Wearable Biomimetic Appendage

ELEC-4000: Capstone Design Project

Department of Electrical and Computer Engineering

University of Windsor

August 2nd, 2023

Submitted in partial fulfillment of the requirements for the course

Wearable Biomimetic Appendage

Faculty Advisor: Dr. Rashidzadeh

Steven Caro: 105209197

Gian Favero: 105215891

Uygur Tepe: 105006877

August 2nd, 2023

Wearable Biomimetic Appendage

No action by any design team member contravened the provisions of the Code of Ethics and we hereby reaffirm that the work presented in this report is solely the effort of the team members and that any work of others that was used during the execution of the design project or is included in the report has been suitably acknowledged through the standard practice of citing references and stating appropriate acknowledgements.

The presence of the authors' signatures on the signature page means that they are affirming this statement.

Signed: 8/2/2023

Steven Caro

105209197

Signed: 08/02/2023

Gian Favero

105215891

Signed: 08/02/2023

Uygur Tepe

105006877

Abstract

Author: Uygur Tepe

The Wearable Biomimetic Appendage is a device that can mimic a user's hand and finger movements to aid in basic tasks. The device is composed of three main components: a control glove, a calibration and controller program, and a wearable robotic hand.

The control glove uses Hall sensor-neodymium magnet pairs to track a user's finger movements with high precision. A graphical user interface was developed to aid in calibrating the glove, making it robust to unique hand sizes and shapes. The GUI also features a virtual reality environment to validate the effectiveness of the calibration before usage with a robotic hand. The robotic hand is able to mimic the user's finger movements, providing them with a physical augmentation of their own hand.

The device has the potential to be used in a variety of virtual- and real-world applications. In virtual reality, the device could be used to provide users with a more immersive virtual reality control and experience. In the real world, assistance can be provided to people with finger or hand disabilities, or provide workers with a more safe way to perform tasks.

Contents

Introduction

Author: Gian Favero

Background

There are a variety of environments and conditions that hinder one's ability to carry out tasks with their hands. Injury, a loss of grip strength, temperature sensitivity, skin sensitivity, or a risk of harm being present are a few of many applicable conditions. In Canada, hands are the most frequently injured body part with over 500,000 occurrences per year. Furthermore, 43% of people ages 65 and over live with arthritis on top of the natural decrease in grip strength that occurs over a lifespan [1]. In the presence of these conditions, users are forced to use more complicated methods when using their hands would be preferable.

Objective

This Capstone Design project intends to create a Wearable Biomimetic Appendage (WBA) that can mimic a user's hand and finger movements to overcome the conditions which may hinder their ability to carry out tasks. Features intended to be demonstrated include basic functions such as grasping and holding an upright object, waving fingers, and opening and closing a fist upon command of the user.

The WBA will be controlled by a glove that will be worn by the user. An effective calibration process will be implemented to ensure that the WBA can be used by a variety of users and track the users fingers accurately. The WBA aims to be able to be used in a variety of environments, including virtual reality (VR), and applications including assisting people with hand or finger disabilities, or providing workers with a safer way to perform tasks.

Scope

The development of the device is at the intersection of many topics including precision sensing, electronic and computer communications, control algorithms, mechanical interfacing, and embedded system power management. It is the expectation that a robust sensing solution will be developed that will interface with a suitable microcontroller (MCU) and servo motors in a system fabricated in-house with resources available at the University of Windsor.

The WBA will entail the unique design of a variety of mechanical parts which may include: sensor fixtures, a control glove, a graphical user interface, a simulation environment, a bionic hand, and a wearable harness. On the electronic side, considerations include a breakout board for the MCU, sensors, and motors, safe and durable wiring, and a power solution. Finally, the team will devise a firmware solution to tie the mechanical and electrical components together to create a functional system.

Benchmarking

Author: Steven Caro

Much research was done on existing biomimetic hand projects through academic papers and hobbyist blogs. Will Cogley's Biomimetic Bionic Hand [2], Mohammadi et al.'s Soft FDM Prosthetic Hand [3], and Youbionic's Double Hand [4] are three very different designs that were analyzed to understand the competitive market.

Will Cogley's design is the most comprehensive solution found on the market. It is a nearly completely 3D printed design, aside from the electronic components and metal screws required for assembly. The design consists of a control glove, which reads the finger position of the user, and a bionic hand which mimics this position. The control glove uses a potentiometer on each knuckle with an additional one at the base of each finger. The potentiometers are spun by the user and the change in voltage indicates the angle at which each knuckle is bent, and the lateral sway of the finger (captured by the potentiometer at the base of the finger). This allowed Cogley to implement four degrees of freedom per finger. On the bionic hand, each knuckle and the lateral position of the finger was individually controlled. This required four servo motors per finger. Cogley used strings attached to the servos in order to actuate the digits of the bionic hand [2].

Cogley's bionic hand offers an incredibly accurate range of motion (ROM), but at the cost of weight and practicality. Cogley admits that 90% of the ROM of the finger can be achieved just by controlling the PIP and MCP joints [2] (see Figure 24). This would require half as many motors and significantly reduce the weight of the hand. Also, he warns of the degradation of potentiometers (PODE) and raises concerns about the accuracy of his control glove decreasing over time [2].

Mohammadi et al.'s Soft FDM Prosthetic Hand was the most human-like hand design, and offered unique attributes. The hand was 3D printed with TPU filament, a flexible material, in order to give the hand a soft feel and potentially better grip than a smooth, rigid design. The digits only have one degree of freedom, meaning the design had five motors. The designers were able to embed the motors and MCU in the back of the hand, which is enclosed during use. Strings were used for digit actuation, but interestingly only for contraction. The relaxation of the fingers relied on the tension of the flexible material to restore itself to its natural position [3]. To its disadvantage, this design ranked eighth out of the ten designs compared in Mohammadi et al.'s grip force comparison [3]. Relying on the tension of the flexible material for digit relaxation raises concerns of the digit losing the ability to relax if the tension degrades over time.

Youbionic's Double Hand cHallenges the idea of conventional biomimetic hand design by having the user control two hand which stem off their own arm. The hands are oversized, which allowed for a servo motor to be attached directly to each finger. The rotation of the motor actuated a linear piston, which controlled the contraction and relaxation of the digit. Each hand was controlled by a single strain gauge worn by the user, on their index and ring fingers. Contracting the control finger would contract all five fingers of the respective bionic hand. The design also considers a forearm mount for the apparatus, using a curved 3D printed mount which sits on top of the user's forearm and is secured with Velcro straps around the arm [4].

The InMoov Hand [5], the Humanoid Robotic Hand [6], and the Robotic Hand [7] designs have been used by many as a base hand model to implement robotic hand projects, with the InMoov hand being far the most prominent. Seeing much success, and it being open-source, makes the InMoov hand an ideal model to start with for the WBA.

A shortcoming of all of these designs is that the fingers do not close completely. The hands are only designed to grab larger objects and cannot close enough to grab small objects. This, among the other noted shortfalls, is a key objective of the WBA.

Deliverables, Design Criteria, and Constraints

Project Deliverables

Author: Gian Favero

The deliverables of our project are as follows:

1. Design a control glove that can detect both MCP and PIP joint rotation in all 5 fingers
2. Program a calibration procedure that yields accurate results seen in WBA and VR usage
3. Develop an easy-to-use graphical user interface application for calibration and usage of the glove in a VR environment
4. Design a robotic hand and forearm mount that can actuate alongside real-time finger movements and hold objects

Sensing Technology

Author: Uygur Tepe

The first type of sensing technology assessed was rotary potentiometers which is a form of resistive sensing. From our initial research, this solution is one of the most popular among hobbyists and researchers. While this solution is relatively inexpensive and straightforward to implement, it has its limitations due to PODE, as noted by Cogley [2]. Ultimately, due to these limitations, the potentiometer was not chosen as the sensing technology solution. A second resistive option was flex strain gauges, which are commonly used in control gloves, but are not ideal for the WBA project due to their restriction of movement and inconsistencies in the data they provide [8].

The second sensing option considered was a CV-based solution. MediaPipe, a landmark detection library based on Google/OpenCV, has proven to be a reliable tool for identifying the precise location of hand and finger joints. To assess the feasibility of this sensing solution for the WBA project, we developed a proof-of-concept Python program that leverages the power of this API.

The final sensing solution considered are analog Hall effect sensors. Through the use of a 3D printed control glove equipped with Hall sensors and magnets at each finger joint, this solution offers several advantages, including affordability, accuracy, and compatibility with low-power MCUs. Unlike potentiometer-based solutions, which are prone to performance degradation and inconsistencies, the Hall effect sensor-based solution is highly reliable over time. Additionally, it overcomes the limitations of CV solutions, such as lighting, distance, and angle restrictions. However, there is still a significant amount of iteration required to refine the glove's design and magnet placement to achieve optimal performance.

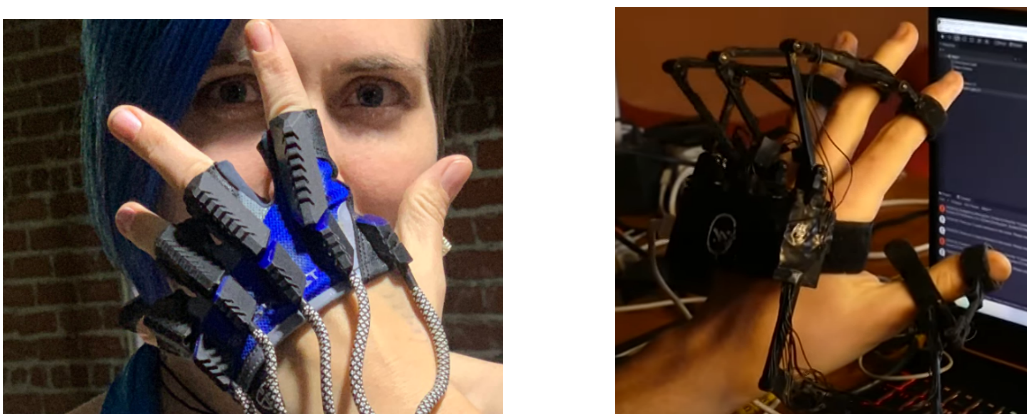
The research done has led to the conclusion that the Hall effect sensor solution is most viable for the WBA project.

Control Glove

Author: Gian Favero

With the key objective of designing a minimalistic and ergonomic solution that senses an analog measurement of a user's PIP and MCP joints, the following designs were considered.

Two main contributors to the body of Hall sensor glove designs come from YouTube channel owners Zak Freedman [9] and NepYope [8], who have each designed a variation suited for computer interaction and robot control, respectively. Both designs capture joint rotation effectively, albeit in very different ways (Figure 1). The Somatic glove designed [9] consists of Hall sensor and magnet fixtures attached to a weightlifters glove. Preferred aspects of this design is the simplicity, sleekness, and ergonomic coupling of the sensors and a wrist mounted MCU. NepYope's design is of an entirely different esthetic and function, but offers two key advantages in that it directly senses both the PIP and MCP joints of each finger, as well as that it provides an analog signal output from its Hall sensors [8].



(a) Somatic AI Glove

(b) NepYope Haptic Glove

Figure 1: Hall sensor control glove alternatives.

Taking advantage of the structure from the Somatic glove and the sensor positions and type from the NepYope glove, a hybrid solution that accomplishes all three objectives was be met.

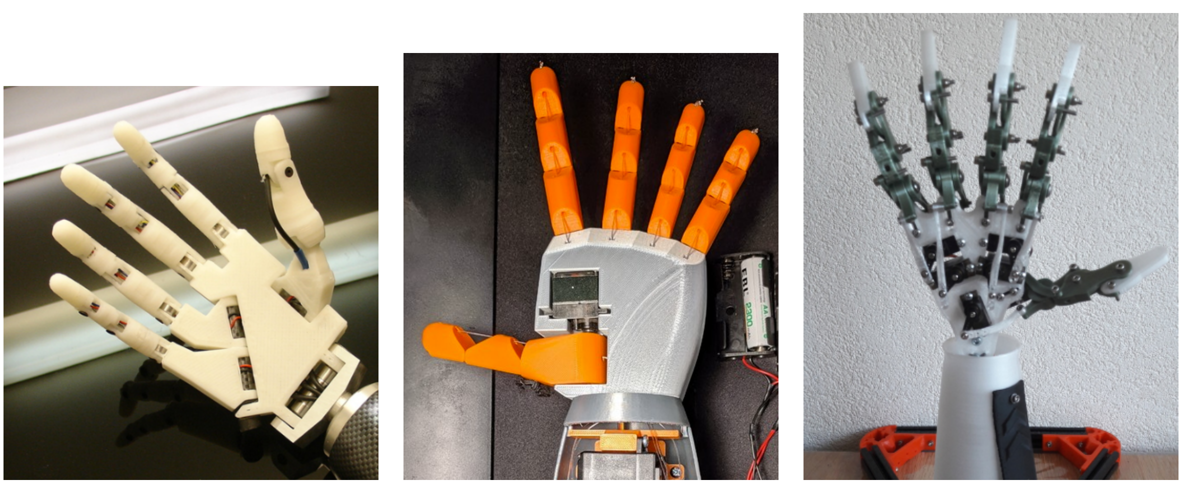
Finger and Hand Design

Author: Steven Caro

Multiple design routes were considered for the robotic hand design. The first being to completely create the 3D models of the hand and fingers from scratch. This would allow for complete control over every specification of the model: from appearance, to desired ROM, to built-in housing for electrical components. The obvious downside to this approach is that it is a very ambitious mechanical design project with many potential points of failure, which could turn into an exorbitant time sink. On the other end of the spectrum laid the second considered design route, which was using a pre-designed open source model. This would be the quickest method, however there is not a pre-existing model which satisfies all the design requirements of the WBA.

Weighing these options led to the optimal decision of remodeling an existing design. This route is significantly less time-consuming than designing a model from scratch, while still enabling the hand to achieve all the design requirements, and with fewer points of failure considering that the base model has already been tested by the original creator. It was expected that modifications would be required to fulfill the deliverables of grabbing and holding objects upright, making the hand mountable to a user, and making the hand as lightweight as possible.

While performing competitive research, it was found that the InMoov hand was used by many creators of 3D printed robotic hand projects. Compared to other open-source hands, namely Tohaker's Humanoid Robotic Hand [6] and Hatsyflatsy's Robotic Hand [7], the InMoov hand offered a much simpler design, and an aesthetically cleaner look. Having a simple design was essential when selecting a base model for this project, since it would be easier to recreate in CAD software. Due to its popularity and simplicity, the InMoov hand was selected to be the base model of the WBA. All three models can be seen in Figure 2.



(a) InMoov (b) Tohaker (c) Hatsyflatsy

Figure 2: Open-source hand designs

Microcontroller

Author: Uygur Tepe

Three MCUs were considered as the main device for interfacing between the sensors and servo motors: the Raspberry Pi 4, the BeagleBone Black, and the Raspberry Pi Pico. Processing power, power consumption, size, and the type of sensing solution being used were some of the many factors in consideration for the MCU choice.

The Raspberry Pi 4 is the most powerful of the options, making it most suitable for CV applications. It can run a variety of CV libraries such as OpenCV and MediaPipe and is compatible with a variety of options such as the Raspberry Pi camera module, a USB webcams and a GoPro. A drawback of the Raspberry Pi 4 is that it has the largest footprint and needs cooling measures in place to keep it from thermal throttling. CV applications are intensive processes needing maximal available performance to run smoothly [10], thus adding a cooling solution to the board is a necessity.

The second board considered was the BeagleBone Black, which a common option for embedded vision and system applications. It has many GPIO ports (65 digital expansion and 7 ADC) easily capable of interfacing with the sensing system and servo motors. Furthermore, the BeagleBone has a relatively powerful chip (1GHz ARM Cortex-A8) which can handle certain OpenCV applications and other real-time programs [11]. The most notable downside to the BeagleBone is the limited onboard memory (512 MB) which is not sufficient for the more complex image processing tasks. External memory could be added, but that would increase the size and cost of the board. It is overpowered for the Hall sensor solution and under powered for the CV solution, but nonetheless a robust option overall.

The final board considered was the Raspberry Pi Pico (Pico), this solution is very compact and would only be considered for the Hall sensor solution due to the lack of CV processing capabilities. For its size, the Pico has high processing power containing a 133 MHz Arm Cortex-M0+ processor which is enough for real-time finger tracking. Due to the compact nature and low power consumption of the Pico, it would easier to integrate into the control glove design without the need for a large battery to power the device. The Pico also contains a wide range of GPIO ports including, analog, digital and PWM ports that satisfy our requirements for interfacing with the Hall sensor and the servo motors. Hence, it is the preferred option for the Hall sensor solution. Given that the Hall sensor solution was chosen, the Pico was selected as the MCU for the WBA project.

Design Methodology

Control Glove

Author: Gian Favero

The design of the control glove centered around two prominent topics: mechanical dynamics, electrical signal flow.

At the focal point of designing a finger tracking glove is a hand. The hand is a complex system of bones, tendons, and muscles that work together to create a wide range of motion. The design of the glove had to take into account the natural range of motion of the hand, and the placement of the sensors had to be carefully considered ensuring that the sensors would not interfere with any natural movement.

A custom design akin to an exoskeleton was created. Key components include magnet-housing fixtures, Hall sensor-housing rods, a fastening system to anchor the glove to a user's hand, and finger straps. Pivot points were constructed at each PIP and MCP joint to allow a magnet to rotate around a Hall sensor and thus detect a change in position based on the magnetic field strength. The design of the glove was iterated upon multiple times to ensure that the glove was comfortable to wear, and that the sensors were placed in the optimal position to detect finger movements at the PIP and MCP joints. The glove was designed to be 3D printed using TPU filament to allow for flexibility and comfort.

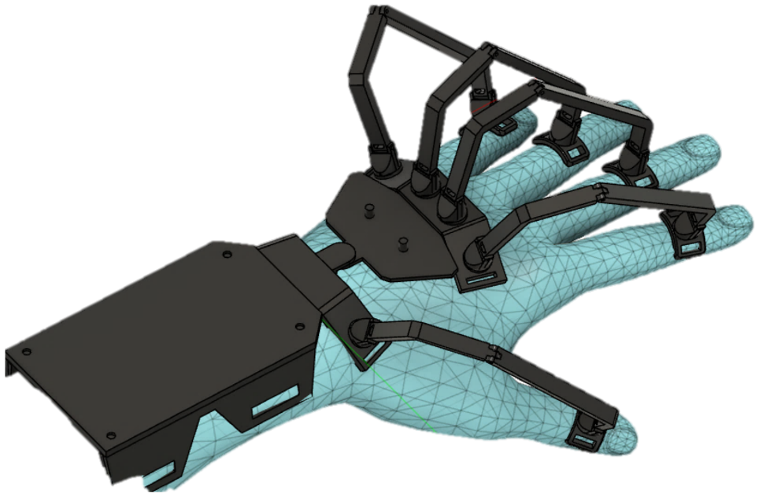


Figure 3: Control glove rendering

The design is modular in the sense that each finger has a set of repeating parts that are tied together on the palm component of the skeleton. Each finger has a PIP rod and an MCP rod that are fastened together with a hinge that allows extension and retraction based on finger size and rotation. A small slot was built into these rods for placement of Hall sensors. Each finger has a magnet housed in a pop-in-place fixture that is attached to either a user's PIP joint via a finger strap, or MCP joint via the palm component of the glove. These fixtures are an essential component to the glove as they allow non-restrictive, full, finger movement while maintaining a constant distance between the magnet and Hall sensor.

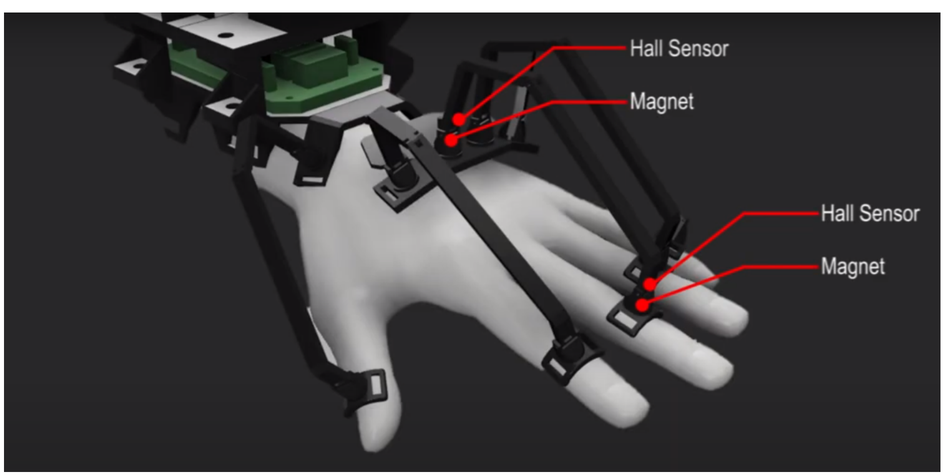
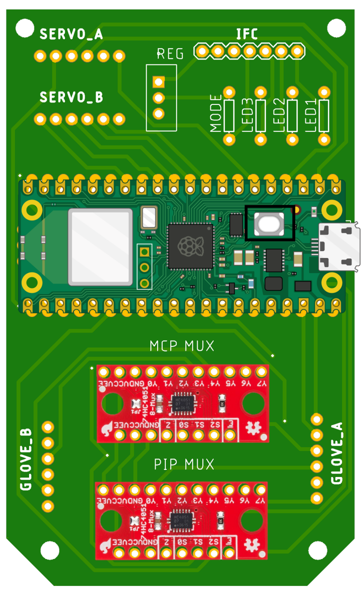
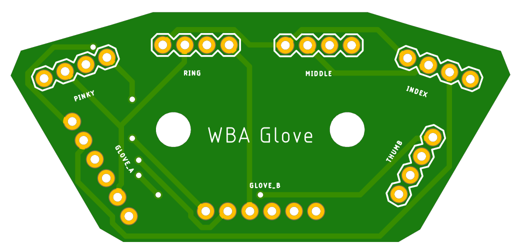


Figure 4: Control glove annotation

Two PCBs were designed for the control glove using EAGLE, one acting as the main control unit for the device, and the other to serve as a power and data signal distribution board. Key components worked into the PCBs include 3×8 multiplexers (74HC4051) that expand the onboard ADC channels of the RP2040 MCU from 2 to 16. In addition, regulator was placed on the board to allow components to run at the source voltage or at 3.3 V if needed. The PCBs were designed to fit into the WBAs housing, hence the precisely shaped form factors. Full schematics and PCB layouts can be found in the Appendix.



(a) WBA ECU



(b) Glove ECU

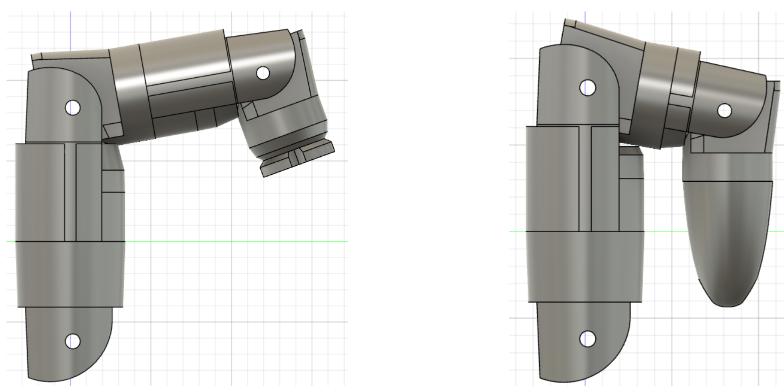
Figure 5: Control glove PCBs

Harnesses were designed to distribute power and collect the data signals from the glove, based on the simplified circuit design found in the Appendix (Figure 23). Molex connectors were used to connect the glove to the harnesses, allowing for easy removal of the glove. The breakout PCB, deemed the Glove ECU, was housed on the top of the palm. A second set of harnesses were designed to connect the Glove ECU to the main ECU, deemed the WBA ECU, which was housed in a forearm mount.

Robotic Hand

Author: Steven Caro

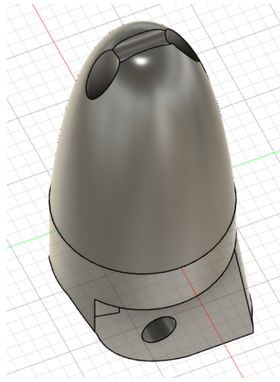
Many mechanical design challenges were present in the creation of the WBA's robotic hand. As mentioned above, the InMoov hand was used as a base model for this project. A sample finger was printed and assembled for initial model evaluation and multiple issues became apparent. Each portion of the finger came in two separate pieces, which had to be glued together and the fingers themselves could only bend 80° per joint. These were problems because gluing the portions of the finger together created a much weaker bond than having the portions combined and printed as one, and the 80° of ROM would not be enough for the fingers to close around an object. To address these issues, the fingers were remodeled using Autodesk Fusion 360. Seen in Figure 6 below, each of the three sections of the finger were turned into single pieces, and the ROM was increased to 100° per joint.



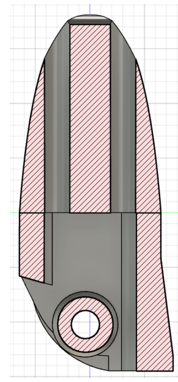
(a) Original ROM (b) Improved ROM

Figure 6: Finger ROM

A specially designed fingertip was created with an internal loop, shown in Figure 7. This allowed for fish line to loop through the fingertip before routing through the rest of the apparatus. This greatly reduced the slip of line when it was operating the finger.



(a) New Fingertip Design

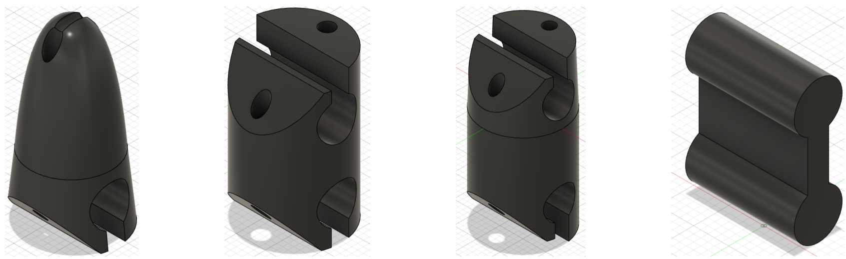


(b) Cross-Section View

Figure 7: Fingertip design

An apparatus was set up in order to test the operation of the finger with one motor. The hinges of the finger were of the axle and rod family. Fish line was passed in above each rod, around the fingertip, through the fingertip loop, then back through, under each rod. The line was then tied to a pulley provided by InMoov, which attached to a servo motor. Through this testing, a tremendous problem was discovered. At the extremes of the finger ROM, there was enough tension in the line to keep the finger rigid, however there was little tension in between the extremes. The finger had no rigidity for most of the travel of the finger. This was due to the loop path shortening when the finger contracted, causing slack in the fish line.

Multiple potential solutions were considered for this problem, one being a spring tensioner fixed somewhere to the test apparatus to constantly remove any slack from the line, however this solution was not favoured due to the team's inexperience with this method, which could have caused unforeseen hardships further in the design process. Inspiration for the pursued solution came from Bob Houston's Flexible Joints for InMoov's Fingers [12]. Houston had the idea to print hinges in a flexible material (TPU) such that when flexed, the hinge tries to restore its natural position. The flexible hinges would replace all of the axle and rod hinges on the fingers. This solves the problem of tension, as there is a constant force against the fish line to keep it taut. Houston had a great idea, however his physical implementation was poor. The hinges did not fit into the fingers. Using the same idea, the fingers of the WBA were redesigned to incorporate flexible hinges and the hinges themselves were also designed, Figure 8. The hand model also had to be modified to accommodate the flexible hinge design.



(a) Tip (b) Middle (c) Base (d) Hinge

Figure 8: Finger Redesign for New Flexible Hinge

Below is a rendering of the final design of the WBA robotic hand.

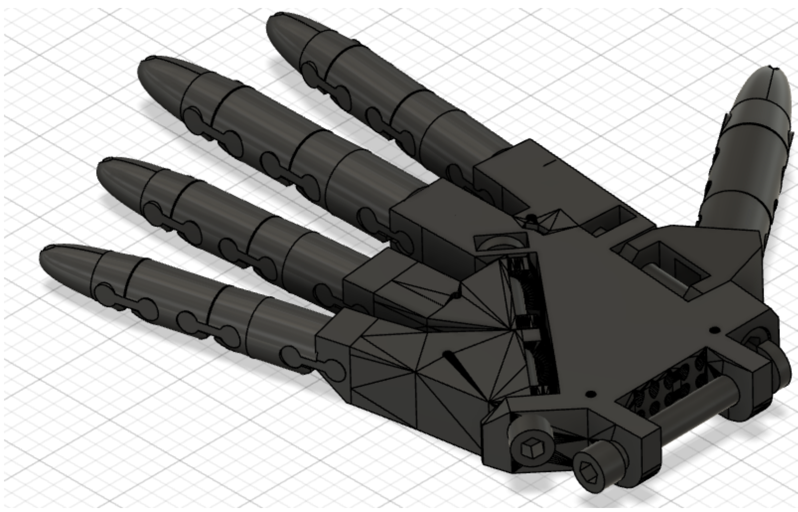


Figure 9: Robotic hand rendering

Forearm Mount

Author: Steven Caro

A main deliverable of the project was for the WBA to be wearable. A unique mounting solution was required to make this possible, as no project has ever attempted to mount the InMoov hand to the user's arm. An initial sketch of the solution was made (Figure 10). The initial plan was to allow the hand fixture to be able to slide in and out of the rest of the mount. Later, the team decided that the forearm mount was the best place to house the main ECU of the WBA, which sacrificed the sliding feature. A raised platform was created to hold the hand-mounting system above the ECU. The forearm mount and platform are seen in Figure 11.

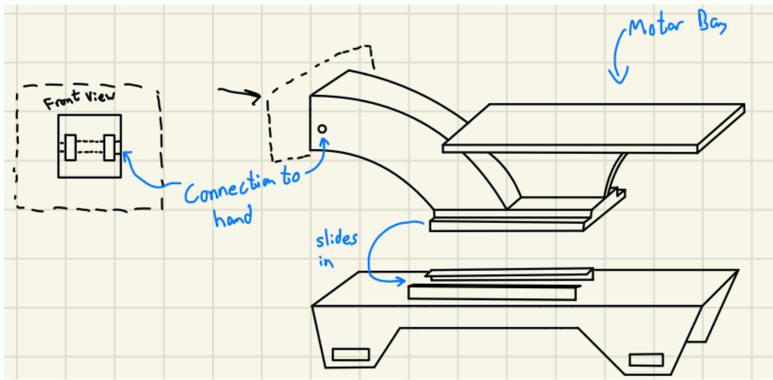
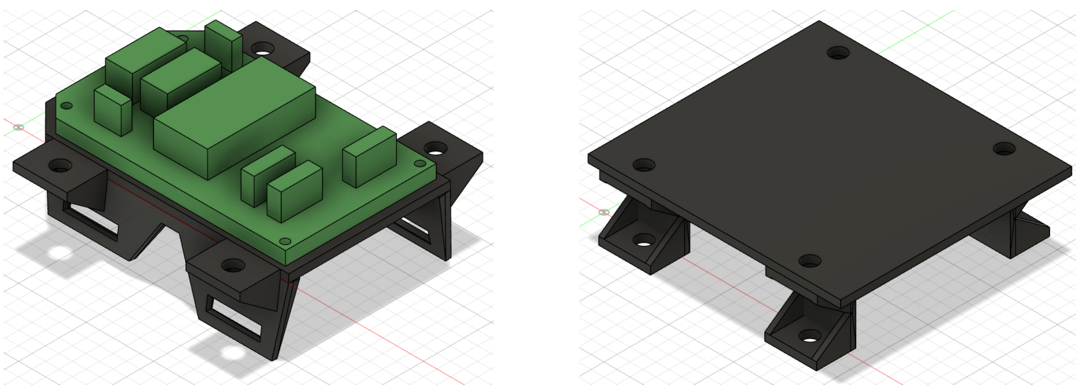


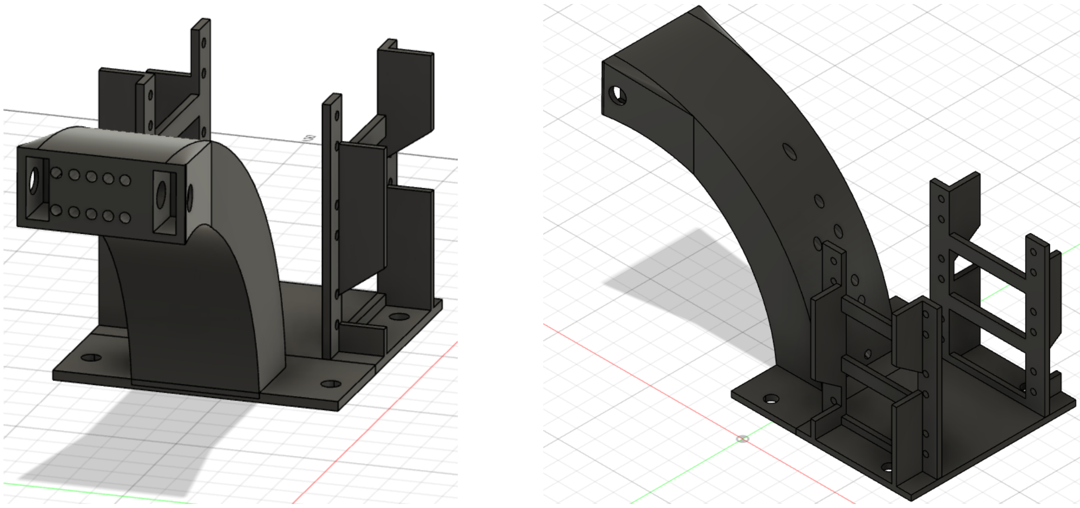
Figure 10: Forearm mount sketch



(a) (b)

Figure 11: Forearm mount (a) and raised platform (b)

The hand already had a hinge and bolt built into the design in order to connect to a wrist, which was a perfect method of connecting to the mounting arm. The motor bay was not designed as shown in the initial sketch, but in a vertical fashion such that there were two columns of motors, in order to save space. In the robotic hand, there is a network of tunnels leading to each finger from the wrist area. A similar network was created in the mounting arm to align with the holes in the wrist area, which led down the arm to line up with each servo (Figure 12).



(a) (b)

Figure 12: Mounting arm

Below is a rendering of the final design of the full forearm mount with the robotic hand attached.

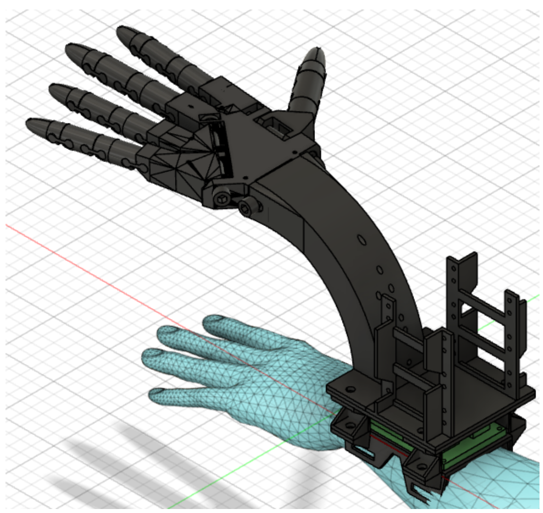


Figure 13: Forearm mount rendering

Microcontroller Software

Author: Uygur Tepe

The software developed for the MCU was an important portion of the design of the solution. The first step in developing the software for the Pico MCU was to identify the project requirements. The software needed to handle reading the Hall sensor outputs, processing the readings through a robust calibration model, and outputting the processed readings as angles to control the servo motors and the GUI.

In order to streamline the amount of modules needed by the control glove, it was decided to follow an object-oriented approach to ensure easier implementation of the required functionality. A "glove" class was created, which encapsulated all the necessary functionality for the control glove, including reading sensor values, calibration, and processing angles. Since the object-oriented paradigm was used for the control glove, MicroPython was the programming language selected, which is a variant of Python 3 optimized for MCUs.

A crucial aspect of the software development process was to ensure robust and accurate calibration of the Hall sensors. A GUI was planned to be integrated to facilitate real-time calibration. This GUI would enable users to observe the calibration results immediately through a virtual hand representation. The goal in mind was to dynamically model the joint rotation and store them in a "relationships.json" file, which was stored on the Pico's non-volatile flash memory. This ensured that the calibration file remained intact even if the Pico was powered off, allowing for one-time calibration, which could be used for all future runs of the program by the specific user.

In order to meet all requirements for the functionality of the control glove, there were four main states that the control glove would be in: calibration, VR, WBA, and idle. The calibration state would be entered when the user pressed the calibration button on the GUI. In this state, the glove would read the sensor values and calculate the calibration values. The calibration values would then be stored in the "relationships.json" file. The VR, or WBA state would be entered when the user pressed the appropriate button on the GUI. In this state, the glove would read the sensor values, process the angles, and output the angles to the virtual hand model or the servo motors respectively. The idle state would be entered when the user pressed the stop button on the GUI. In this state, the glove would not read any sensor values and would not output any angles to the GUI or the servo motors. To model this state machine, a state machine diagram was created, as shown in Figure 14.

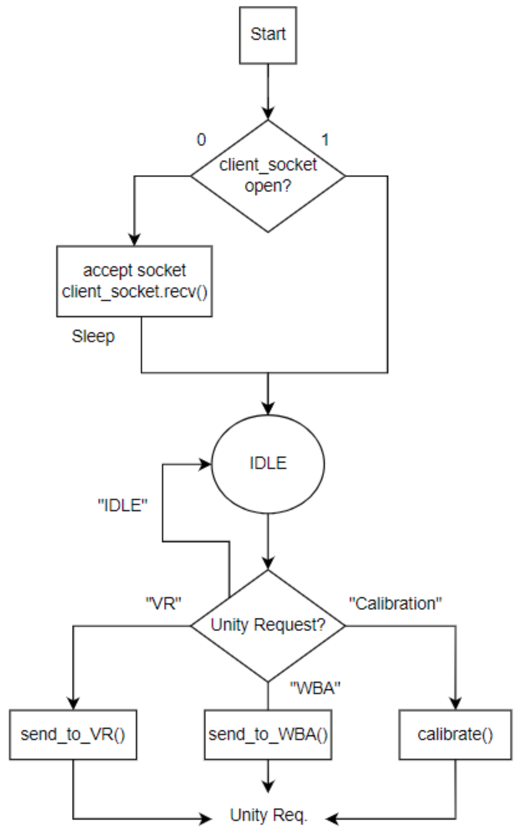


Figure 14: State machine diagram

To ensure the step-by-step process outlined worked correctly, the following procedures were employed to test the software. Firstly, unit tests were conducted to verify each individual method, such as reading sensor values, calibration, calculating processed angles of the glove class worked as intended. Then integration tests were performed to determine the interactions between the different methods of the software. Finally, a simple moving average (SMA) algorithm used to reduce noise from the Pico's 12-bit ADC was tested with different window durations. The accuracy of the algorithm was assessed against known calibration values to validate its effectiveness in reducing noise while preserving the angle precision. All tests were performed in a separate testing folder to ensure the main program was not affected.

Throughout the software development process, iterative testing, debugging and refactoring were performed to achieve a reliable and user-friendly software solution for the Pico MCU control glove. Any identified issues were promptly addressed and resolved to ensure the final design met the project's requirement and our deliverables.

Physical Implementation and Simulation

Control Glove

Author: Gian Favero

The control glove exoskeleton and forearm mount were 3D printed using TPU filament and were fastened to a user with hand-sewn elastic straps. Hinges were formed with paperclips to allow smooth and non-restrictive finger motion. Hall sensor probes were reinforced with hot glue to prevent bending and breaking during usage.

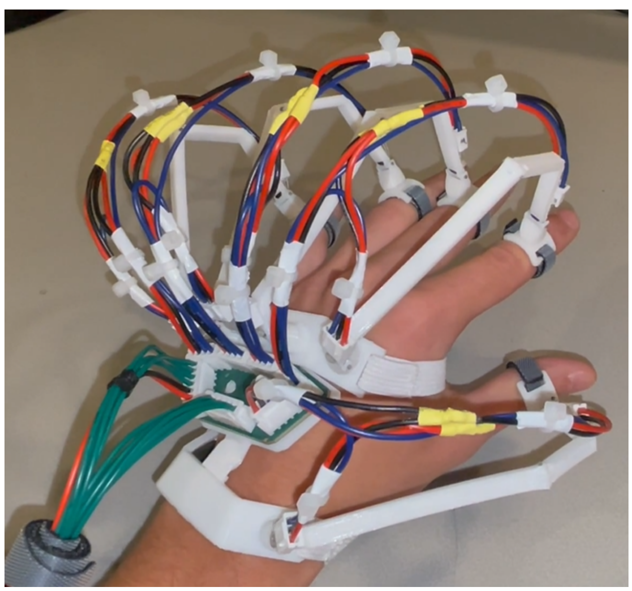


Figure 15: Control glove

The glove was largely designed for snap-in-place assembly, leveraging the flexibility of TPU. The magnet fixtures form a ball and socket joint with the exoskeleton and finger straps, allowing 4 degrees of rotation for each joint. Finger straps were designed to be adjustable to allow for a variety of finger sizes. The straps were designed to be fastened with Velcro, allowing for easy removal of the glove.

The PCBs were fabricated at two locations: the University of Windsor and PCBWay. Components were hand soldered and tested for continuity and shorts. The PCBs were designed to be mounted on the glove and forearm mount with TPU inserts. Robust and reliable, the PCBs served as a point of stability for the control glove, and later the motor bay. The electrical harnesses were fitted with Molex crimps and connectors with colour-coded wiring, with zip ties being used to allowed for strain relief and cable management.

The glove was designed to be worn on the user's left hand, with the glove harnesses and Glove ECU being worn on the user's left forearm. Upon implementation, the glove was found to be comfortable and non-restrictive and able to be worn for extended periods of time without discomfort. The glove was also found to be robust and reliable, with solid accommodations being met for the varying hand and finger sizes of the team members.

Robotic Hand and Forearm Mount

Author: Steven Caro

The bulk of the robotic hand was 3D printed using an inflexible, versatile plastic (PLA), while the flexible hinges were printed in flexible TPU. The mounting arm, motor bay, and raised platform were also printed with PLA for strength, while the forearm mount was printed in TPU for comfort, so it could flex around the user's arm. A flexible Velcro strap was made to strap the user's arm to the mount.

The servo motors were fastened to the motor bay via brass wood screws. Two runs of fish line were installed per finger. One in the track above and one below each hinge. The lines were knotted at the fingertip and tied to the servo pully. Each line had to be tensioned to the correct amount. The tension was held by wrapping the end of the line around a screw in the pully, then tightening the screw.

Due to the high center of gravity of the WBA, the apparatus had a tendency to slip left and right on the user's arm. To eliminate this, the team discovered that sliding a medical brace on the arm of the user prior to strapping the WBA on would completely stop any slippage. The brace served two purposes. For one, the brace compressed the user's skin, reducing skin slippage. The brace also increased the friction on the interface of the forearm mount, reducing slippage on the interface.

GUI and Simulation Model

Author: Uygur Tepe

The physical simulation model was implemented and developed using the Unity game engine. This streamlined the GUI and simulation model as they were able to be developed in parallel. The GUI was developed using the Unity UI system, which is a collection of graphical user interface features that are used to create the interface for the user to interact with. Using these GUI features, buttons and text were added to the GUI to be able to control the different states of the control glove itself. The GUI was also used to display the virtual hand model (created by Oculus [13]). The default idle mode of the GUI is seen in Figure 16. The virtual hand model was used to display the angles of the control glove as a simulation model to determine if the processed angles were accurate to what the user was doing.

Figure 16: Unity GUI in idle mode

In addition to having the virtual hand mimicking the users hand movements, it was also used as an aid when walking users through the calibration process. When the user pressed the calibration button on the GUI, the glove would go into calibration mode and the virtual hand would display what motion the user must perform during that specific calibration step. This was done to ensure that the user was performing the correct motion during the calibration process.

Unity development is done in the C# language, and there are two key functions which execute when a Unity program is started. The first is the start method and the second is the update method, whenever a Unity program runs first the start method is run, then at every frame the update method is called. Leveraging these two methods the main logic of the GUI was implemented. The start method was used to initialize the socket connection between the GUI and the glove MCU. The update method was used to update the GUI based on the current mode of the glove. The update method was also used to update the virtual hand model based on the processed angles of the glove.

The software interfacing between the GUI and the control glove MCU was done using TCP/IP communication. The Pico has built in WiFi capabilities, so TCP/IP communication was a natural choice. The socket library in MicroPython was used to create a socket server on the Pico which would listen for incoming connections from the GUI. Once a connection was established, the Pico would wait until a button was pressed in the GUI. When a button was pressed in the GUI, Unity would send a message to the Pico with the mode that the glove should be in.

The Unity controller class (TcpClient.cs) was implemented using threads. One thread would be used for sending messages to the Pico, the other thread was opened to continuously listen for incoming requests. This was done to ensure that the GUI would not freeze while waiting for a response from the Pico.

The development of the physical simulation model was a successful process. The use of the

Unity game engine streamlined the development process and the use of TCP/IP communication and threads ensured that the model was reliable and efficient.

Experimental Methods and Model Validation

Control Glove

Author: Gian Favero

Much experimentation was needed to derive a calibration algorithm that would accurately map the Hall sensor readings to the finger angles. First, it was to be determined whether the Hall sensor readings were linear or non-linear. This was done by plotting the Hall sensor readings against a set of finger angles derived from an experimental setup similar to that of Figure 25. The results can be seen in Figure 17. Note that the Hall sensor readings are in millivolts.

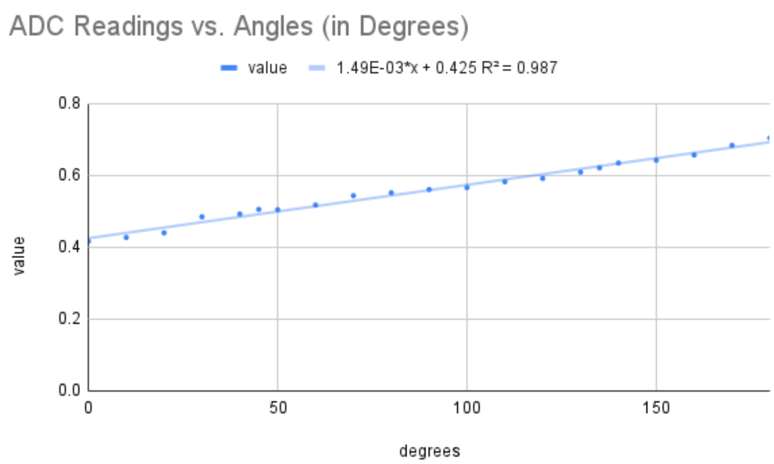


Figure 17: Hall sensor readings vs finger angles

The conclusion was that the Hall sensor readings were linear based on the correlation coefficient being very close to 1 with a linear regression. From there, it was determined that a linear fitting of the Hall sensor readings could be done over a full range-of-motion (ROM) of 0 to 90 degrees for PIP and MCP joints.

To isolate the PIP and MCP joints, a calibration routine was devised to hold one of the joints at a fixed angle while its counterpart was moved through its full ROM, resulting in two linear mapping models per finger. Validation of the calibration algorithm was done via an eye test, where a user would move their fingers and observe the behavior of the virtual hand in the GUI.

The initial algorithm had mediocre success. It was noticed that an interdependency existed between the PIP and MCP joints, meaning the models were not strictly linear, but actually a combination of two inputs - MCP rotation and PIP rotation. Two compensate for this, some further experimentation was done to determine the best way to combine the two inputs.

It was found that the MCP joint was most independent of the two joints being sensed. Thus, it was used as the primary input to the model and assumed to be independent of PIP movement. The PIP joint rotation was then modelled at multiple fixed MCP angles, and the results were averaged to create a single model for each PIP joint. This "regional" calibration algorithm resulted in much more accurate performance indicated by the virtual hand in the GUI.

Robotic Hand

Author: Steven Caro

Servo motors are able to rotate to a precise location based on the inputted duty cycle. For the application of this project, it was practical to create a conversion between an angle and its corresponding duty cycle. The duty cycle input ranged from 2000 to 8000 ns, where 2000 ns corresponded to 0° and 8000 ns corresponded to 180°. Therefore, the following formula was devised, which enabled the team to input an angle to the motors:

$$dutyCycle = 6000 \times (angle/180) + 2000$$

Since the robotic hand had one motor per finger, but the control glove recorded the angle of both the MCP and PIP joints of the finger, a method of combining the two recorded angles into one average angle of the finger. The algorithm was finalized after iterations of altering the weights between the MCP and PIP angles until the average angle closely approximated the angle of finger as a whole. The team determined that taking a weighted average of 35% MCP angle and 65% PIP angle was the correct balance.

Software and Simulation Model

Author: Uygur Tepe

As stated in the control glove subsection, a big portion of testing went to ensuring the accuracy of the calibration algorithm. The calibration algorithm was tested by having a user perform a set of motions and observing the virtual hand in the GUI. As seen from Figure 18, when the GUI is in VR mode, it mimics the users hand movements accurately when properly calibrated.

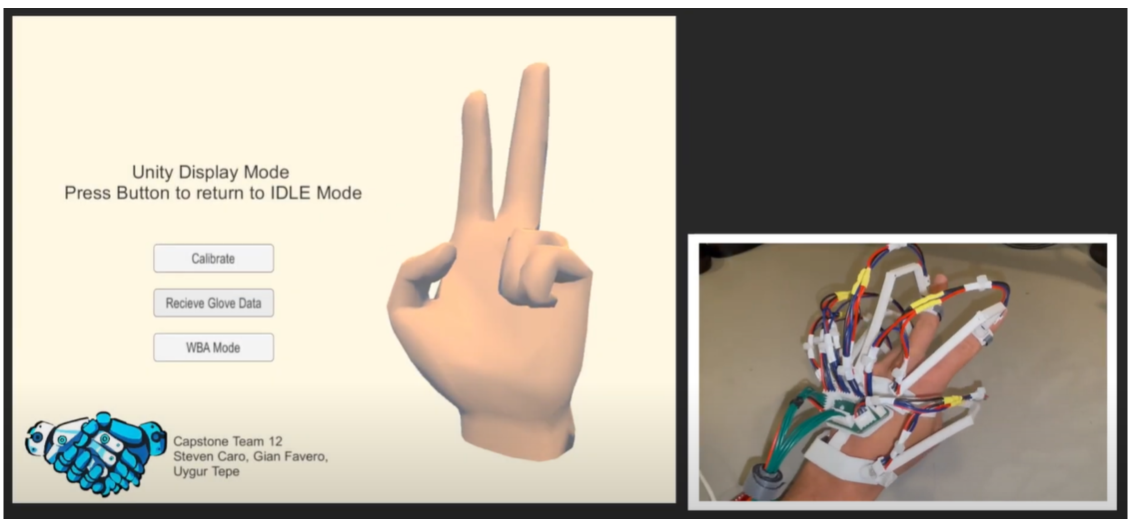


Figure 18: Unity GUI in VR mode

Based on the results seen from the GUI in VR mode, it was estimated that the glove when properly calibrated can track the users hand movements with reasonable accuracy. However, a more accurate testing model such as the one seen in Figure 25 in the Appendix would be needed to determine the true accuracy of the glove.

Design Specifications and Evaluation Matrix

Author: Gian Favero

A summary of design specifications can be seen in Table 1.

Table 1: Design Specifications

Specification	**Value**
Dimensions	37x14x16 cm
Weight	780 g
Number of Motors	5
Mounting Style	Velcro straps
Power Input	Wired, No Battery
Operating Voltage	5 V
Power Consumption	12 W Max
Latency	2 ms
Connectivity	2.4 GHz WLAN
Interface	GUI
Grip Strength	11 N/cm per finger
PIP Sensing	Yes
MCP Sensing	Yes

The table provides a comprehensive overview of the design specifications for a specific product. The product has dimensions of 37x14x16 cm and weighs 780 g. It is equipped with five motors and can be mounted using Velcro straps. The power input is through a wired connection with no battery requirement, operating at a voltage of 5 V and a maximum power consumption of 12 W. The product offers low latency at 2 ms and connects via 2.4 GHz WLAN for seamless communication. Users can interact with the product through a graphical user interface (GUI). Notably, each finger has a grip strength of 11 N/cm. Additionally, the product incorporates PIP sensing and MCP sensing capabilities, enhancing its usability and functionality. These specifications make it suitable for various applications that demand precision, control, and real-time feedback.

An evaluation matrix based on the outlined deliverables is seen in Table 2.

Table 2: Evaluation Matrix

Criteria	**Score (1-5)**
Hold Upright Objects	4
Lightweight	3
Low Latency	5
Wearable	5
Calibration Accuracy	4
Overall	4.2

The table represents an evaluation matrix that assesses various criteria for a certain product. The criteria and their corresponding scores from 1 to 5 are as follows: "Hold Upright Objects" received a score of 4, indicating its good performance in this aspect. "Lightweight" achieved a score of 3, suggesting it meets the requirement but has some room for improvement. "Low Latency" and "Wearable" both obtained the highest score of 5, indicating excellent performance in these areas. "Calibration Accuracy" received a score of 4, showing that it is highly accurate but may have minor room for enhancement. Overall, the product scored an average of 4.2, indicating a well-rounded and promising performance across the evaluated criteria.

Budget

Author: Steven Caro

In total, the team has spent a total $401.85 CAD for research and development, leaving a remaining budget of $498.15 CAD. The total expenses relative to our initial budget can be seen in Table 3.

Table 3: R&D Budget

Category	**Allocation**	**Spent**	**Remaining**
Motors	$300.00	$91.28	$208.72
MCU	$100.00	$68.80	$31.20
Sensors	$200.00	$146.43	$53.57
Filament	$100.00	$85.34	$14.66
Other	$200.00	$10.00	$190.00
Total		$401.85	
Remaining			$498.15

In summary, 44% the available funds were spent in the development of the WBA project.

Tables 4 and 5 show the cost breakdown per unit for the WBA control glove and the fully assembled WBA, respectively.

Table 4: WBA Control Glove Cost Breakdown

Item	**Quantity**	**Cost**
Hall Sensor	10	$18.20
3x8 MUX	2	$9.24
3.3V Regulator	1	$1.13
40 POS Socket	2	$2.56
Molex Connectors	22	$3.18
TPU Filament	0.061 kg	$3.25
RPi Pico W	1	$8.45
5 mm Magnet	10	$2.00
18 AWG Wire	7.45 m	$5.93
WBA PCB v2	1	$0.40
Glove PCB v1.1	1	$0.40
Total		$54.75

Table 5: Fully Assembled WBA Cost Breakdown

Item	**Quantity**	**Cost**
Servo Motors	5	$40.89
Hall Sensor	10	$18.20
3x8 MUX	2	$9.24
3.3V Regulator	1	$1.13
40 POS Socket	2	$2.56
Fish Line	0.37 m	$0.32
Molex Connectors	38	$3.18
PLA Filament	0.419 kg	$13.40
TPU Filament	0.061 kg	$3.25
RPi Pico W	1	$8.45
5 mm Magnet	10	$2.00
18 AWG Wire	8.78 m	$6.99
WBA PCB v2	1	$0.40
Glove PCB v1.1	2	$0.80
Total		$110.83

Conclusion

Author: Uygur Tepe

Each component of the WBA was successful in its role in achieving its deliverables and meeting its design specifications.

The control glove was successful in that it met its deliverable to measure a user's PIP and MCP joints with a reasonable degree of accuracy. This was confirmed with simulation testing using the Unity GUI and simulation model. The control glove was also successful in that it met its deliverable to be able to control a robotic hand, as evidenced by successful implementation of the full WBA assembly.

A GUI based calibration algorithm and device interface was successfully implemented using a Unity application. The GUI effectively communicated steps needed for calibration to users and seamlessly integrated the various operating modes of the WBA. Further, it provided a real-time VR environment that had a multitude of uses including validation, testing, and demonstration.

Finally, the full WBA assembly successfully met its targets in being wearable, being able to hold and grasp objects, and mimic a user's hand movements.

Novel contributions of this project include the use of Hall sensors and magnets to sense the position of finger joints in a control glove. In addition, pairing a five finger-sensing control glove, that uses Hall sensors, with a wearable mounted appendage is a novel accomplishment. Future works involve decreasing latency further, increasing the degrees of freedom being tracked on each finger, allowing for multiple mounting options for the appendage, finding the true angle accuracy of the control glove, and implementing haptic feedback from the robotic hand back to the control glove.

Overall, the results of this project have demonstrated the potential of the WBA to be a valuable tool for a variety of applications in both VR- and real-world settings.

References

[1] Our hands at work! [Online]. Available: https://www.canada.ca/en/employment-social-development/services/health-safety/reports/hands.html

[2] Will Cogley, "Fully 3d printable biomimetic mechatronic hand progress and 3d renders." [Online]. Available: https://www.youtube.com/watch?v=uESxUEB-v3Q

[3] A. Mohammadi, J. Lavranos, H. Zhou, R. Mutlu, G. Alici, Y. Tan, P. Choong, and D. Oetomo, "A practical 3d-printed soft robotic prosthetic hand with multi-articulating capabilities," vol. 15, no. 5, p. e0232766. [Online]. Available: https://dx.plos.org/10.1371/journal.pone.0232766

[4] Mashable, "This 3d-printed wearable is giving you an extra hand... literally." [Online]. Available: https://www.youtube.com/watch?v=8hRuS69-j9M

[5] Gael Langevin. InMoov. [Online]. Available: https://inmoov.fr/

[6] Tohaker, "Humanoid Robotic Hand." [Online]. Available: https://www.thingiverse.com/thing:3105948

[7] Hatsyflatsy, "Robotic hand servo powered." [Online]. Available: https://www.thingiverse.com/thing:2171811

[8] NepYope, "I built VR haptic gloves to control robots." [Online]. Available: https://www.youtube.com/watch?v=iPtgvh6fNdQ

[9] Zack Freedman, "AI data glove: Somatic." [Online]. Available: https://www.youtube.com/watch?v=6raRftH9yxM

[10] D. Hertz. How to keep your raspberry pi 4 from overheating. [Online]. Available: https://maker.pro/raspberry-pi/tutorial/how-to-prevent-your-raspberry-pi-4-from-overheating

[11] BeagleBone black. [Online]. Available: https://beagleboard.org/black

[12] Bob Houston, “Flexible joints for InMoov’s fingers.” [Online]. Available: https://www.thingiverse.com/thing:2379548

[13] Oculus. Oculus Hand Models. [Online]. Available: https://developer.oculus.com/downloads/package/oculus-hand-models/

Appendix

YouTube Video: `https://www.youtube.com/watch?v=UIM3DzoZ4W8`

Code

The GitHub repository for the Pico code can be found here:

`https://github.com/utepe/Capstone`

The GitHub repository for the Unity GUI and Simulation model can be found here:

`https://github.com/utepe/UnityHandModel`

PCB Schematics

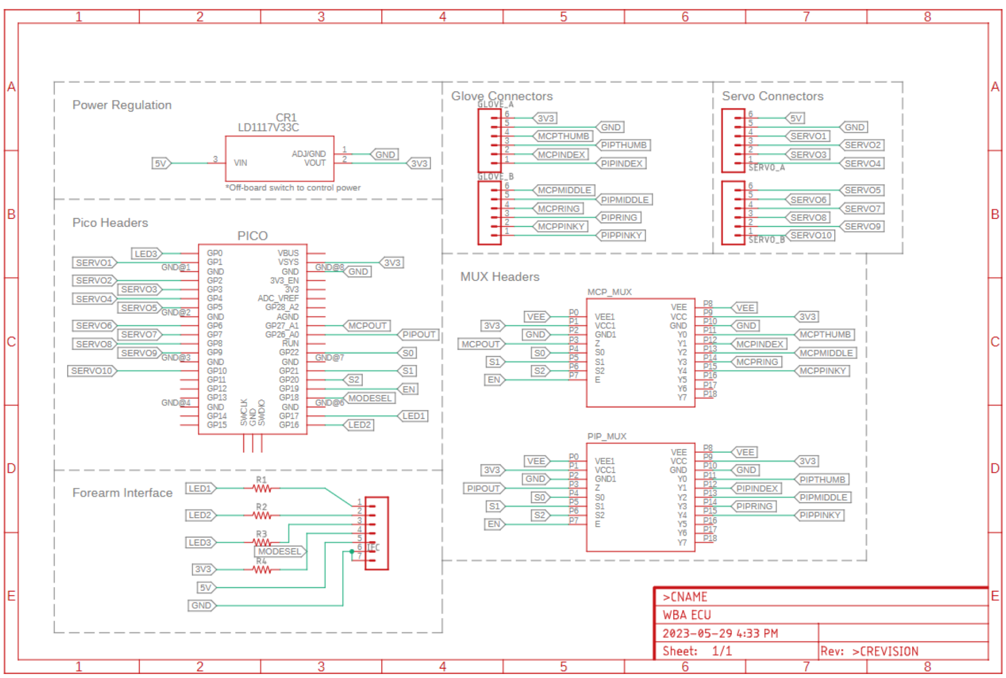


Figure 19: WBA ECU Schematic

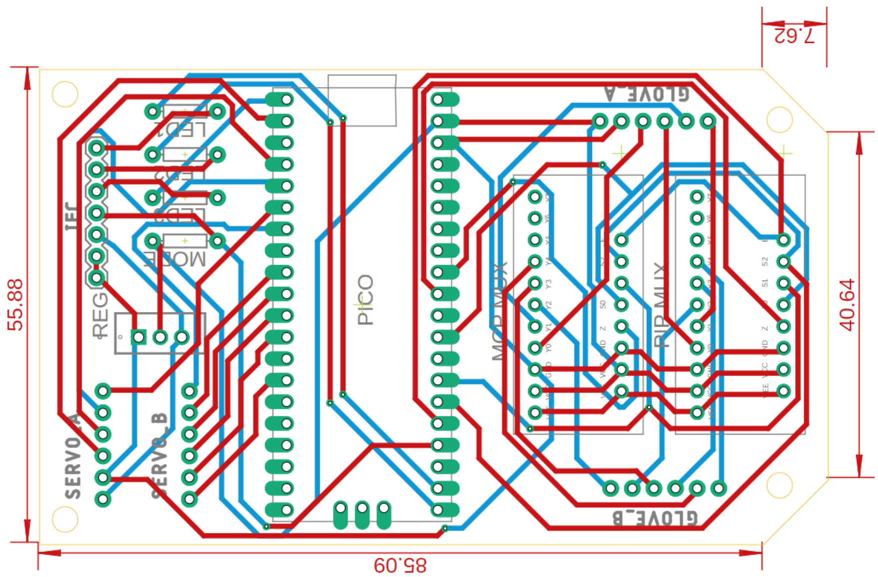


Figure 20: WBA ECU Layout

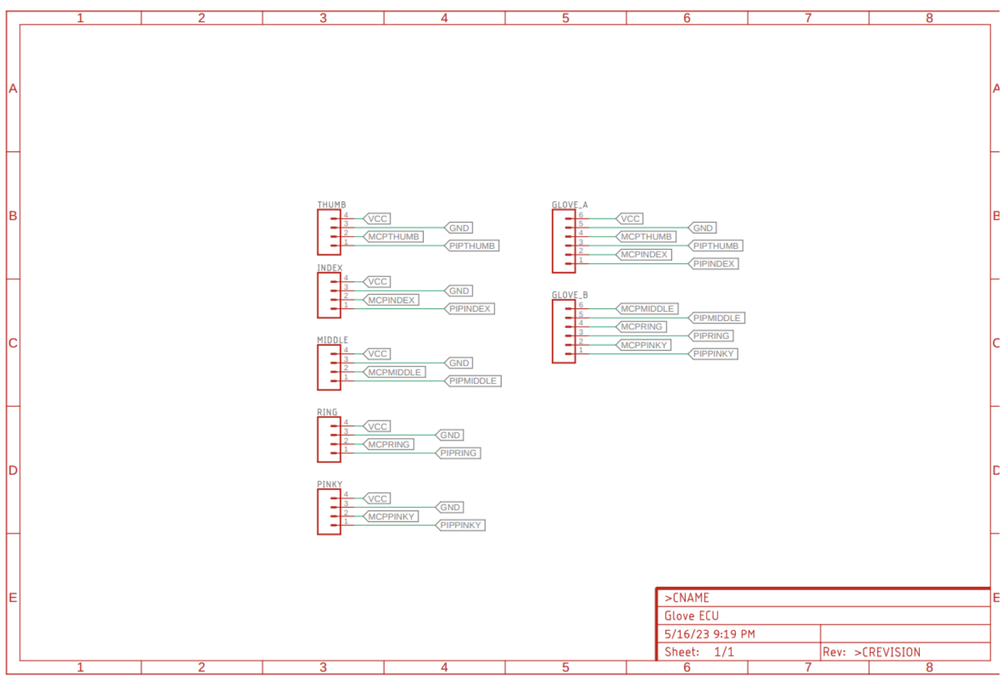


Figure 21: Glove ECU Schematic

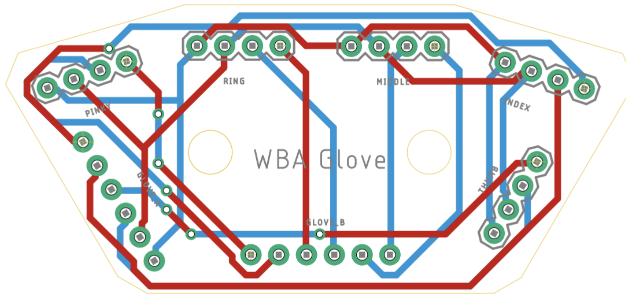


Figure 22: Glove ECU Layout

Intermediate Designs

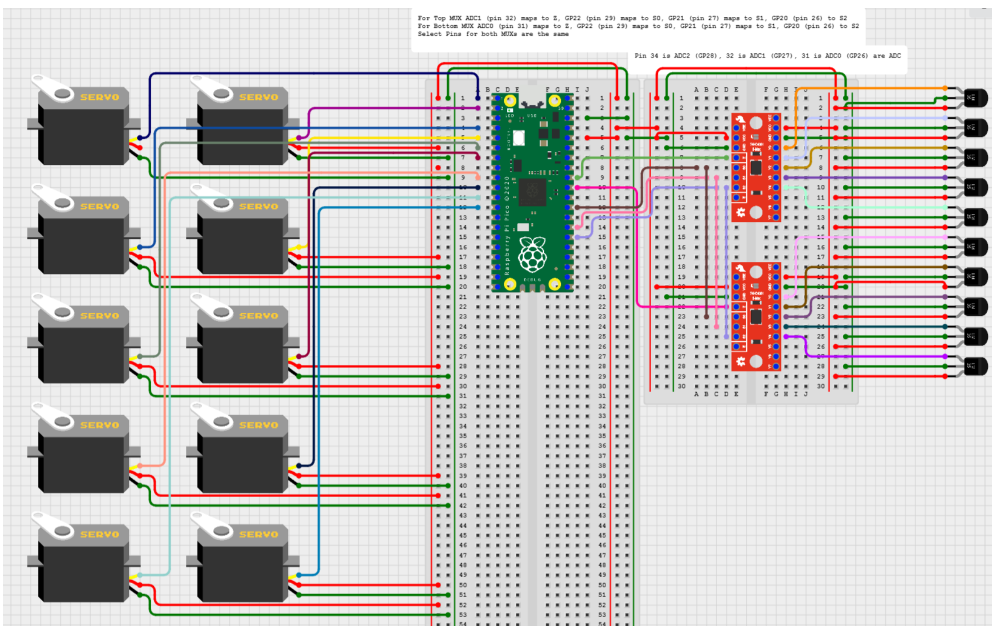


Figure 23: Inital circuit design used for testing the control glove

Supplemental Figures

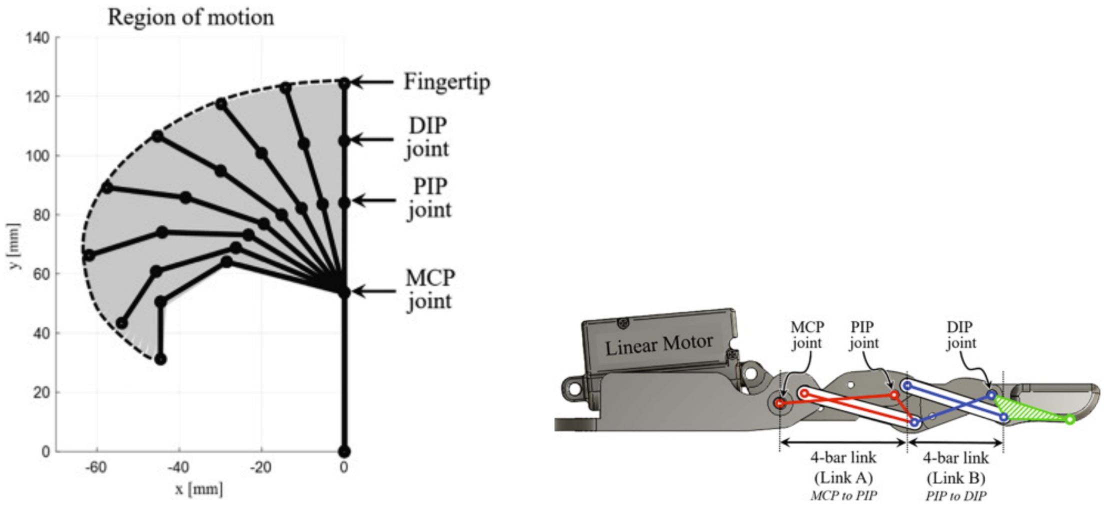


Figure 24: Finger joint angles

Figure 25: Finger angle measurement apparatus