VR Force Feedback Gloves for Lateral Finger Movement

By

Aaron Gros

Sean Kim

Yoonseo Choi

Final Report for ECE 445, Senior Design, Fall 2022

TA: Jason Paximadas

7 December 2022

Project No. 23

Abstract

This report goes over our design for force feedback gloves that track and restrict the finger's lateral movement in VR applications. We introduce the project by going over the motivation behind the project, the requirements, and introducing the design. Then we go into more detail by delving into our design decisions and going over our results. Finally, we end by discussing what worked, failed, and where to go next

Contents

1. Introduction

1.1 Problem

Virtual reality is becoming increasingly popular and with it comes the desire to physically interact with the virtual. Force Feedback gloves may be an important part of the future of VR given the increased sense of immersion they can provide when interacting with virtual objects. However, current commercial force feedback VR gloves like HaptiX [3] or Sense Glove [4] are too expensive for regular consumers and are only targeted towards industry and research. On the other hand, open-source force feedback VR gloves like Lucid Gloves [5] are economically accessible to regular consumers but they are bulky and require expertise and time to build them. Also, although some gloves like the Sense Glove [4] provide tracking for the finger's lateral movement, none provide force feedback for this movement. This type of force feedback would be desired for a greater sense of immersion when for example grabbing an object between the sides of your fingers, or when the friction of an object you're grabbing should prevent you from sliding your fingers.

1.2 Solution

We propose a new design for force feedback VR gloves that are still economically accessible to the regular consumer and that restrict the finger's side to side movement. The result is that if a user tries to grab an object in VR with the sides of their fingers, this movement is reflected in VR, and their fingers' movement is restricted when they collide with an object in VR. Since flexion/extension force feedback in combination with lateral force feedback is the ideal product for users, this project won't serve as a final product for the market but a starting point in the integration of flexion/extension force feedback with lateral movement force feedback.

Our design uses a potentiometer and a servo for each finger in combination with a mechanical design, to track and restrict finger's lateral movement, respectively. It also has a Dead Man switch to stop the gloves from restricting movement in case of an emergency. The actuators, potentiometers, and Dead Man switch are connected to a microcontroller (STM32). To share the finger's position and know when to restrict each finger's movement, the microcontroller communicates over a micro-USB through a Serial connection with a computer running the VR application. The power for the glove is provided by a barrel jack.

1.3 Visual Aid

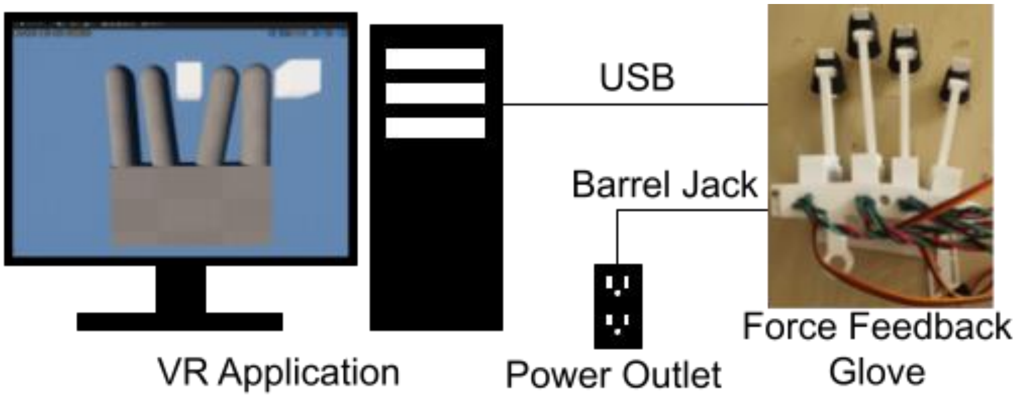

Figure 1 Depicts how the user will interact with the product.

1.4 High-Level Requirements

The gloves are able to:

- Accurately track and recreate each finger's angle within +/- 2.5 degrees
- Restrict the finger's side to side movement when they are in contact with a VR object such that the finger's restriction point is within +/- 2.5 degrees of the object's boundary
- Restrict the finger's side to side movement with 1.9 to 4 Kg/cm of torque when the finger collides with a virtual object otherwise with less than 0.5 Kg/cm.

1.5 Block Diagram

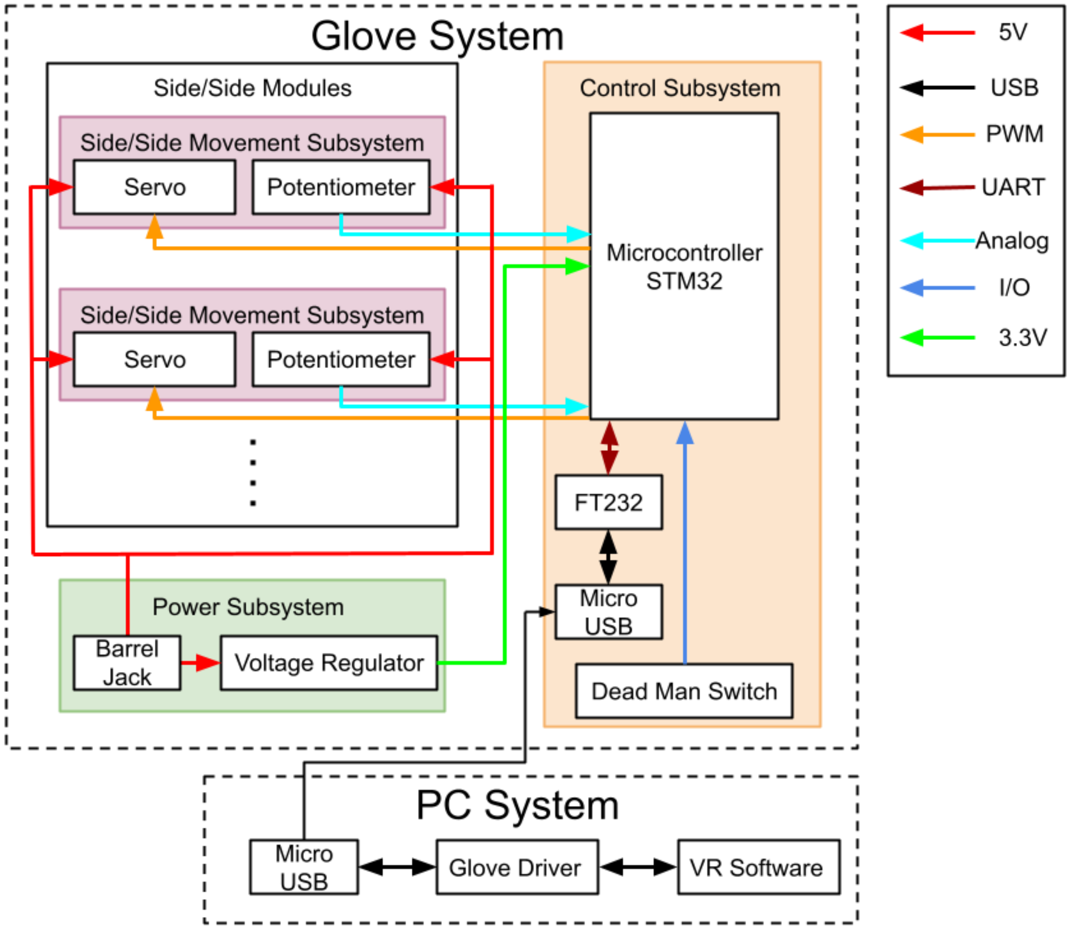

Figure 2 Shows all the different subsystems and how they interact with each other

2. Design

2.1 Control Subsystem

The control subsystem controls the servos, reads potentiometers, handles the dead man's switch, and communicates with the computer running the VR application. It receives analog potentiometer signals and a digital button signal, it sends PWM signals for the servos, and receives and sends serial signals.

Design Description

To do all this we chose to use the STM32 as opposed to the ESP32 or an Atmel processor because the STM32 seemed to have everything we needed (enough ADCs, PWM outputs, digital IO, Serial communication, and speed) as well as the best ADCs which was important for us to accurately track the finger movement. We also used an FT232 with a micro-USB instead of using the more universal USB C port because of its simplicity to implement and to avoid having to deal with the specific protocol manually.

To communicate with the PC subsystem over serial we sent the packets with the characters "{" and "}" to indicate the beginning and end of the packets, respectively, as well as separated the potentiometer values with the character ",". This was to allow the PC subsystem to reliably receive and process the packets. The packets received by the control subsystem were formatted to require as little data as possible. Therefore, we only needed to send two bits for each finger. The most significant bit to indicate if there was a collision of the left side of the corresponding finger and the least significant bit to indicate if the was a collision on the right side of the finger.

The general control flow of the control subsystem in described by *Figure. 3* and *Figure. 4*.

To get the servo to restrict the finger movement only when there's a collision, we had to manually calibrate the algorithm to function. We began by finding $Pulse_{max}$ and $Pulse_{min}$ which were the PWM pulses that made either side of the servo arm came into contact with the stopping pin. Then we measured $Angle_{max}$ and $Angle_{min}$ which were the potentiometer readings for when the servo arm came into contact with the stopping pin at $Pulse_{max}$ and $Pulse_{min}$, respectively. Last we found $Pulse_{mid}$ which was the center position of the servo.

The algorithm we used was that if there was no collision, we set the servo PWM to $Pulse_{mid}$ to allow free movement in either direction. If there was a collision, we set the servo PWM to the result of Eq. 1.

$$(Angle_{cur} - Angle_{min})\frac{Pulse_{max}-Pulse_{min}}{Angle_{max}-Angle_{min}} \qquad (1)$$

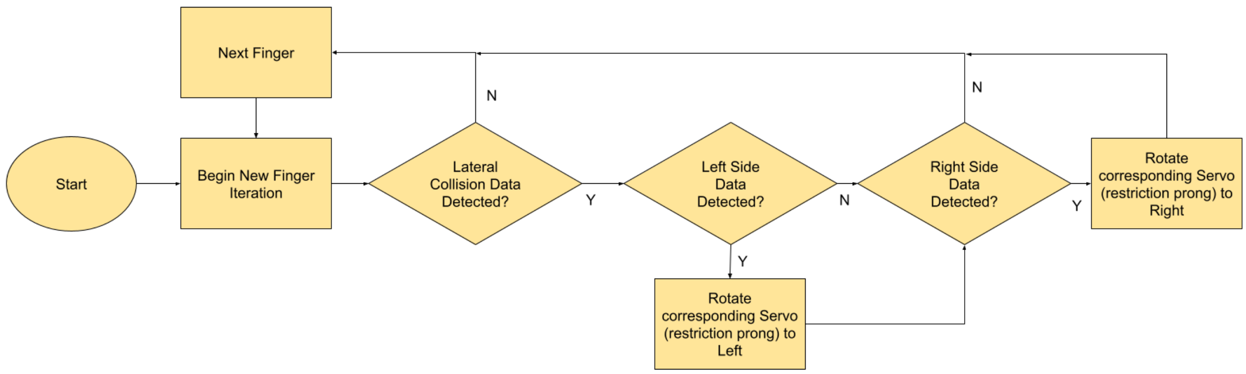

Figure 3 Shows the firmware algorithm for processing the input data from the computer and restricting finger movement if necessary

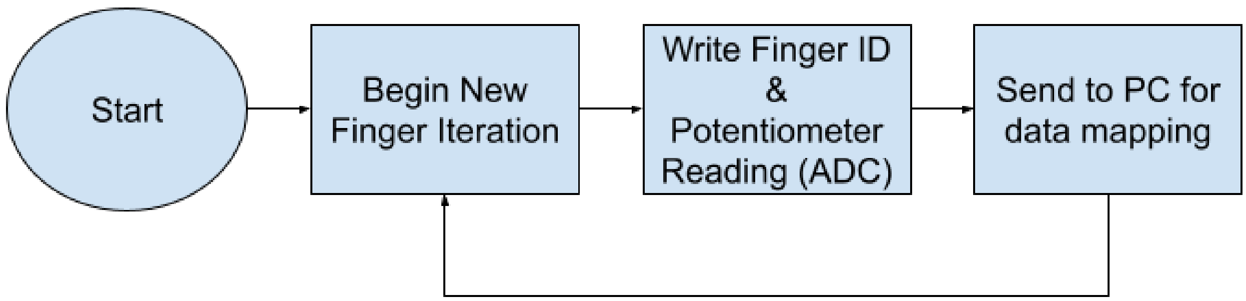

Figure 4 Shows the firmware algorithm for processing the finger's position data and sending it to the computer

2.2 Power Subsystem

The power subsystem is in charge of regulating the voltage received from the barrel jack to the necessary levels, as well as removing the necessary amount of noise to prevent the potentiometer readings from being too noisy. It sends 5V and 3.3V signals to the other subsystems.

Design Description

Our first design got its 5V from the micro-USB and 3.3V from the FT232. However, we learned that each servo could draw 1A but the micro-USB could only provide 1A, and we learned that the microcontroller could draw up to 250mA from the FT232 but it could only provide up to 50mA.

Therefore, we chose to get our 5V from a 6A wall wart power supply over a barrel jack, and we chose to use a linear voltage regulator to go from 5V to 3.3V. We also used several decoupling capacitors to remove noise so our potentiometer readings wouldn't be very noisy.

2.3 Side/Side Subsystem

The side/side subsystem is responsible for providing an interface to track the angle of the finger with a potentiometer and to restrict the finger movement with a servo. It sends the microcontroller the potentiometer signal and receives a PWM signal from the microcontroller for the servo.

Design Description

This subsystem is separated in an independent module for each finger. These modules contain a potentiometer, located at the pivot point of the joint near each knuckle, which will measure the angle of each finger. They also contain a servo with an extended arm which rotates at the same pivot point as the potentiometer. The extended arm splits into two and is located in between a stopping pin. When the extended servo arm and stopping pin collide, the finger's movement is restricted in the corresponding direction but not in the opposite direction. *Figure. 5* depicts the different parts of the mechanism.

Some alternatives to this design were to have the finger's side to side movement tied to a fixed servo, use an electromechanical brake to stop the movement, use a dc motor, or a brushless motor. However, all these options would've required a closed loop to allow for free movement or to stop the finger at a specific position. Our design allowed us to restrict both directions with just one servo while also not requiring us to implement a closed loop system to allow for free movement or to stop at a specific location.

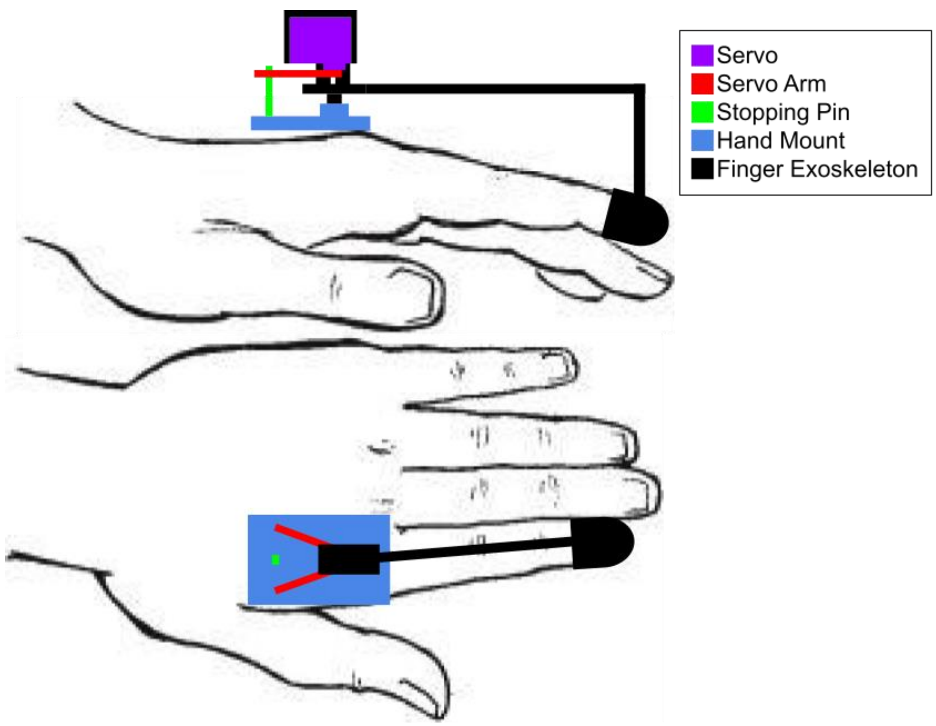

Figure 5 A drawing of the proposed mechanism to restrict the finger's movement. The hand mount and stopping pin are one piece. The servo, servo arms, and finger exoskeleton rotate with the finger. The movement is restricted by moving the servo arm to act as a hard stop against the stopping pin. Hand images taken from [6]

Since these are force feedback gloves, they will have to be able to resist a comfortable amount of torque from the fingers. If the actuators are unable to resist the torque from the fingers, when attempting to grab a virtual object your fingers will go through the object, and you'll lose your sense of immersion. However, we don't want to pull with too much torque on the fingers as this could hurt the user. Therefore, to guarantee this operation we must find the casual torque a finger can apply and the maximum torque a finger can apply.

After measuring our fingers' strength on *Table 1* we have found that our fingers can comfortably produce 0.2 Nm of torque with side-to-side movements. When exerting our fingers more we can produce more than 1.9 Kg/cm of torque with side-to-side movements and less than 4 Kg/cm of torque. Therefore, we've made a mechanical design that in conjunction with the actuator can produce a torque between 1.9 Kg/cm and 4 Kg/cm.

Table 1 Calculation of high exertion torque and low exertion torque produced by each finger

	Pinky	**Ring**	**Middle**	**Index**	**Thumb**
Hard Push Right (g)	500	360	650	360	640
Light Push Right (g)	120	100	100	105	170
Hard Push Left (g)	440	500	690	450	630
Light Push Left (g)	170	170	183	200	150
Length (cm)	8.89	10.795	11.43	10.16	11.43
Average Hard Push (g)	470	430	670	405	635
Average Light Push (g)	145	135	141.5	152.5	160
Average Hard Torque (Kg/cm)	4.1783	4.64185	7.6581	4.1148	7.25805
Average Light Torque (Kg/cm)	1.28905	1.457325	1.617345	1.5495	1.8288

2.4 PC Subsystem

The PC subsystem simulates the hand movement while also checking for collisions. It communicates with the glove over serial.

Design Description

This subsystem was designed using Unreal Engine. To form a serial connection and receive/transmit data we used SerialCOM[9], an Unreal Engine plugin, to simplify the serial interface. Since the PC subsystem's speed was greater than that of the glove, the serial data would be read in pieces. Therefore, we had to setup a buffer to put all the serial data, and we had to adjust the communication format to include a delimiter so we could search for complete packets.

For our hand model we used several separate static meshes since it was our first time using unreal engine and they seemed to be simpler to use than skeletal meshes. Each of the fingers and grabbable objects had a collision boundary which allowed us to handle the collisions through events to avoid as much delay.

3. Verification

3.1 Control Subsystem

All our requirements on the control subsystem placed restrictions on the rate at which actions should be performed to guarantee responsive performance and make sure the glove keeps up with quick finger movements. Specifically, the restrictions were on the potentiometer and dead man switch read rates, serial data transfer rates, and PWM set rates. To test the requirements, we measured how many iterations would happen in half a second since in our final implementation of our firmware we were performing all these actions once every iteration of an infinite loop.

To check this, we sent a message over serial every iteration and using the Arduino interface we were able to look at the timestamps for the serial communication. After counting, we found that there were 39 *(Appendix C Figure 6)* iterations in half a second which should give us about 78 iterations per second.

This result by itself verifies requirements 2,3, and 4 *(Appendix B Table 5)*. To verify requirement 1 *(Appendix B Table 5)* we will compute how much data is transferred in 78 iterations. The control subsystem sends 21 bytes and receives 4 bytes per iteration. That's 168 and 32 bits per iteration, respectively. Therefore, we send 13,104 bits per second and receive 2496 bits per second. This verifies the first half of requirement 1 *(Appendix B Table 5)* but not the second half. However, our current receive rate is lower because our data format is more efficient that we originally planned so we don't require that much receive data to still be fully functional.

3.2 Power Subsystem

We didn't get our PCB to work as we were unable to program our microcontroller. Therefore, we didn't get to test our power subsystem since we would've had to see how it operated while under load from the other components. However, our linear regulator was working, and we were receiving a 3.3V signal. In addition, in our proposal we missed that our microcontroller required a 3.3V power signal so this subsystem should include a restriction on its 3.3V supply

3.3 Side/Side Subsystem

Given that the side/side subsystem was responsible for providing an interface to track and restrict finger movement, all its requirements have to do with its ability to do so accurately and with enough torque.

To verify that the restrict the finger in a position close enough to the collision point in the simulation we placed a requirement of the step size for our restricting mechanism. To verify this, we marked the position of a restriction point then took the smallest step me could and marked its position. The angle we measured was 2 degrees *(Appendix C Figure 7)* which verifies requirement 1 *(Appendix B Table 7)*.

To ensure that the simulation accurately replicated the finger's position we placed a restriction of this subsystem's ability to track the finger. To verify this, we measured the angle of the physical finger with a protractor and checked the angle the simulation claimed the finger had. The angle was within simulation angle was always within 2 degrees *(Appendix C Figure 8)* of the actual angle which verifies requirement 3 *(Appendix B Table 7)*.

Finally, we wanted that the restricting mechanism be able to fight the finger's torque but not completely overpower so be placed requirements of our mechanism's torque. To verify this, we measured the torque the servo was able to generate with the servo arm attachment we designed. It was able to exert 1.1Kg at a distance of 2.54cm, therefore generating 2.79Kg-cm or torque and verifying requirement 2 *(Appendix B Table 7)*.

3.4 High Level Requirements

The first and third high level requirements were verified by the side/side subsystem's requirements 3 and 2, respectively.

To verify the second high level requirement, we measured the finger's position with a protractor and told our software to simulate a collision. Then we measured how much further towards the collision the finger could move. We found that the finger would at most move an extra two more degrees, therefore verifying the requirement.

4. Costs & Schedule

4.1 Parts

The total parts costs for everything used in the project was $232.99. (See Appendix A Table 3)

The bill of materials of just the electronics with the price adjusted for bulk orders in case the product was manufactured is $44.6021. (See Appendix A Table 4)

4.2 Labor & Schedule

The average EE graduate from ECE at UIUC has a salary of $80,296 [7] and the average CE graduate from ECE at UIUC has a salary of $105,352 [7]. Working 40 hours weekly would result in working 2080 hours per year. Meaning a $38.60/hour wage for EE graduates and a $50.65/hour wage for CE graduates. Following the calculation in *Table 2*, we get a total labor cost of $30,684.

Table 2 Calculation of the labor costs for the project

	Yoonseo	Sean	Aaron
$/Hour	38.6	38.6	50.65
Fudge Factor	2.5	2.5	2.5
Hours/Week	12	12	12
Weks	8	8	8
Individual	9264	9264	12156
Total	30684		

The total cost of our parts for the project was $232.99, bringing the total cost of the project to $30,916.99.

5. Ethics & Safety

ACM CoE 1.1 [1] states that products should be broadly accessible. Our design is built on the idea that technology should be accessible to all consumers at a cheap cost and for all interested individuals to take their ideas and build upon ours for the advancement of our society. To follow through with these ideas we've made the project open source and have chosen components that fulfill our requirements but aren't unnecessarily expensive.

In addition to our project's accessibility, we also wanted to emphasize that the VR glove's design prioritizes the safety of its users, and its features make sure the device doesn't inflict damage to the user's hand. This follows ACM CoE 2.9 [1] which clearly mentions that we should make sure our design works as it's supposed to and that our safety features are intuitive. To accomplish this, we've chosen components that can't hurt the user and have tested our design. In case of an emergency, we've also added an easily accessible dead man's switch to disengage the actuators.

Since we were influenced and built on top of other people's mechanical design and software, we feel it's necessary to follow ACM CoE 1.5 [1] and give credit to those whose work we build upon.

Our mechanical design was inspired by the LucidGloves [5], the Sense Gloves DK1 [4], and the Remote Feelings Project [8]. Our mechanical design was also partially built on top of the Remote Feelings Project [8]. Our virtual simulation of the hand used the SerialCOM[9] plugin for unreal engine to simplify serial communication.

Lastly, following IEEE CoE I.5, as engineers, not only is it our responsibility to advance technology and safety implementations but also to have the ability to acknowledge our shortcomings of technical work and to honestly correct these in a professional manner. Our group has focused on creating the best product we could, and now we'll evaluate our design and communicate its strengths and weaknesses.

6. Conclusion

Our proposed solution was supposed to be able to successfully track and restrict the user's movement according to our quantitative requirements. The final design accomplishes what we set out to do but it isn't without faults.

6.1 Accomplishments

We successfully made a mechanical design that allowed us to fulfill the rest of our requirements, implemented a virtual simulation of the hand that could recreate movement, check collisions, and communicate over serial, and wrote firmware that combined these components to accomplish all our high-level requirements.

6.2 Shortcomings

Our biggest shortcoming was that our final PCB design didn't work since we weren't able to program our microcontroller. Since our PCB design didn't work, we didn't get around to testing the power module so even though we know our voltage regulator was working and we had a 3.3V signal, we're unsure if it was within our required operation. Our final design also didn't receive 4Kb of data from the virtual simulation as mentioned in our requirements, but this value was computed using an unnecessarily large data format and in reality, we needed to receive much less data to still achieve our desired functionality.

6.3 Future Work

There are a lot of improvements that could be made through future work. To begin, the mechanical design was made after the image of one of our hands and given the variability in hand shape it doesn't work great with different hands. Therefore, a design that would allow the user to adjust the finger modules in a position that works for them would be needed. Next, the potentiometer readings were a bit noisy which showed up in the simulation as the fingers vibrating. To improve this, the noise from the potentiometer's power source could be lowered using some capacitors, or a potentiometer with less freedom of movement could be used to be able to scale down the reading further, or since our potentiometer read rate is pretty fast, we could average their readings. Next, we manually calibrated the firmware but noticed that overtime the calibration was a bit off most likely because of the potentiometers. To solve this an automatic calibration program could be written, or perhaps hall effect potentiometers could be tried to see if they don't suffer from the same issues as a mechanical wiper. Next, the lateral force feedback could be combined with flexion/extension force feedback. Finally, depending on if this were to become a commercial or open source, the pcb could be fixed or the project could be made on an easily accessible dev board.

References

[1] “ACM Code of Ethics and Professional Conduct” *ACM*. [Online]. Available: https://www.acm.org/code-of-ethics. [Accessed: 15-Sep-2022].

[2] “IEEE code of Ethics” *IEEE*. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 15-Sep-2022].

[3] “Haptic gloves for virtual reality and robotics | HaptX” HaptX https://haptx.com/. [Accessed: 15-Sep-2022].

[4] “VR & Haptic Technology Gloves | SenseGlove | Feel the virtual like it’s real” SenseGlove https://www.senseglove.com/. [Accessed: 15-Sep-2022].

[5] “Arduino/ESP32 based DIY VR Haptic gloves. Compatible with SteamVR via OpenGloves” Lucid Gloves https://github.com/LucidVR/lucidgloves. [Accessed: 15-Sep-2022].

[6] “How to draw hands, Side view drawing, Drawing body poses,” *Pinterest*. https://www.pinterest.ca/pin/232568768238451837/. [Accessed: 28-Sep-2022].

[7] “Salary Averages,” *UIUC ECE*. https://ece.illinois.edu/admissions/why-ece/salary-averages. [Accessed: 29-Sep-2022].

[8] A. Curtis, “Remote Feelings: Haptic feedback for hand based control of distant robots,” *GitHub*, Nov. 23, 2022. https://github.com/BerkeleyCurtis/EECS249_HapticGlove [Accessed: 08-Dec-2022].

[9] R. M. D. Oca, “Serial Communication Plugin for Unreal Engine,” *GitHub*, Nov. 29, 2022. https://github.com/videofeedback/Unreal_Engine_SerialCOM_Plugin [Accessed:08-Dec-2022].

Appendix A Bill of Materials Tables

Description	Manufacturer	Part Number	Link	Quantity	Cost
Voltage Regulator	TI	TLV73333PQDBVRQ1	Link	3	0.54
1.5K Ohm Resistor	Stackpole Electronics	RNCP0805FTD1K50	Link	10	0.073
10 uF Cap	Kemet	C0805L106K9PAC7800	Link	3	0.91
0.1 uF Cap	Kemet	C0805C104J8RAC7800	Link	10	0.373
Micro USB	Molex	1051330031	Link	1	1.56
100 uF Cap	Murata Electronics	GRM21BR60J107ME15L	Link	3	1.15
STM32 Microcontroller	STMicroelectronics	STM32F446RET6	Link	1	11.92
1K Ohm Potentiometer	uxcell	3362P-101	Link	2	7.99
ST-Link STM Programmer	CANADUINO	26146	Link	1	0
Scale	Etekcity	817915024638	Link	1	12.99
Female Barrel Jack	AiTrip	NA	Link	1	5.99
5V 6A Power Supply	Aclorol	NA	Link	1	15.99
FT232 on PCB	HiLetgo	3-01-0661-1	Link	1	6.49
Bearings	uxcell	a19092000ux1227	Link	1	12.99
Latching Button	DIYhz	10130	Link	1	9.99
Pin Headers	DEPEPE	DE37566	Link	1	12.99
Servos	RGBZONE	MG90S	Link	1	20.99
Velcro Straps	YiwerDer	8541757494	Link	1	9.99
Fingertip Guards	xcpmm	NA	Link	1	3.89
PLA Filament	OVERTURE	OVPLA175	Link	1	19.99
PCB V1	PCBWAY	NA	NA	1	5
PCB V2	PCBWAY	NA	NA	1	24.38
PCB V3	PCBWAY	NA	NA	1	29.6
Total					232.99

Table 3. Bill of materials for everything used in the project

Description	Manufacturer	Part Number	Link	Quantity	Cost
1K Ohm Potentiometer	Bourns	3362R-1-102LF	Link	4	0.69084
STM32 Microcontroller	STMicroelectronics	STM32F446RET6	Link	1	6.9474
Female Barrel Jack	CUI Devices	PJ-002A	Link	1	0.33482
FT232	FTDI	FT232RL	Link	1	2.95
Voltage Regulator	TI	TLV73333PQDBVRQ1	Link	3	0.174
1.5K Ohm Resistor	Stackpole Electronics	RNCP0805FTD1K50	Link	10	0.00657
10 uF Cap	Kemet	C0805L106K9PAC7800	Link	3	0.29164
0.1 uF Cap	Kemet	C0805C104J8RAC7800	Link	10	0.12272
Micro USB	Molex	1051330031	Link	1	0.76756
100 uF Cap	Murata Electronics	GRM21BR60J107ME15L	Link	3	0.38638
5V 6A Power Supply	Aclorol	NA	Link	1	15.99
Latching Button	DIYhz	10130	Link	1	2
Servos	RGBZONE	MG90S	Link	4	2
PCB	PCB Way	NA	NA	1	1
Total					44.6021

Table 4. Bill of materials of just the electronics with the price adjusted for bulk orders in case the product was manufactured

Appendix B Requirements & Verifications Tables

Table 5 R&V table for the control subsystem

Requirements	Verifications	
1. Must be able to send 4Kb of position data to the computer per second and receive 4Kb of data from the computer per second over UART.	1. **A)** Connect the microcontroller to the computer via a micro-USB. **B)** Measure how long it takes to send 4Kb from the microcontroller to the computer and 4Kb from the computer to the microcontroller	1. Half Verified
2. Must sample each potentiometer at least 30 times per second	2. **A)** Connect 5 potentiometers to the microcontroller **B)** Read the analog signal for each potentiometer **C)** Repeat 29 more times **D)** Measure the time it takes to finish reading the signals.	2. Verified
3. Must sample the dead man's switch at least 2 times per second	3. **A)** Read the digital input switch **B)** Measure how long it takes to finish sampling	3. Verified
4. Must be able to adjust the PWM signal at least 5 times per second	4. **A)** Connect the digital out on the microcontroller to an oscilloscope **B)** Change the PWM signal five times **C)** Change the PWM signal five times	4. Verified

Table 6 R&V table for the power subsystem

Requirements	Verifications	
1. Must be able to output 5V +/- 1V from a 5V source	1. **A)** Using an oscilloscope, measure the output voltage and check if it stays with the target	**1. Not Verified**

Table 7 R&V table for the side/side subsystem

Requirements	Verifications	
1. The mechanical design in conjunction with the actuator must have a step size of at most 3 degrees	**1. A)** Increase the PWM 1 at a time until the servo moves **B)** Measure the change in angle with a potentiometer	1. Validated
2. While in conjunction with the mechanical design, they must produce a stall torque in between 1.9 to 4 Kg-cm	**2. A)** Set a 2 Kg weight a centimeter away from the mechanical design's pivot point **B)** Set the device such that the force will be perpendicular to the ground an see if the design can hold the weight **C)** Try the Same with a 4 Kg weight and see if it fails	2 Validated
3. The mechanical design in combination with the potentiometer must measure the finger's angle within +/- 2.5 degrees	3. **A)** Measure the finger's angle with a protractor and compare it to the measurement by the potentiometer **B)** Repeat for a couple of different locations	3 Validated

Appendix C Verification Results

Figure 6 The results of the tests to verify the control subsystem. Highlighted in blue are thirty lines. On the left is the time stamp, and on the right is the serial data transmitted. From 04:36:47.5 until 04:36:48:00 39 messages are transmitted

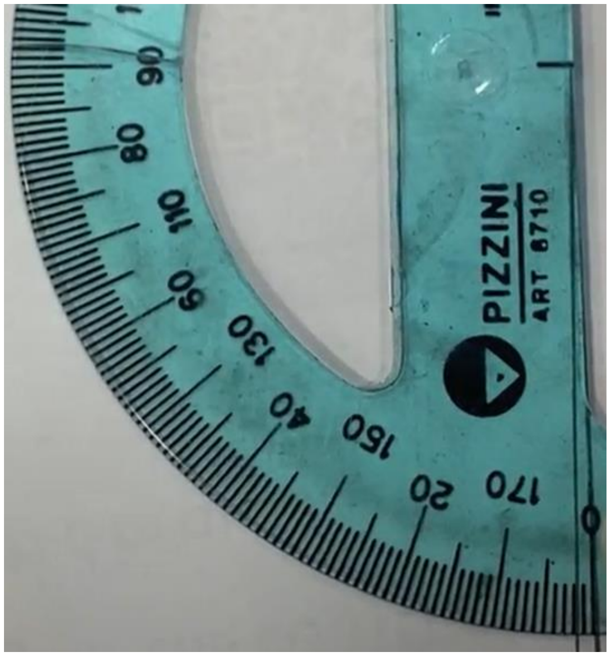

Figure 7 The results of the tests to verify requirement 1 in the side/side subsystem. In the bottom right there are two line that represent the step and measure just under two degrees.

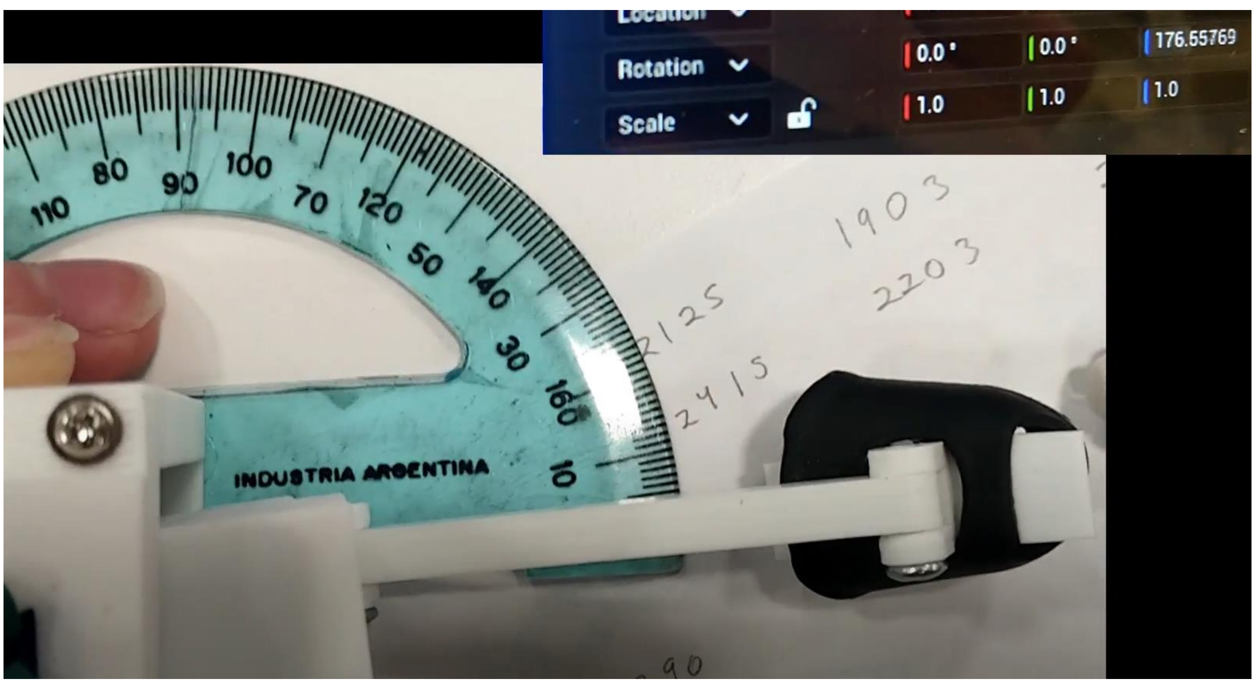

Figure 8 The results of the tests to verify requirement 3 in the side/side subsystem. The middle of the image shows that the finger is at 175 degrees and top right shows the angle in the virtual simulation. The angle is just over 1.5 degrees off.