 10/17								Alex, Jacob	Alex, Jacob	Alex, Jacob				
Testing of Control Algorithms w/ Robot Hand	Alex, Jacob											Alex, Jacob	Alex, Jacob			
Glove Mechanical:																
Full Haptic Glove	Max, Bhargav	Due 11/24								Max, Bhargav	Max, Bhargav	Max, Bhargav				
Single Haptic finger with angle sensors	Max	Due 10/27					Max	Max								
CAD design of wrist and back of hand housing	Max	Due 11/3							Max							
Magnetic brake system	Max	Due 10/13			Max	Max										
Testing of Mechanical Function	Max, Bhargav											Max, Bhargav	Max, Bhargav			
Glove Electrical:																
Magnetic brake electrical	Bhargav	Due 9/29	Bhargav	Bhargav	Bhargav											
IMU Design	Jackson	Due 9/29	Jackson	Jackson	Jackson											
IMU Communication	Jackson	Due 9/29			Jackson											
Intial Glove PCB	Jackson	Due 10/13			Jackson	Jackson	Jackson									
Final Glove PCB	Jackson	Due 11/3						Jackson	Jackson	Jackson						
Sensor Signal Processing Dev.	Jacob	Due 9/29	Jacob	Jacob	Jacob											
PCB Testing	Jackson									Jackson	Jackson	Jackson				
Testing of Glove Electrical Function	Bhargav											Bhargav	Bhargav			

Figure 17: Initial Gantt Chart

Each of the respective subsystems was initially structured to be developed in parallel, with some tasks within each subsystem being achieved periodically before moving on to the next. We soon realized that this was not the case: for example, integration testing for the robotic arm required the robotic arm to be assembled, which was a lengthy and tedious process. As the project progressed, we realized that we had to change some components of the project, such as using the LRA-based haptic system instead of the electromagnetic brake system. Also, some tasks required more time than initially anticipated. Thus, we created a second version of our Gantt chart which not only updated our progress but also made changes to the pending tasks and their respective timelines:



Task	Person	Status	Sept 9 - 15	Sept 16 - 22	Sept 23 - 29	Sept 30 - Oct 6	Oct 7 - 13	Oct 14 - 20	Oct 21 - 27	Oct 28 - Nov 3	Nov 4 - 10	Nov 11 - 17	Nov 18 - 24	Nov 25 - Dec 1	Dec 2 - 8	Dec 9 - 15
Deliverables																
Proposal	All	Due 9/20		All												
Poster	All	Due 9/27			All											
Poster Session	All	Due 10/4				All										
Initial PCB Design (Review)	All	Due 10/14						All								
Midterm Design Review	All	Due 10/15						All								
Integration Testing	All	Due 12/5												All	All	
Final Report	All	Due 12/5													All	
Final Video	All	Due 12/5													All	
Demo	All	Due 12/9														All
Robot Hand Mechanical:																
Full Robot Armature Assembly	Max, Bhargav	Due 10/20						Max, Bhargav	Max, Bhargav	Max, Bhargav	Max, Bhargav	Max, Bhargav				
Full Robot Hand Assembly	Max	Due 9/29	Max	Max	Max	Max	Max									
CAD of elbow base	Max	Due 10/6						Max								
Integration of force sensors	Bhargav	Due 10/6										Bhargav				
Testing of Robot Hand Mechan	Max, Bhargav	Due 11/24											Max, Bhargav			
Robot Hand Software:																
Single finger movement	Alex, Jacob	Due 10/13				Alex, Jacob	Alex, Jacob	Alex, Jacob								
All Finger movement	Alex, Jacob	Due 10/27							Alex, Jacob	Alex, Jacob						
Wrist and Forearm movement	Alex, Jacob	Due 11/17									Alex, Jacob	Alex, Jacob				
Control Algorithm Dev.	Alex, Jacob	Due 10/6		Alex, Jacob	Alex, Jacob	Alex, Jacob										
Inter-uC Communication	Alex, Jacob	Due 10/6	Alex	Alex	Alex	Alex, Jacob										
Pressure sensor communicatio	Jacob	Due 10/27						Jacob	Jacob							
Haptic feedback	Alex, Jacob	Due 10/17								Alex, Jacob	Alex, Jacob	Alex, Jacob				
Testing of Control Algorithms w	Alex, Jacob	Due 11/24										Alex, Jacob	Alex, Jacob			
FingerControl Library	Max									Max						
SerialServo Library	Alex										Alex					
ForceSensors Library	Bhargav												Bhargav			
Glove Mechanical:																
Full Haptic Glove	Max, Bhargav	Due 11/24								Max, Bhargav	Max, Bhargav	Max, Bhargav				
Single Haptic finger with angle	Max	Due 10/27						Max								
CAD design of wrist and back c	Max	Due 11/3							Max							
Testing of Mechanical Function	Max, Bhargav	Due 11/24										Max, Bhargav	Max, Bhargav			
Glove Electrical:																
LRA Development	Bhargav	done	Bhargav	Bhargav	Bhargav	Bhargav	Bhargav	Bhargav								
LRA Integration w/ ESP32	Bhargav									Bhargav	Bhargav					
IMU Design	Jackson	Due 9/29	Jackson	Jackson	Jackson											
IMU Communication	Jackson	Due 9/29			Jackson											
Intial Glove PCB	Jackson	Due 10/13			Jackson	Jackson	Jackson									
Final Glove PCB	Jackson	Due 11/3						Jackson	Jackson	Jackson						
Sensor Signal Processing Dev.	Jacob	Due 9/29	Jacob	Jacob	Jacob											
Solder PCB	Max										Max					
PCB Testing	Jackson	Due 11/17										Jackson	Jackson			
Testing of Glove Electrical Fun	Bhargav	Due 11/24											Bhargav	Bhargav		
Glove Software:																
Haptics Library	Bhargav											Bhargav				
HallEffectSensors Library	Jackson										Jackson					

Figure 18: Final Gantt Chart

The second version of the Gantt chart kept us on track until we fell behind once more. This was inevitable, as our project is very involved and depends on assembling various complex subsystems together to achieve our design. Once we realized that the Gantt chart was not working for our organizational purposes due to the lack of parallelism of certain tasks, we implemented a to-do-list that was edited on a weekly basis. This allowed us to focus on developing the components that were pending with logical, technical progression. In the end, the majority of the tasks were completed the way we initially set out, with the two major factors affecting our Gantt chart and timeline being complexity, task dependency, and limited time.

Costs

The costs of the two systems are summarized in the two tables below. The most expensive components were the 20 servos for the robotic arm, and the haptic modules and IMUs on the control glove.

Robotic Arm			
Part	**Price**	**Quantity**	**Total**
ES3352 Servo	$10.94	16	$175.04
SCS2332 Servo	$28.83	2	$57.66
SCS15 Servo	$19.88	2	$39.76
TTLinker	$5.38	1	$5.38
3.3V DC-DC converter	$5.58	1	$5.58
ESP32S3 DevkitC	$6.99	1	$6.99
Ball bearings 6x10x3mm pack of 10	$8.69	5	$43.45
Ball bearings 2x6x3mm pack of 10	$10.19	1	$10.19
Ball bearings 15x21x4mm pack of 10	$9.59	1	$9.59
Ball bearings 3x8x4mm pack of 10	$7.99	1	$7.99
Sufix 832 Braid 80 lb 150 yards .33mm	$16.99	1	$16.99
Micro cord 100Ft 220Lb 0.8mm	$9.95	1	$9.95
MPU-6050 IMU breakout pack of 3	$9.99	1	$9.99
1x2mm 10 feet PTFE tubing	$7.99	1	$7.99
M3 x 0.5 mm Thread, 70 mm Long bolt	$4.26	2	$8.52
6V 10A wall power supply	16.99	1	$16.99
2 pos 28 guage wire 25ft	9.95	1	$9.95
dupont wire variety pack	6.98	1	$6.98
pressure sensors (FSR05BE)	6.59	6	39.54
Robotic Arm Total			**$488.53**

Figure 19: Robotic Arm Costs

Control Glove			
Part	**Price**	**Quantity**	**Total**

Hall effect sensor AH49E	$0.75	16	$12.00
ESP32S3 DevkitC	$6.99	1	$6.99
LRA Driver Wireling (haptic module)	$14.95	5	$74.75
5x 200mm 5 pin JST SH	$3.99	1	$3.99
PCB order 1	$23.76	1	$23.76
PCB order 2	20.6	1	$20.60
bypass cap for Hall effect max	$0.50	2	$1.00
Input/output caps for voltage regulator	$0.49	4	$1.96
Resistors	$0.01	10	$0.11
Hall effect mux	$0.58	2	$1.16
3.3V voltage regulator	$1.59	2	$3.18
Buck-converter breakout board	$8.50	1	$8.50
ribbon cables	$2.58	3	$7.74
Ribbon cable connectors	$2.00	5	$10.00
3-pin JST connectors	$0.48	34	$16.15
5-pin JST connectors	$0.86	7	$6.02
20x 3-pin JST cables	7.99	1	$7.99
IMUs	32.05	2	$64.10
Control Glove Total			**$270.00**

Figure 20: Control Glove Costs

Final Results

In terms of the functionality of our prototype, we met all of our performance objectives except for resistive haptic feedback and elbow movement. We completed the design of a new control glove with finger and wrist tracking and a sensory feedback system in the form of a buzzer on the fingertips. The finger tracking works as intended, converting the raw hall-effect sensor readings into angles and the robotic arm moves to these angles. The robotic arm finger movement is also very responsive. There is very little delay between the sensing and movement on the arm, almost imperceptible. We achieved a delay of around 0.04 milliseconds. Similarly, the IMU-based wrist orientation tracking works mostly as expected. There is a hardware bug in the I2C communication interface for ESP32s that sometimes prevents the connection from initializing. The actuation of the servo motors controlling the wrist provided the commands from the IMUs works as intended as well.

However, due to time constraints we were unable to implement 3DOF elbow movement. Initial prototyping and CAD had begun for the base but no advances were made towards assembly and testing. In addition, the original plan to implement resistive feedback using electromagnetic brakes was deemed unfeasible shortly after the proposal. There was little information online to aid in the design of this system as well as limited off-the-shelf components.

Our project rubric breakdown proposed in the proposal is shown in Figure 21 below. Figure 22 is the grading scale associated with this rubric. Complete movement is defined as the robotic arm moving in the correct direction for the fingers, wrist, forearm, and elbow as commanded by the control glove. We desired to have a reaction time of the robotic arm of less than 100 ms. Complete tracking is defined as measuring the raw sensor values associated with the finger, wrist, forearm, elbow and converting them to accurate (less than 10% difference) estimates of their position/rotation.

Figure 21: Project Rubric Breakdown

Points	Robotic Arm	Glove Sensing	Glove Haptics	Control Alg.
3	Complete (as defined above) movement of fingers, wrist, forearm, and elbow	Complete (as defined above) tracking of fingers, wrist, forearm, and elbow	The haptic glove provides real-time buzzer feedback, replicating the resistance faced by the robotic arm by creating a proportional buzzing sensation	Control algorithm allows for haptic glove mirroring and reasonably easy usage
2	Complete movement of 2 or more of the fingers system, wrist, forearm, and elbow	Complete tracking of 2 or more of fingers, wrist, forearm, and elbow	The haptic glove provides real-time buzzing feedback proportional to the resistance faced by the robotic arm, but there is a noticeable delay in the system	Control algorithm calibration provides some control over movements (robotic arm movement varies largely from glove movement)
1	Partial movement of fingers, wrist, forearm, or elbow,	Partial tracking of fingers, wrist, forearm, or elbow,	The haptic response of the glove is	Control algorithm is unstable or not robust

	or the delay is >0.1 seconds	or the delay is >0.1 seconds	inconsistent	
0	There is no movement	There is no tracking	There is no haptic response	Control algorithm is not implemented

Figure 22: Grading Scale

Points	Grade
11-12	A+
8-10	A
5-7	B
3-4	C
1-2	D
0	F

Engineering Insights

A variety of technical skills were developed across several subdisciplines of engineering: mechanical, electrical, and software. 3D printed parts were utilized a great deal in the mechanical design of the armature. We found that parts 3D printed with resin rather than FDM (fused deposition modeling) exhibited better properties such as smoothness and tolerance. In addition, a great deal was learned about robotic joint kinematic and motor control. Due to the high current draw from the servos, we learned that a separate high-power supply was necessary for the operation of the robotic arm. Different motors were utilized for fingers and wrist which required the team learning different input signal generation methodologies. The finger servos required usage of PWM signals and the wrist servos serial data signals. Something new about PCB design that the team did not have experience with was inter-dependent PCBs. Our control glove has two different PCBs: one for the back of the hand and one for the forearm. These PCBs are connected together with two ribbon cables. We desired to have a completely wireless communication between the control glove and robotic arm for our project. To do this we needed to learn a new communication protocol associated with ESP32 microcontrollers called ESPNOW.

In the way of time management, one issue was that the Gantt chart developed at the beginning of the semester was under utilized. We defaulted to more of a week by week plan. At the beginning of the week, we would meet to discuss next steps over the course of the week for each member. This worked well for most weeks, but longer time frame planning could have allowed us to rule out certain goals of the project earlier rather than later. Communication between team members was very efficient as individuals regularly provided the team with updates on the progress and hiccups of their tasks.

For software development, we utilized Git version control with a centralized repository of our PCB designs and code on Github. Team members would regularly make commits of their code updates and push them to the central repository. This allowed other team members working on the project to pull the most recent code developed by other members when needed. In addition, we established early on that only working code would be pushed to the central repository, so in the event that someone pulls the most recent commit from the repository and develops new, buggy code of that commit, they could revert back to the working code.

During the initial planning phase of the project, we designated specific roles for each group member based on their strengths. These roles were further assigned tasks that they were responsible for. This allowed the team to easily keep track of who was working on what. In the interest of efficiency, all team members remained versatile in the tasks they would work on. If a team member completed their tasks, they would work on tasks under the umbrella of another team member's role. This minimized downtime of team members working on the project and increased throughput of task completion.

One tip for future capstone students would be to budget time for fixing PCB errors, changing designs, and debugging code. Partway through the semester, our group discovered that the pinout had been flipped from what we expected with the 16-pin ribbon cable connector on the forearm PCB. It took a bit of testing to find this error, some time to reroute a new PCB, order it, get it shipped, populate the new board, and test it. All in all, PCB development and testing took two to three times as long as expected. This project was very code heavy which led to a lot of time spent debugging. It was not uncommon for a team member to say it would take a few days to finish up a library and then at the end of the week be still debugging a part of the library. On the team communication front, something very helpful to talk about for expectations for time spent working on capstone on a weekly basis. Some members might be taking a less time-intensive course load and therefore have more time to dedicate to capstone and vice-versa. It is important to establish both a minimum and maximum time per week on capstone. From there, teammates can work together to optimize their time to completing essential tasks.

Future Work

There are several improvements to this project that could increase learning outcomes and further advance the project technically. On the control glove side of the project, improvements could be made to the existing PCB, resistive feedback could be implemented, and/or other methods for estimating the angular difference between the hand and forearm could be explored. The existing glove PCB design(s) could be improved by soldering the ESP32 microcontroller, IMUs, and I2C multiplexer chips and associated components directly to the board (surface mount) rather than using breakout boards and header pin slots. In addition, the ribbon cables between the wrist and forearm PCB for the glove are very stiff. Perhaps a better solution could be found.

In our initial planning phase, we were hoping to implement a 3DOF (three degrees of freedom) elbow base. Late into the semester, this was deemed infeasible due to development times of more critical, time-consuming tasks like completing the robotic arm and control glove assemblies. A 3DOF elbow base would allow the robotic arm to move its forearm toward objects, increasing range of motion and utility substantially. Had the robotic forearm and wrist development been completed prior to the semester start, it is likely that both an elbow and shoulder base could be implemented, even further increasing the utility of the arm. Of course, this comes with added complexity in the controller and cost increases.

An unforeseen difficulty was mechanical development of both the robotic arm and control glove. Assembly of the robotic arm and mechanical development of the control glove fell onto one group member who hoped to finish them one after the other. In retrospect, these tasks should have been performed in parallel by two or more members. During the initial planning phase of the project, the complexity and probable delays in the first task (robotic arm assembly) should have been taken into account and the second task (control glove design and assembly) assigned to another teammate or split amongst several.

References

[1] T. Shumay, “DexHand,” DexHand.org. Accessed: Dec. 06, 2024. [Online]. Available: https://www.dexhand.org/

[2] “Gloves G1,” HaptX. Accessed: Dec. 06, 2024. [Online]. Available: https://haptx.com/gloves-g1/

[3] “SenseGlove for Virtual Reality prototyping,” SenseGlove. Accessed: Dec. 06, 2024. [Online]. Available: https://www.senseglove.com/solutions/senseglove-for-virtual-prototyping/

[4] “WEART haptic solutions | WEART.” Accessed: Dec. 06, 2024. [Online]. Available: https://weart.it/

[5] Lucas VRTech, *How to build cheap VR Haptic Gloves to FEEL VR.*, (Jul. 10, 2022). Accessed: Dec. 06, 2024. [Online Video]. Available: https://www.youtube.com/watch?v=2yF-SJcg3zQ

[6] NepYope, *I built VR Haptic Gloves to Control Robots*, (Jul. 16, 2022). Accessed: Dec. 06, 2024. [Online Video]. Available: https://www.youtube.com/watch?v=iPtgvh6fNdQ

[7] Z. Miller, B. Stidham, T. Fairbanks, and C. Maldonado, “The Use of Stereolithography (SLA) Additive Manufacturing in Space-Based Instrumentation,” in *2023 IEEE Aerospace Conference*, Mar. 2023, pp. 1–10. doi: 10.1109/AERO55745.2023.10115988.

[8] Md. M. Hoque, Md. M. H. Jony, Md. M. Hasan, and M. H. Kabir, “Design and Implementation of an FDM Based 3D Printer,” in *2019 International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering (IC4ME2)*, Jul. 2019, pp. 1–5. doi: 10.1109/IC4ME247184.2019.9036538.

[9] A. I. Michael, A. O. Iyun, O. A. Olawoye, S. A. Ademola, R. E. Nnabuko, and O. M. Oluwatosin, “Normal Values of Key Pinch Strength in a Healthy Nigerian Population,” *Ann. Ib. Postgrad. Med.*, vol. 13, no. 2, pp. 84–88, Dec. 2015.

[10] E. Peña-Pitarch, *Abduction/adduction(Ab/Ad), flexion/extension (F/E) for one finger*. Accessed: Dec. 06, 2024. [Online]. Available: https://www.researchgate.net/figure/Abduction-adductionAb-Ad-flexion-extension-F-E-for-one-finger_fig5_279493096

[11] S. Kim, “Figure 2. joints of real hand [11] As illustrated in Figure 2, there...,” ResearchGate. Accessed: Dec. 06, 2024. [Online]. Available: https://www.researchgate.net/figure/joints-of-real-hand-11-As-illustrated-in-Figure-2-there-are-19-joints-in-a-real_fig2_382915424

[12] “Linear Hall-Effect IC Data Sheet.” BCD Semiconductor Manufacturing Limited, Aug. 2010. Accessed: Dec. 06, 2024. [Online]. Available: https://www.diodes.com/assets/Datasheets/AH49E.pdf

[13] D. M. Henderson, “Euler angles, quaternions, and transformation matrices for space shuttle analysis,” DN-1.4-8-020, Jun. 1977. Accessed: Dec. 06, 2024. [Online]. Available: https://ntrs.nasa.gov/citations/19770019231

[14] “Joints - V3.2,” SoftBank Robotics. Accessed: Dec. 06, 2024. [Online]. Available: http://doc.aldebaran.com/2-5/family/robots/joints_robot_v32.html

[15] C. Fisher, “AN-1057: Using an Accelerometer for Inclination Sensing,” Analog Devices. Accessed: Dec. 06, 2024. [Online]. Available:

https://www.analog.com/en/resources/app-notes/an-1057.html
[16] C. Chao, Z. Jiankang, and J. Hu, “Improving robustness of the MAV yaw angle estimation for low-cost INS/GPS integration aided with tri-axial magnetometer calibrated by rotating the ellipsoid model,” *IET Radar Sonar Navig.*, vol. 14, no. 1, pp. 61–70, 2020, doi: 10.1049/iet-rsn.2019.0200.
[17] M. Paluszek, R. Perez, E. Canuto, C. Perez Montenegro, R. Langari, and A. Morris, “Integrating Gyro,” ScienceDirect. Accessed: Dec. 06, 2024. [Online]. Available: https://www.sciencedirect.com/topics/engineering/integrating-gyro
[18] “BNO08X Data Sheet.” CEVA. Accessed: Dec. 06, 2024. [Online]. Available: https://www.ceva-ip.com/wp-content/uploads/2019/10/BNO080_085-Datasheet.pdf
[19] D. Eberly, “Quaternion Algebra and Calculus,” Sep. 2002, Accessed: Dec. 06, 2024. [Online]. Available: https://www.sci.utah.edu/~jmk/papers/Quaternions.pdf
[20] N. H. Hughes, “Quaternion to Euler Angle Conversion for Arbitrary Rotation Sequence Using Geometric Methods,” *Braxton Technol.*, Accessed: Dec. 06, 2024. [Online]. Available: https://www.euclideanspace.com/maths/geometry/rotations/conversions/quaternionToEuler/quat_2_euler_paper_ver2-1.pdf
[21] “SCServo.h,” GitHub. Accessed: Dec. 06, 2024. [Online]. Available: https://github.com/TheRobotStudio/V1.0-Dexhand/blob/main/V1.0%20DexHand_12/SCServo.h
[22] “LRA Driver Wireling,” TinyCircuits. Accessed: Dec. 06, 2024. [Online]. Available: https://tinycircuits.com/products/lra-wireling-drv2605
[23] “FSR05BE,” DigiKey Electronics. Accessed: Dec. 06, 2024. [Online]. Available: https://www.digikey.com/en/products/detail/ohmite/FSR05BE/10127621
[24] “3D printing and its environmental implications,” in *The Next Production Revolution: Implications for Governments and Business*, Paris: OECD, 2017, pp. 171–213. Accessed: Dec. 06, 2024. [Online]. Available: https://www.oecd-ilibrary.org/science-and-technology/the-next-production-revolution/3d-printing-and-its-environmental-implications_9789264271036-9-en
[25] “PCB Manufacturing & Assembly Capabilities,” JLCPCB. Accessed: Dec. 06, 2024. [Online]. Available: https://jlcpcb.com/capabilities/pcb-capabilities
[26] *Generic Standard on Printed Board Design*, IPC-2221A, May 2003. Accessed: Dec. 06, 2024. [Online]. Available: https://www-eng.lbl.gov/~shuman/NEXT/CURRENT_DESIGN/TP/MATERIALS/IPC-2221A(L).pdf
[27] *Sectional Design Standard for Rigid Organic Printed Boards*, IPC-2222, Sep. 1991. Accessed: Dec. 06, 2024. [Online]. Available: http://www.edatop.com/down/faq/pads/pads-pcb-IPC-2222-%E5%88%9A%E6%80%A7%E6%9C%89%E6%9C%BA%E5%8D%B0%E5%88%B6%E6%9D%BF%E8%AE%BE%E8%AE%A1%E5%88%86%E6%A0%87%E5%87%86-504.pdf
[28] *Generic Requirements for Surface Mount Design and Land Pattern Standard*, IPC-7351, Feb. 2005. Accessed: Dec. 06, 2024. [Online]. Available: https://datasheet.datasheetarchive.com/originals/crawler/fancort.com/428761382a93d283e0ca120b5c72d850.pdf
[29] *Enclosures for Electrical Equipment (1,000 Volts Maximum)*, 10250–2024, Aug. 09, 2024. Accessed: Dec. 06, 2024. [Online]. Available:

https://www.nema.org/standards/view/enclosures-for-electrical-equipment-(1-000-volts-maximum)
[30] *Ergonomics of human-system interaction*, ISO 9241-940:2017, Nov. 2017. Accessed: Dec. 06, 2024. [Online]. Available: https://www.iso.org/standard/61362.html
[31] *Robots and robotic devices — Safety requirements for industrial robots*, ISO 10218-1:2011, Jul. 2011. Accessed: Dec. 06, 2024. [Online]. Available: https://www.iso.org/standard/51330.html
[32] M. Y. Goupil, B. L. Rojanachaichanin, K. C. Sjoberg, P. Piller, N. J. B. JR, and R. M. Takatsuka, "Haptic feedback glove," US10809804B2, Oct. 20, 2020 Accessed: Dec. 06, 2024. [Online]. Available: https://patents.google.com/patent/US10809804B2/en?assignee=HAPTX%2c+INC
[33] J. T. Hamdi, S. M. Munshi, and S. Azam, "Robotic surgical finger and controller with tactile feedback and robotic hand using the same," US20170144312A1, May 25, 2017 Accessed: Dec. 06, 2024. [Online]. Available: https://patents.google.com/patent/US20170144312A1/en
[34] V. Levesque, A. Modarres, D. A. Grant, J. F. DIONNE, and D. M. Birnbaum, "Systems and methods for providing haptic feedback for remote interactions," US10613627B2, Apr. 07, 2020 Accessed: Dec. 06, 2024. [Online]. Available: https://patents.google.com/patent/US10613627B2/en