Haptic Feedback Controller for Robotic Neurointervention

A Major Qualifying Project
Submitted to the Faculty of
Worcester Polytechnic Institute
in partial fulfillment of the requirements for the
Degree in Bachelor of Science
in
Electrical and Computer Engineering
Robotics Engineering
Biomechanical Engineering
Computer Science
Data Science
By

Troy Mullenberg

Ethan Wilke

Shiyue Wang

Date: 3/24/23

Sponsoring Organization:

Approved By:

Professor Zheng, Yihao, Advisor

Professor Zhang, Haichong, co-Advisor

Professor Zhang, Ziming, co-Advisor

Abstract

This exploratory research project aimed to develop, prototype, and evaluate an intuitive haptic feedback-centric user interface to simulate forces to the likeness of traditional methods of neuro-interventional procedures. Decoupling the interventionist from the location of the procedure seeks to lower the prolonged radial exposure to the surgeon as well as the barrier of entry needed for the adoption of existing interventionist patient-side robotics for this procedure on the market. The system proposed and implemented methods to replicate the traditional over-the-wire operation with accurate tactile sensation on the wires and catheters. The team was able to effectively utilize mechanical, electrical, software, and systems engineering techniques to provide haptic feedback within a 5% accuracy of force replication. Further work on this technique could yield a real-time system that could allow surgeons to operate on patients regardless of location or robotic platform being employed.

List of Contents

Authorship Table

Section	Main Author	Editor
Abstract	Troy Mullenberg	Ethan Wilke
1.1. Intro to Haptic Robotics in the Medical Field	Ethan Wilke	Shiyue Wang
1.2. Literature Review	Shiyue Wang	Troy Mullenberg
2.1. Mechanical implementation	Ethan Wilke	Troy Mullenberg
2.2. Electronic Hardware implementation	Troy Mullenberg	Ethan Wilke
2.3. Software system Implementation	Troy Mullenberg	Ethan Wilke
2.4. Client Side Software Implementation	Troy Mullenberg	Ethan Wilke
2.4.1. Client Side data flow	Shiyue Wang	Troy Mullenberg
2.4.2. ROS Implementation/exploration	Shiyue Wang	Troy Mullenberg
2.5. Firmware Implementation	Troy Mullenberg	Ethan Wilke
2.5.1. Data flow in the firmware	Troy Mullenberg	Ethan Wilke
2.5.2. Control loop	Troy Mullenberg	Ethan Wilke
2.5.3. Data Parsing	Troy Mullenberg	Ethan Wilke
2.5.4. Calibration Method	Troy Mullenberg	Ethan Wilke
2.5.5. Data Post Processing	Troy Mullenberg	Ethan Wilke
2.6.1. Calibration accuracy	Troy Mullenberg	Ethan Wilke
2.6.2. Accuracy testing	Troy Mullenberg	Ethan Wilke
2.7.1. Interfacing with Existed System	Shiyue Wang	Troy Mullenberg
3.1. Mechanical Friction/realization	Ethan Wilke	Troy Mullenberg
3.2. Calibration of the system	Troy Mullenberg	Ethan Wilke
3.2.1. Linear Calibration	Troy Mullenberg	Ethan Wilke
3.2.2. Rotational Calibration	Troy Mullenberg	Ethan Wilke
3.3.1. Rotational results	Troy Mullenberg	Ethan Wilke
3.3.2. Linear results	Troy Mullenberg	Ethan Wilke
3.4.1. “Real-time performance”	Shiyue Wang	Troy Mullenberg
3.5. Force application rising time	Shiyue Wang	Troy Mullenberg
3.5.1. Bottlenecks of the implementation	Shiyue Wang	Troy Mullenberg
4.1.1. Smoother linear rail	Ethan Wilke	Troy Mullenberg

4.1.2. More poles on the motors	Ethan Wilke	Troy Mullenberg
4.1.3. Current (amp) sensors on the motors	Troy Mullenberg	Ethan Wilke
4.1.4. Communication over a full duplex medium	Troy Mullenberg	Ethan Wilke
4.1.5. Real-time force feedback	Troy Mullenberg	Ethan Wilke
4.2.1. Clutch placement	Ethan Wilke	Troy Mullenberg
5.1. Engineering ethics	Ethan Wilke	Troy Mullenberg
5.2. Societal and global impact	Troy Mullenberg	Ethan Wilke
5.3. Environmental impact	Troy Mullenberg	Ethan Wilke
5.4. Codes and standards	Ethan Wilke	Troy Mullenberg
5.5. Economic factors	Ethan Wilke	Troy Mullenberg

1. Prior Art

1.1. Intro to Haptic Robotics in the Medical Field

The field of medical robotics has seen significant growth and development in the last decade. Medical robots are now widely used in various applications, including surgery, rehabilitation, diagnosis, and patient care. These systems offer several benefits compared to their traditional procedures, such as improved precision, reduced invasiveness, and enhanced safety, leading to better outcomes for patients. The focus on producing minimally invasive robotic platforms for various surgical procedures has been a focus of the market. The use of robotic systems in surgical procedures has been shown to reduce surgical trauma, minimize the length of hospital stays, and speed up recovery times. Regarding diagnosis, medical robots are used for imaging, biopsy, and pathology applications. These robotic systems can enhance the accuracy and precision of procedures such as biopsies, reduce procedure times, and increase patient comfort.

One area of medical robotic research that has taken off in recent years is the development and implementation of neuro/cardiac interventional robots that assist doctors during endovascular procedures. Endovascular surgery is a cutting-edge, minimally invasive technique used to treat blood vessel issues including aneurysms, such as spinal disorders, which are enlargements or "ballooning" of the blood vessels. To access the blood vessels, a small incision must be made close to each hip during the procedure. A catheter, which is a long, narrow, flexible tube, is used to enter an endovascular graft via the femoral arteries and place it into the aorta. This particular fabric tube device is supported by stainless steel self-expanding stents. Once in position, the graft enlarges and closes the aneurysm, stopping blood flow to the area. The graft is permanently implanted in the aorta.

By detecting the modest axial forces/torques at the fingers (haptics) in conjunction with 2D fluoroscopy imagery, the operator is able to route the catheters and guidewires through the various arteries in the body in practice. Insertion, retraction, and twisting at the proximal end are used to navigate the catheter. Catheter instability, operator experience, vessel tortuosity, and angulation, which make it difficult to direct instruments and reach the target location, are all factors that might exacerbate difficulties and raise the risk of procedural consequences.

Currently available robotic-controlled endovascular navigation devices can be broken down into two main categories: magnetic and electromechanical. Commercial platforms that use electromechanical motion include the CorPath vascular robotic system, Amigo remote catheter system, and Sensei and Magellan robotic catheter systems from Auris Surgical Robotics. There are many advantages to the use of these robots.

Robotic systems provide more precise positioning, navigation, and stability, especially in anatomical structures with more complex shapes. The technique may potentially offer other advantages to the operator and patient in addition to these technological ones. Robotic technology enables remote catheter and guidewire control, which not only lowers radiation exposure on the radiologists but also has the potential to facilitate remote treatments where the primary operator controls the device from a different location. Also, the operator can stay sitting the entire time, which may increase comfort during lengthy procedures, lessen tiredness, and ultimately improve performance.

Robotic technology is user-friendly, simple to pick up, and offers proven advantages assessed by reliable measures. Robotic catheters are able to maneuver more efficiently and with greater accuracy.

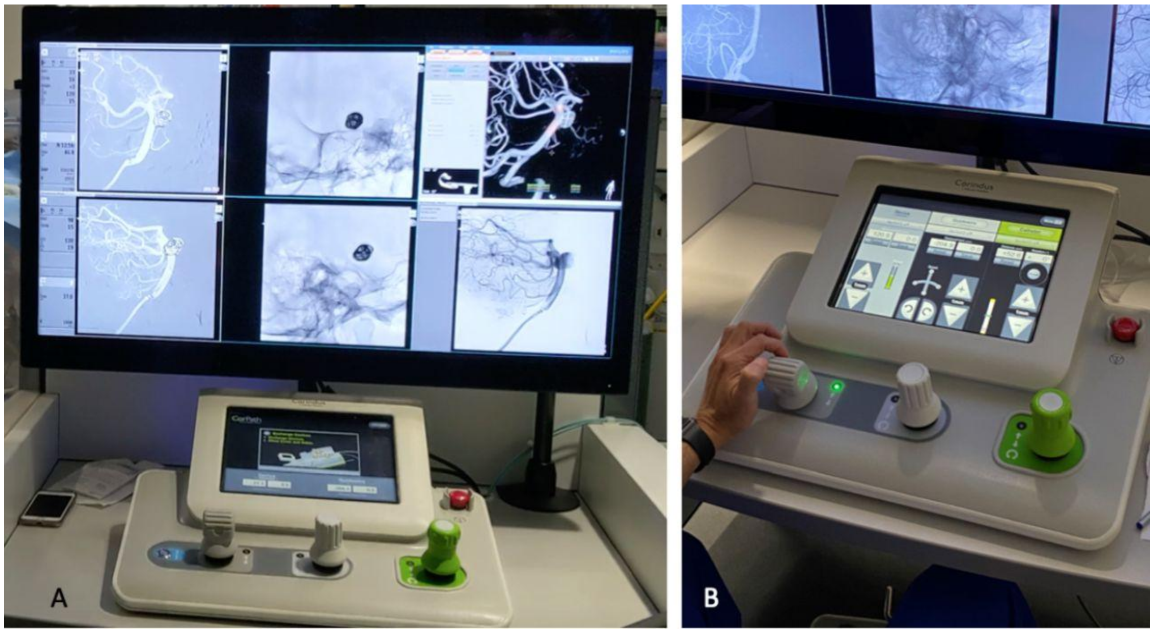


Figure 1: CorPath GRX Robotic System control panel [2]

The largest drawback of current robotic-controlled endovascular navigation devices is a complete lack of haptic feedback to the operator during the procedure. As an example in figure 1, many of the solutions on the market utilize simple joysticks and buttons to communicate human input to the robot. These joysticks and buttons provide no force or torque feedback (haptics) to the operator, which completely removes the previously important haptic feedback aspect of the "manual" procedure. Our project aims at providing a proof-of-concept control system that combines all of the benefits of currently available robotic-controlled endovascular navigation devices while also maintaining the formality and haptics of the manual procedure.

Haptic robots are robots that are designed to interact with humans using touch or force feedback. These robots have been used in various applications such as teleoperation, virtual reality, and rehabilitation. In medical applications, haptic robots are being used for tasks such as surgery, diagnosis, and training. The transition from traditional medical robots to haptic robots has been gradual but is gaining momentum in recent years.

One of the significant benefits of haptic robots in medical applications is the enhanced sense of touch and feedback they provide. Haptic robots can simulate the sensation of touch, providing the surgeon or operator with a more realistic and accurate representation of the patient's anatomy. This allows for more precise and accurate procedures, reducing the risk of errors or complications. Additionally, haptic robots can provide feedback to the surgeon or operator, allowing them to better understand the forces and pressures they are exerting on the patient's body, which can be critical in minimizing damage to surrounding tissue and organs.

Another advantage of haptic robots is their potential to improve medical training. Haptic simulators can provide medical students and professionals with a realistic experience of performing a surgical procedure or diagnostic test, without the need for human patients. This allows for repeated practice and feedback, enhancing the trainee's skills and improving patient outcomes.

1.2. Literature Review

The development of an interventional robot with haptic feedback has the potential to significantly improve clinical practices and patient outcomes. With this technology, medical professionals can perform procedures with greater precision and accuracy, reducing the risk of complications and improving overall treatment efficacy.

The development of interventional robots with haptic feedback has been an active area of research and development in recent years, and there are several products and solutions available in the market. Immersion Corporation is a leading provider of haptic technology, offering solutions for gaming, automotive, and medical applications. Their haptic technology uses a combination of vibrations and forces to create realistic and immersive experiences for users. Vibration is a commonly used methodology in haptic feedback solutions, as it can simulate various sensations such as impact, texture, and movement. Force feedback is another commonly used methodology in haptic feedback solutions, as it provides users with a realistic sense of touch and allows them to interact with virtual objects as if they were real. Ultraleap is a company that provides haptic technology for virtual and augmented reality applications. Their technology uses ultrasound to create a tactile sensation, allowing users to feel objects and textures in virtual environments. Ultrasound is a newer methodology that is gaining popularity in haptic feedback solutions. It uses sound waves to create a tactile sensation, allowing users to feel virtual objects and textures. Ultimately, the use of an interventional robot with haptic feedback can help to ensure better quality treatment for patients and improve overall clinical outcomes, making it a valuable tool for medical professionals in various fields. This paper will be going to talk about force feedback implementation in neuro-interventional surgical equipment. The device improved below described in this paper implemented haptic feedback this paper will allow the operator to "feel" what the robot is doing, providing them with an added level of control and feedback during procedures.

To implement force feedback effectively, it is important to design the system with consideration for the specific application and user requirements. The following literature will include choosing the appropriate technology, selecting the appropriate type and strength of forces to apply, and designing the feedback to be accurate and responsive to user input.

2. Methods and Implementation

2.1. Mechanical implementation

In order to design a controller that was more representative of the manual procedure, there were three main mechanical objectives that needed to be successfully implemented in the mechanical design of the control system.

- Two independent degrees of freedom (linear and rotation), each with low levels of friction
- Operator adjustable interface "handle" or collet piece
- The linear range of motion to be similar to the manual procedure

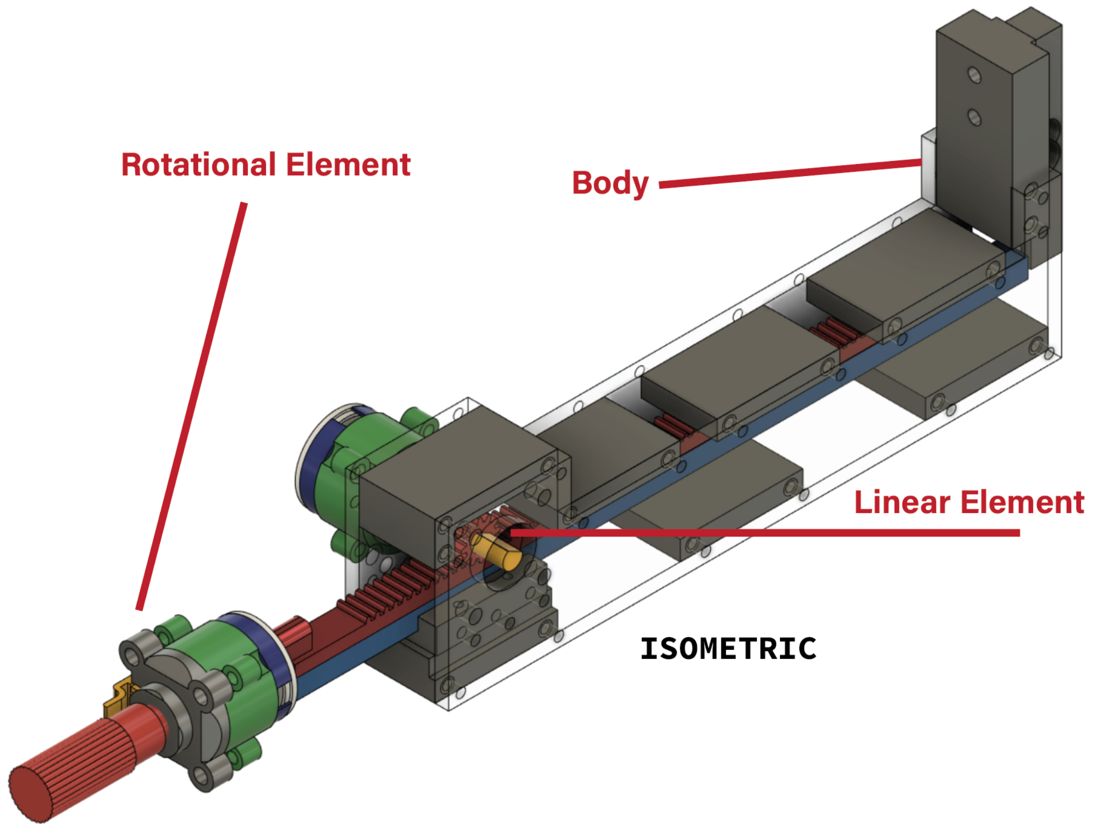


Figure 2: Final CAD design of robotic-controlled endovascular haptic navigation controller

After several prototypes, the team decided to go with the implemented CAD design shown above in figure 2.

The controller consists of three main components listed below.

ROTATIONAL ELEMENT

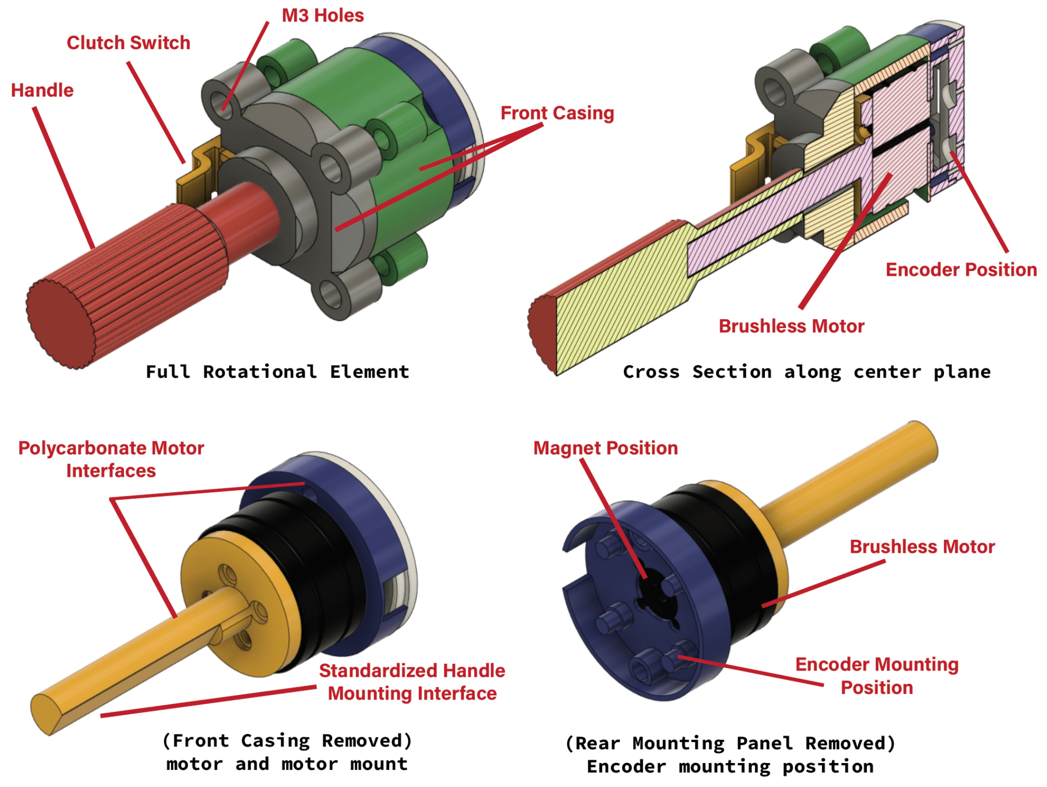

M3 Holes
Clutch Switch
Handle
Front Casing
Encoder Position
Brushless Motor
Full Rotational Element
Cross Section along center plane
Polycarbonate Motor Interfaces
Magnet Position
Brushless Motor
Standardized Handle Mounting Interface
Encoder Mounting Position
(Front Casing Removed) motor and motor mount
(Rear Mounting Panel Removed) Encoder mounting position

Figure 3: Final CAD design of the rotational element

The rotational element of the controller is responsible for providing the surgeon with the rotational torque haptic feedback from the robotic platform. In order to provide closed-loop control to the brushless motors, a magnetic rotational encoder was used. To take advantage of the encoder, a custom enclosure had to be constructed securing and positioning the encoder behind the motor and magnet attached to it. This can be seen in the blue and white elements above. These parts were secured together using M3 threaded inserts and bolts. To transfer the torque of the motor to the user, a standardized handle was constructed that would be secured to the front of the motor hub, essentially converting the hub motor into a keyed shaft motor, seen above in yellow. During the operation of the controller, the motor will put off heat, so to avoid any plastic deformation/melting, the motor/encoder housing and standardized handle were 3D printed out of polycarbonate chosen for its heat-resistant properties. The motor and shaft were fully enclosed inside a front casing, seen in green and gray, preventing any accidental interference with the motor. They also serve as the mounting position for the small clutch button mounted on the side of this case. The handle, seen in red, is designed to be "hot-swappable" on and off of the standardized handle mounting interface, allowing the user to choose their preferred handle characteristics and the potential for future customizations. The back of the rotational element is linearly and rotationally secured via two M3 bolts into the linear element.

LINEAR ELEMENT

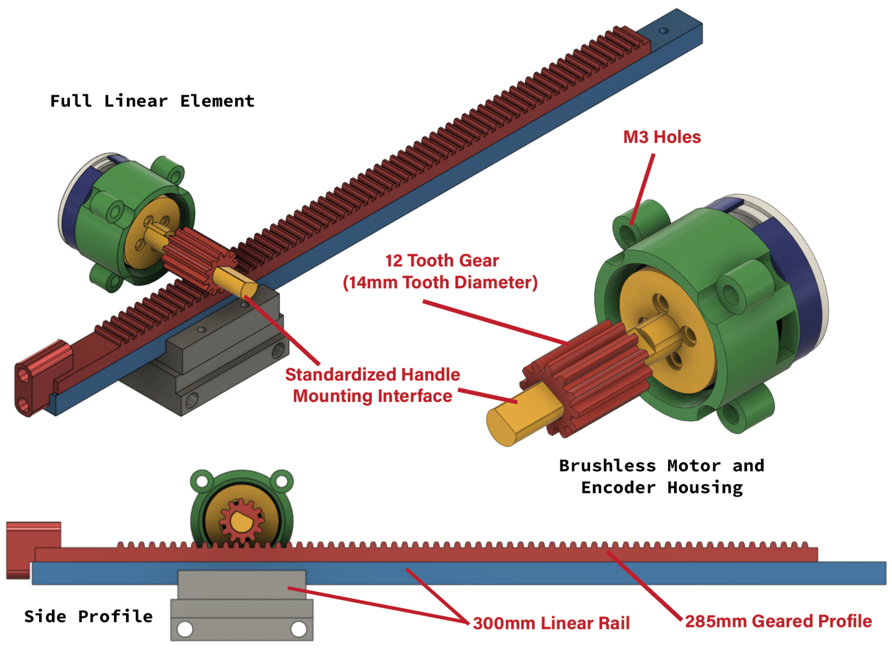


Figure 4: Final CAD design of the linear element

The linear element of the controller is responsible for providing the surgeon with linear haptic feedback from the robotic platform. In order to provide closed-loop control to the brushless motors, a magnetic rotational encoder was used. The same encoder/motor housing used in the rotational element was also implemented in this linear motion design. Instead of the handle (from the rotational element), a 12-tooth gear is mounted to the standardized handle mount coming off of the motor. This gear perfectly meshes with the 285mm linear gear profile, converting the rotational motion of the motor into a perpendicular linear motion. The gear was designed to be as small as possible to reduce the amount of gear reduction that occurs in the system during motion and reduce the amount of total friction. In order to also further reduce the friction of the system, a 300mm linear rail was used to “regulate” and “guide” the linear motion generated by the motor and gear. The linear rail consists of two parts, a mounting base with ball bearings and the 300mm machined aluminum rail that rides in the mount. The mount was secured to the body of the controller by a 3D-printed piece via 2 M3 bolts, seen below the mount. The rigidity of the linear rail also allows the fairly heavy rotational element to not bend down from gravity and therefore maintain a consistent height over the table throughout the linear range of motion. The rotational elements are secured to the body element via M3 bolts and threaded inserts.

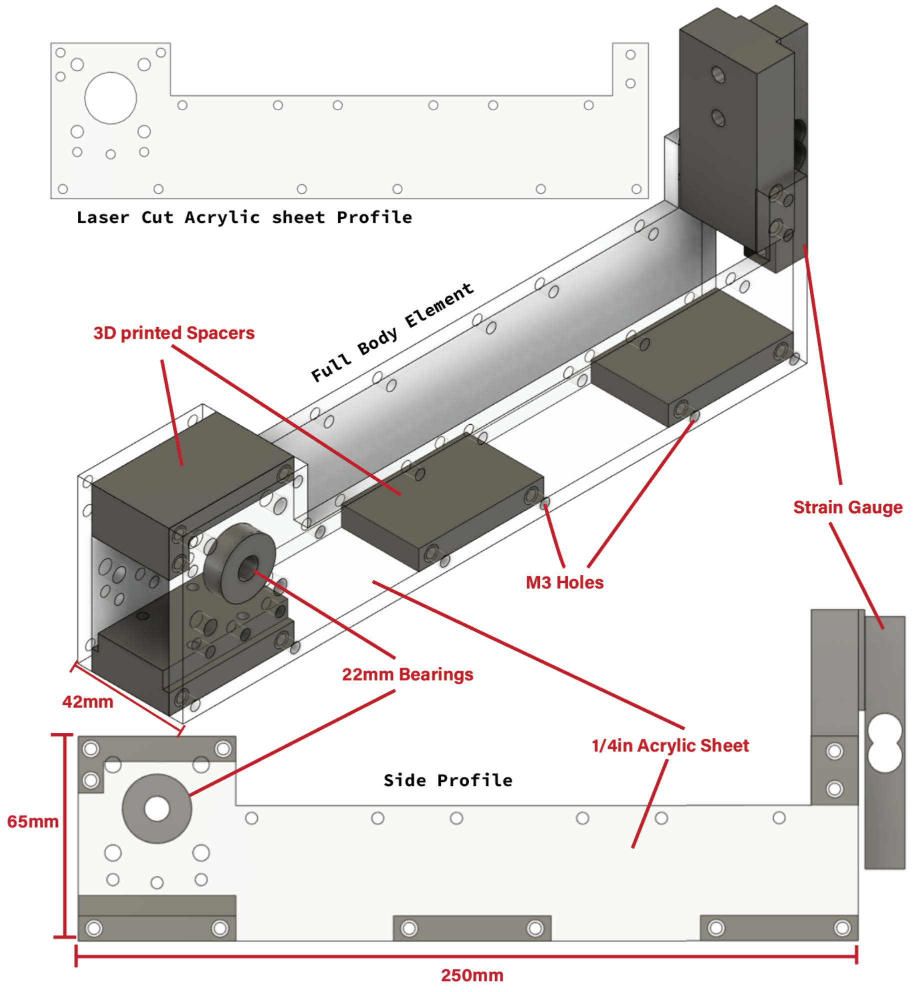


Figure 5: Final CAD design of the controller body

The body element of the controller is responsible for implementing the linear and rotational elements together into one accurate and stable platform while also serving as a mounting position for calibration sensors and electronics. The linear element can be attached to either side of the body depending on the surgeon's preference (left or right-handed). The linear elements shaft is secured to the body by two 22mm rotational bearings that reduce friction while also guaranteeing consistent teeth interactions between the rotational and linear gears. The sides of the body were designed to be manufactured with laser-cut acrylic due to the accuracy and rigidity properties that are desired for this controller body. At the rear of the body, there is a load cell secured that will allow for the pre-operation calibration of the linear feedback system.

Manufacturing and Final Prototype

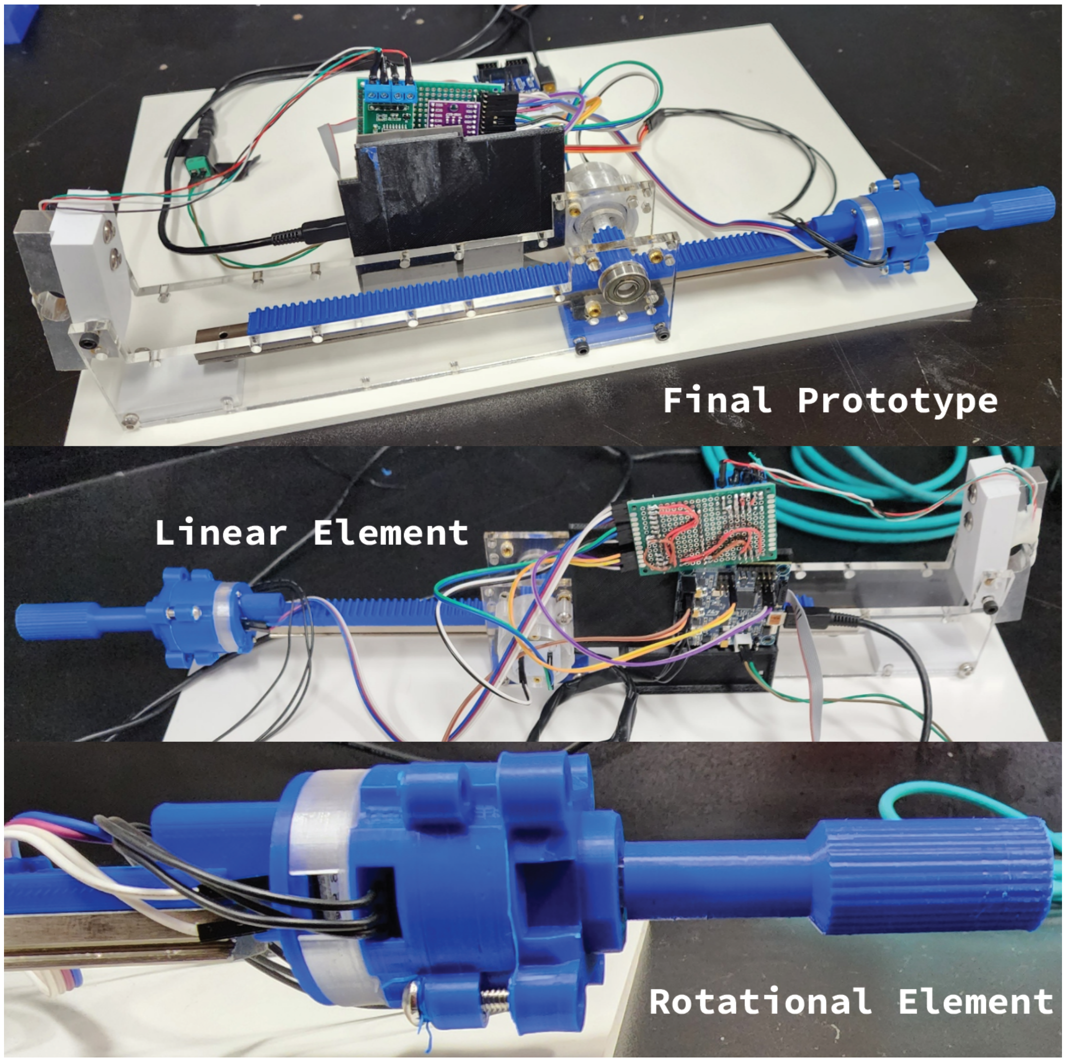


Figure 6: Final Manufactured Prototype

As seen in the figure above, the CAD design was used to manufacture a working prototype of the controller. To create the body, the main side panels were manufactured using lasers cut ¼ in acrylic. These two side panels are secured together via 3D printed parts seen printed in the white filament. Also printed in white PLA filament is the strain gauge mount mounted at the rear of the body and the linear rail mount at the front. Both of the motor/encoder housings were 3D printed out of a combination of PLA+ (blue) and heat-resistant polycarbonate (Clear). The linear motor housing is secured together to the body of the controller with M3 bolts and through the two 22mm bearings secured into the acrylic side panels. Both the circular and linear gear was 3D printed out of PLA+ (blue). The rotational element was secured to the front of the linear rail section using two M3 bolts and threaded inserts. The electrical components were attached to the acrylic side panels using velcro adhesive patches. The final step was securing the controller to a large white base which provided the whole controller with rigidity and prevented the system from moving unintentionally across the workspace.

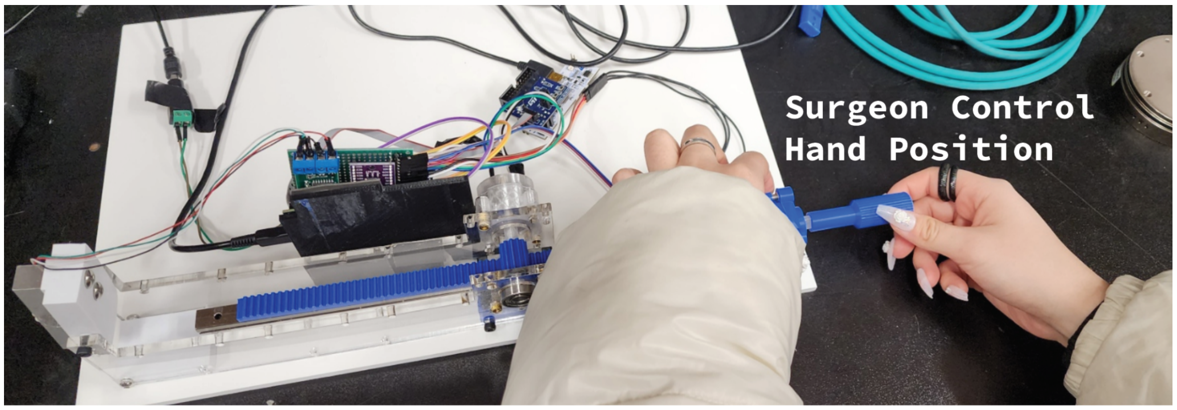


Figure 7: Controller Prototype Demo

The figure above demonstrates how a surgeon would use the controller for a new intervention procedure. The control hand (right hand in this case), applies rotational and linear force inputs to the controller using the tabletop surface as a hand rest to provide more fine-tuned and steady human inputs. At the same time, the controller will be providing real-time force and torque feedback from the robot side directly to the surgeon's hand. The rigidity of the system allows for a much greater range of motion without the risk of bending or kinking that may occur with a wire.

2.2. Electronic Hardware implementation

Looking into options for electronic hardware there was a multitude of factors contributing to the parts chosen. This was implemented with an STM PIN233 gimbal evaluation board (Figure 8), 2 gimbal drone-sized motors, 2 AS5600 magnetic encoders, and 1 I2C multiplexer. During the search one of the underlying constraints was part availability and accessibility within a timely manner, many of the considered microcontroller units and motor drivers compatible with field-oriented control methods became limited. In addition to suitable sensors and sensorless motors with accompanying documented parameters proved to be a challenge in the search.

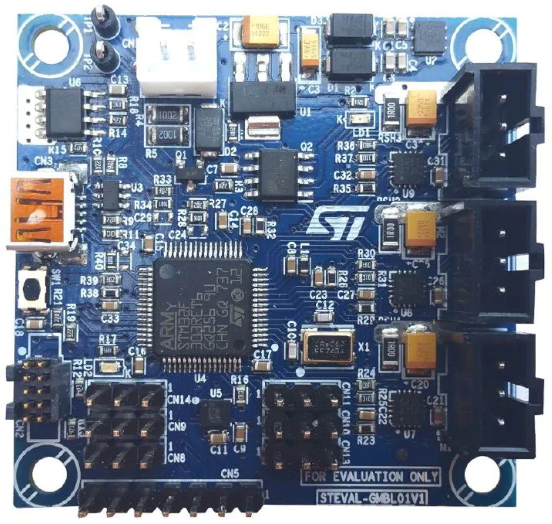


Figure 8: STEVAL-GMBL02V1 housing 3 PIN-233 motor drivers and the STM32f303X [3]

The microcontroller was evaluated on the following criteria; the ability for real-time performance, number of serial ports/communication protocols, compatibility with motor drivers, compatibility with existing libraries and the Arduino build framework, communication protocols are broken out into pin/accessibility of additional pins.

Most solutions were coupled with a motor driver solution on the same PCBA and evaluated on the following criteria; a number of half bridges, max stall current rating, max voltage, compatibility with brushless direct current motors, and drive method.

Using this criterion the following table was made:

Table 1: Microcontroller requirements

Platform considering	Real time / clock rate	Serial communication	compatible with BLDC Motors	Compatible with existing libraries	I2C / usable pins	Motor driver name	Max motor voltage	number of half bridges	Current sensing
ST-EVAL GMBL02V1	72 MHz	1 Serial CDC port, 1 Serial RS-232 on programmer	Up to 3	Using generic STM configuration	1 I2C bus, 1 SPI bus accessible from board pins	STSPIN233	0.5 - 8.4 v	3	no
BGC 3.1	8 MHz	1 Serial port	Up to 2	Requires flashing a new bootloader	1 I2C bus, 1 SPI bus accessible from board pins	ST l6234	7 - 52 V	3	no
Stepper driver L298N	n/a	n/a	n/a	yes	n/a	ST L298N	5v-35v	2 full bridg	no

Due to time constraints and documentation available on the web pages the ST-EVAL GMBL02V1 was chosen for its enumerated features, low voltage motor control, and its ability to be programmed via an ST-Link utility with ease.

However, the evaluation board provided only 1 I2C bus on accessible pins. This is an issue when considering the selected magnetic encoders, AS5600 has a burnt-in address for the bus of 0x36. With the need to put 2 sensors on the same bus implementation a TCA9548A I2C multiplexer was implemented to resolve the issue. This reduced the transmission rate of the data lines but was not a bottleneck in the end-user operation which will be discussed later.

The type of motor chosen for this application was done when considering the friction and overall resistance the motor would have in its lowest state, as one low friction and no noticeable stepping was a desired result. Thus we elected to search for brushless DC motors (BLDC) as a result. Further, the BLDC motors used in this implementation were chosen based on availability as well as relative size and voltage. One issue encountered during this search is that many of the off-the-shelf motors do not have firm specifications as to the minimum torque values that you can obtain from them, only max ratings. Therefore the team evaluated several options and concluded that a BLDC advertised for use in small handheld gimbals would also be suitable for our application. Due to part unavailability of factory-sensored motors, separate magnetic encoders were purchased and mounted to create the two motor units.

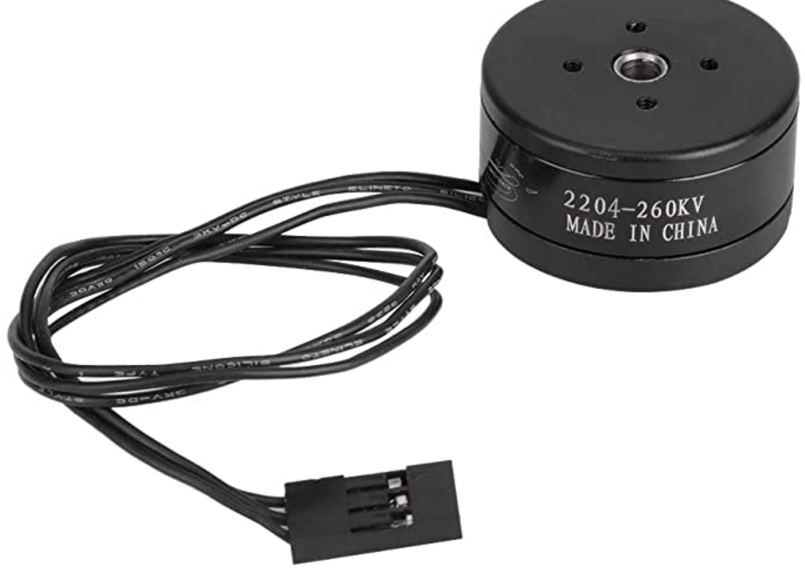


Figure 9: Gimbal style motor was chosen for the prototype [4]

The following diagram depicts how all of the components described above are wired together omitting power and ground wires to the peripheral devices:

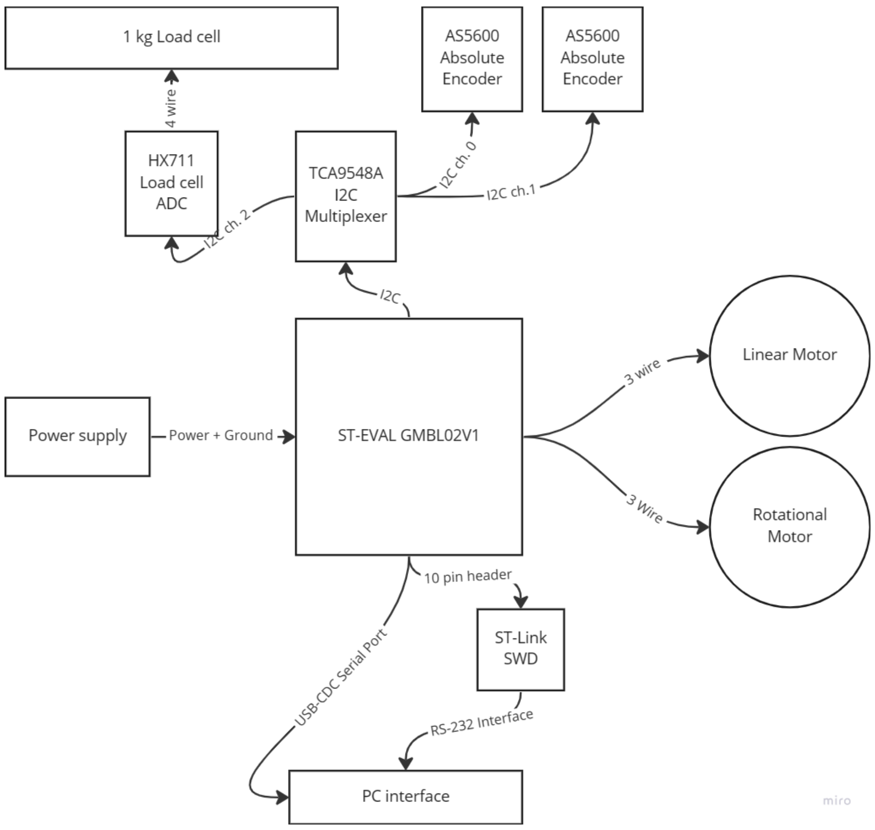


Figure 10: Wiring/connection diagram of the electrical system.

Per the specifications of the microcontroller board, the power supply of the system is an 8.4 volt 1.0 Amp barrel connector AC-DC power supply that interfaces with the battery input port on the system. In niche applications, this would allow the system to be portable and operate on a standard battery if needed. The software debug interface (SWD) breakout board was provided with the main controller, however, the ST-Link portion of the USB-RS232 connection port was broken off of a separate STM Nucleo development board to be used in this implementation as well. This board interfaces with the 10-pin connector on the board and the breakout board and then slots on top of the SWD of an STM Nuculo board development board. The motors plug into ports 1 and 2 of the controller leaving port 3 vacant.

A perforated circuit was wired to reduce the wires between the I2C multiplexer and the connected sensors and their respective power and ground connections. For powering the sensor pins from the remote controller and inputs were used. From the schematic provided for the board pins PB8 and PB9 were broken out into the “Remote control input” pins and were used for the I2C bus.

2.3. Software system Implementation

The overarching software for the device was split into 2 main categories, the client side (PC interface) and the firmware on the controller. As the implementation's goal was to be able to connect to existing systems a data communication schema was developed to ensure fluidity throughout the system. The following figure shows the overarching data flow of the system.

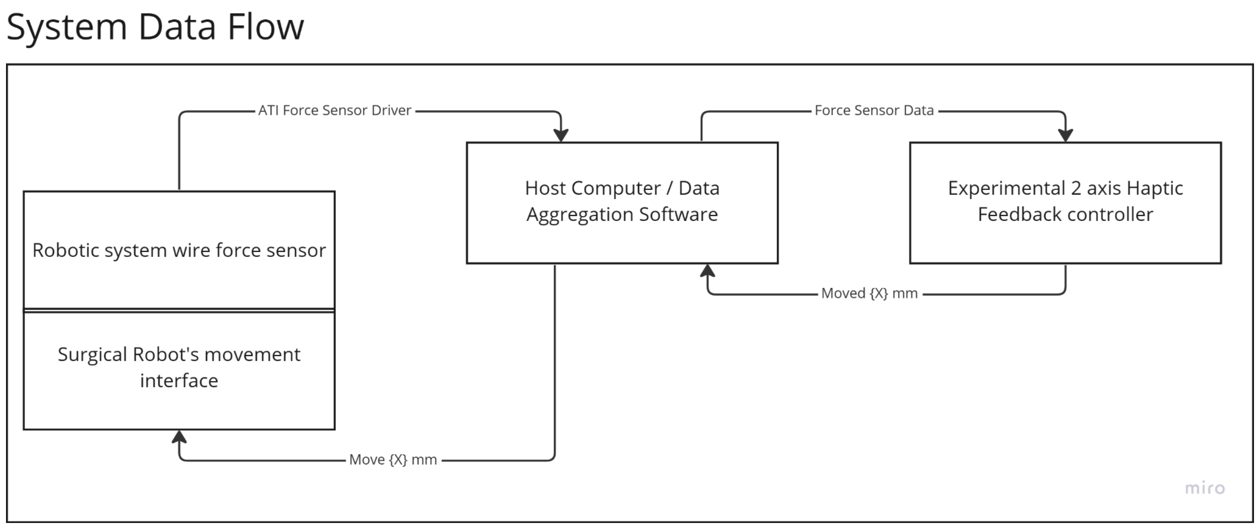

System Data Flow
ATI Force Sensor Driver
Force Sensor Data
Host Computer / Data Aggregation Software
Experimental 2 axis Haptic Feedback controller
Robotic system wire force sensor
Moved {X} mm
Surgical Robot's movement interface
Move {X} mm
miro

Figure 11: data flow throughout the whole system from controller to external robotic platform

The force sensor is to be passed from the sensor onto the PC interface which then passes it down to the controller solution, thus the same force is applied on that side of the system. The user's movement is then sent back through serial communication to the PC interface which then translates the movement data into steps that the robot needs to take to replicate the movement.

2.4. Client Side Software Implementation

2.4.1. Client Side data flow

The client side has an open loop control. It reads ATI force sensor data from the software side, then the data is obtained directly from serial output to the Python script. The corresponding forces or torques are converted into the correct voltage magnitude. After conversion, the designed haptic handler applies it to the existing guidewire handler.

PC Interface Data Flow

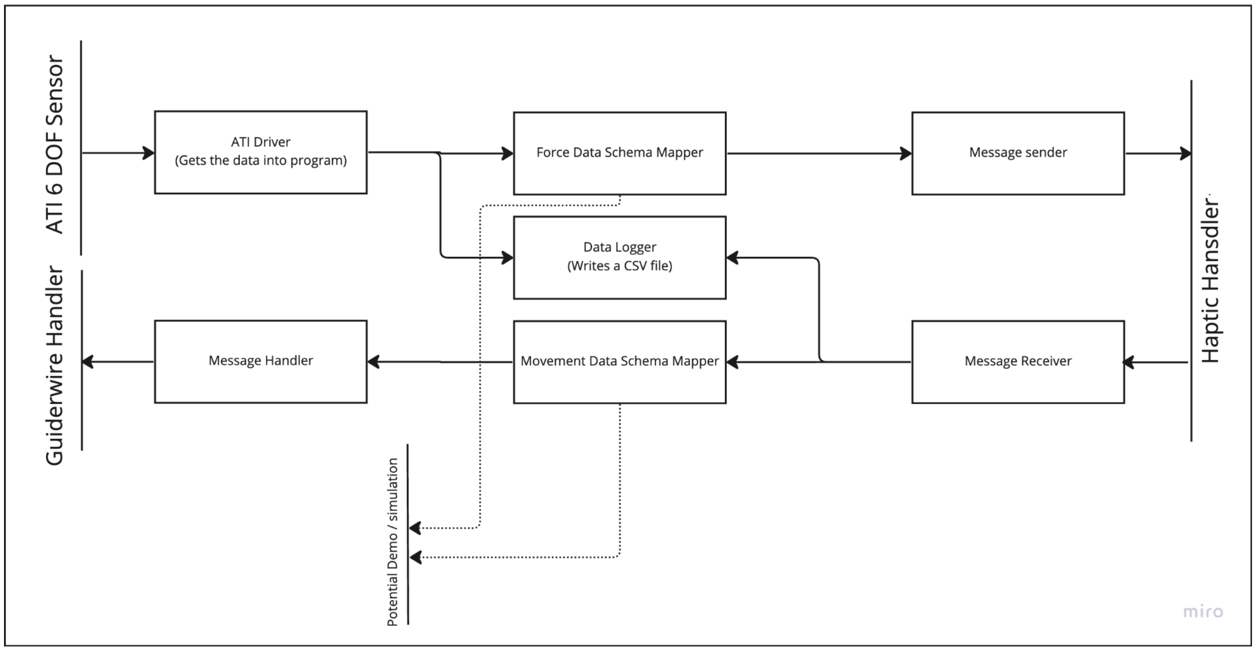


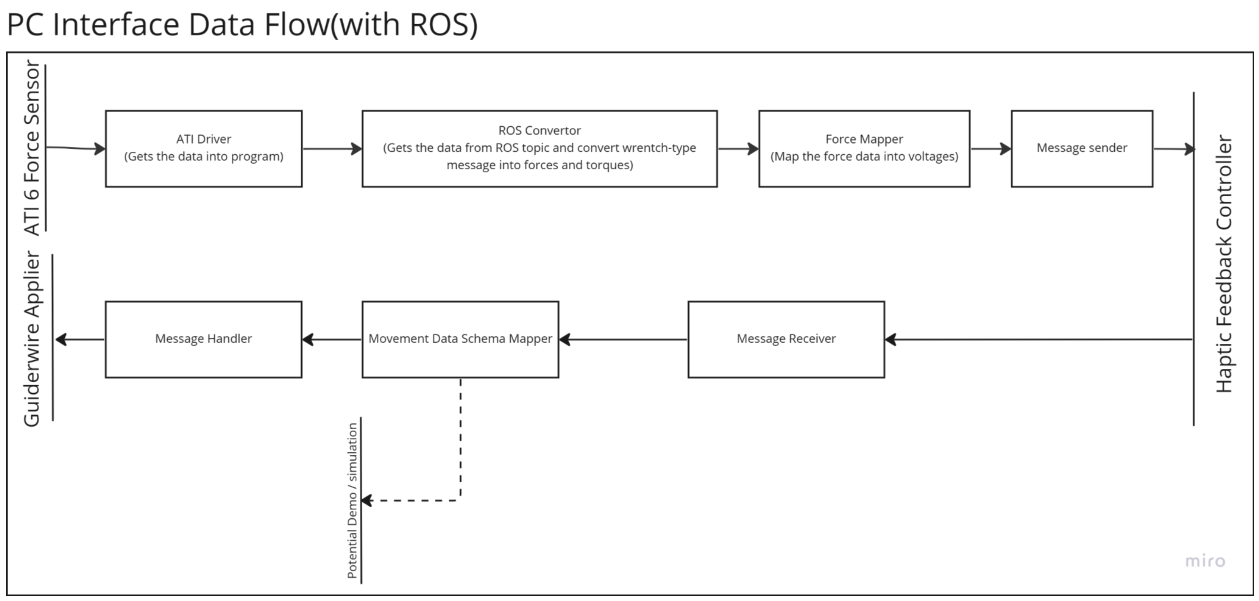
Figure 12: data flow diagram of the PC side interface without ROS implementation

The client side will first grab the force sensor data, in both linear and rotation from the ATI 6 Degree of freedom (DOF) sensor driver into the program and then log them with the corresponding schema mapper.

Another issue that can arise when reading and writing to the same serial port as the buffer overflows. This occurs when there is too much data being sent to the receiving device and it cannot process it quickly enough. This can cause the data to be lost or overwritten, leading to errors or delays in the system. The whole process has been written in python with the implementation of pySerial, designed to read and write directly from the system serial port, which when it's performing real-time communication, will hold a long delay as the system takes time to respond to the reading and writing command in order. The operating system reading with the specific size of buffer waits until the buffer is cleaned to do the next step, which leads to some delay of packet transmission that will cause signal mis-overlapping due to the setting baud rate. This happens because the transmit and receive lines are physically connected, meaning that data can be sent and received at the same time. To mitigate buffer overflows, the channel has been constructed properly to manage the data buffers. This involves ensuring that the buffers are large enough to hold the data being transmitted and that the receiving device can process the data promptly, which the following implementation of ROS will also help in avoiding buffer overflow through a single line communication.

2.4.2. ROS Implementation/exploration

While it is possible to read and write to the same serial port, it is important to take precautions to avoid data collisions and buffer overflows. Using proper synchronization techniques and managing the data buffers effectively, it is possible to ensure reliable and efficient data transfer over a serial communication line. In the following paragraph, it is going to mention the solution using ROS inside the program to separate the other channel handling the ATI force data, and the division of 2 serial ports handling reading and writing data from the CDC output to the STM32 microcontroller.



PC Interface Data Flow(with ROS)
ATI 6 Force Sensor
ATI Driver (Gets the data into program)
ROS Convertor (Gets the data from ROS topic and convert wrentch-type message into forces and torques)
Force Mapper (Map the force data into voltages)
Message sender
Haptic Feedback Controller
Guiderwire Applier
Message Handler
Movement Data Schema Mapper
Message Receiver
Potential Demo / simulation
miro

Figure 13: data flow diagram of the PC side interface with ROS implementation

As it can be told from above, the ATI driver is going to be subscribed from the Robotic Operating System and be converted inside ROS into target force and torque data mathematically, and be sent for force mapper with an appropriate regression model to translate from applied force or torque to target voltages.

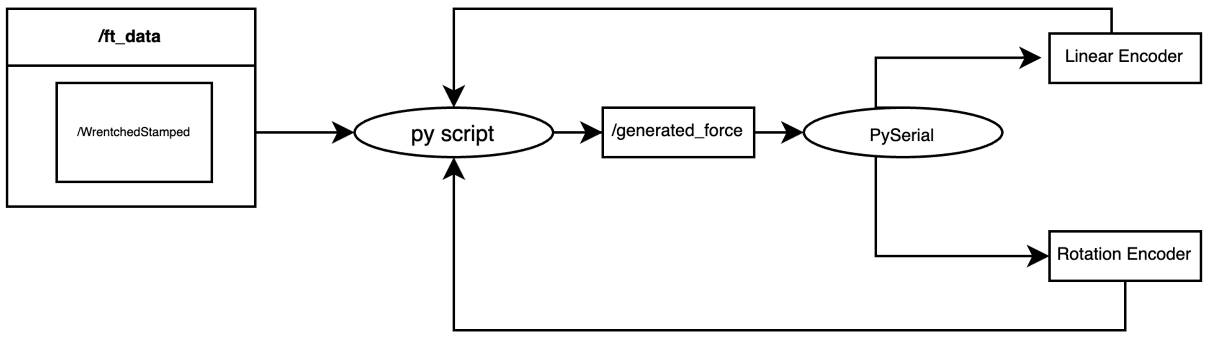


Figure 14: data flow diagram of the ROS side

Scripts are going to be run in ROS, including reading and writing through pySerial. As long as roscore takes response when receiving the corresponding data from a specific topic, /fr_data, the conversion model is going to grab the wrenched value indicated positional data from each axis out from ROS serial into python and handle the following communication inside pySerial. A new ROS node has been created that subscribes to the sensor data topic and publishes the data to a new topic. This data is then used by other nodes in our system for various applications. By plugging the USB-to-serial adapter and connecting it to separate ports on the laptop, less data flow through the same line allows the system to effectively transmit the data between systems. By utilizing ROS's modular approach and separating the ports for reading and writing, it is able to ensure reliable and efficient data transfer between the different components.

2.5. Firmware Implementation

The firmware, found in the GitHub repo at the first reference link [1], is written in C++ utilizing the PlatformIO plugin for Visual Studio Code for compilation in the Arduino framework. It was also responsible for pulling in the proper dependencies that the project relied on. “SimpleFOC” is used for closed-loop torque control based on the position data of the encoders; “GCodeParser” is used for handling GCode input sent to the device as 2 core floating point numbers used to drive the motors. To interface with and initialize the load cell ADC chip, the HX711 module is utilized. Additional drivers were built on top of these to handle calibration and the main control of the controller.

2.5.1. Data flow in the firmware

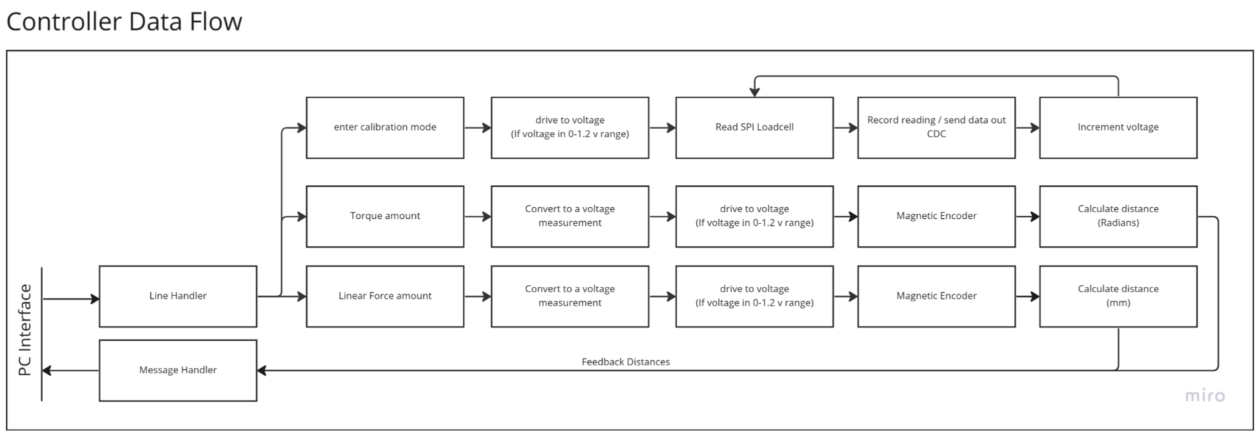

Controller Data Flow
PC Interface
Line Handler
Message Handler
enter calibration mode
drive to voltage (If voltage in 0-1.2 v range)
Read SPI Loadcell
Record reading / send data out CDC
Increment voltage
Torque amount
Convert to a voltage measurement
drive to voltage (If voltage in 0-1.2 v range)
Magnetic Encoder
Calculate distance (Radians)
Linear Force amount
Convert to a voltage measurement
drive to voltage (If voltage in 0-1.2 v range)
Magnetic Encoder
Calculate distance (mm)
Feedback Distances
miro

Figure 15: The internal data flow of the controller data management

Inside the controller firmware, there are 3 main functions of operation, the message handler, the message sender, and the calibration/validation modes. The core objective of the firmware is to be able to handle sending and receiving the data while keeping up with the FOC control loop calculations is time critical. Once the line is received on the input serial port it is then parsed out of the G code formatting of ‘F R0.000 L0.000’ Where the ‘F’ denotes that you are applying a force on the ‘L’ linear and the ‘R’ Rotational axis of the device. Note that the precision of the floats limits the rate of parsing. This float is then taken and converted to a voltage by the calibration curve produced in the routine that runs in the first state of the device startup, disused in a future section. This value is then validated to make sure that it is within an operational voltage range. The target is then applied to the motor control that then drives the control loop to achieve said target.

2.5.2. Control loop

The firmware operates on a control loop that cycles between 3 main states with 2 optional states. The states are INITIALIZATION, CALIBRATION, IDLE, SENDING, RECEIVING / PARSING, and COMPUTING PARAMETERS. Each state only has one core operation, this was deliberately chosen to ensure that the read and write would be alternating and not prevent the FOC parameter calculations from lagging as a result.

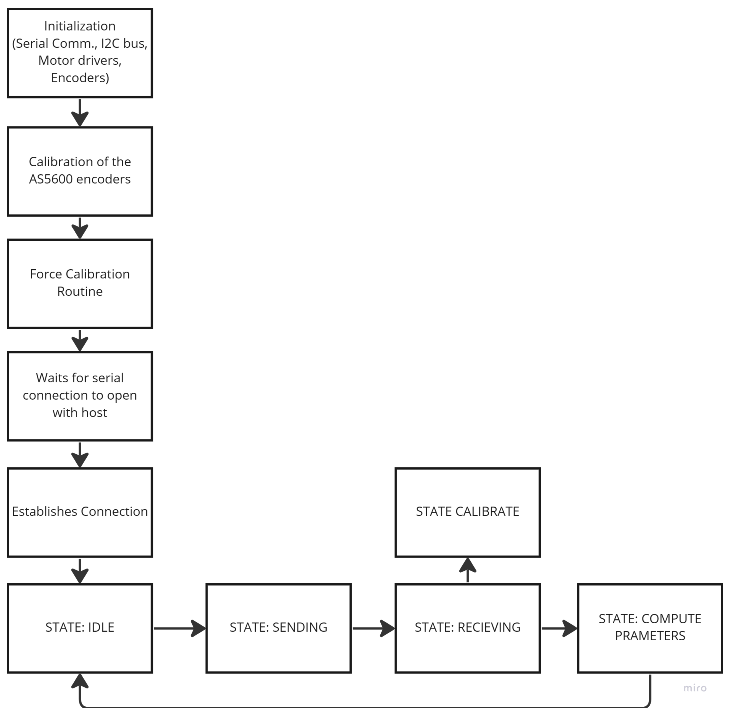

Initialization (Serial Comm., I2C bus, Motor drivers, Encoders)
Calibration of the AS5600 encoders
Force Calibration Routine
Waits for serial connection to open with host
Establishes Connection
STATE: IDLE
STATE: SENDING
STATE: RECIEVING
STATE CALIBRATE
STATE: COMPUTE PRAMETERS
miro

Figure 16: Underlying control loop for firmware

You can break out of the calibration mode if a ‘C’ command is entered into the receiving serial connection. This mode will initialize the load cell at the back of the device and run the routine. Then it will return to the operating control loop and start sending out the movement data and looking for input messages.

The use of 2 serial connections to the host interface was done for the continuity of being able to send and receive data in a full duplex configuration to and from the controller. In initial designs, both actions were done over the CDC serial port. This proved to be overwhelming to the operation of the port as well as significantly more difficult to debug messages going both ways. Another optimization this affords is the ability to reduce the message sizes and flags needed to denote each message header. Speeding up the rate that the device can parse the input strings and apply the forces to the controller.

2.5.3. Data Parsing

The data schema was implemented in the style of “G”. This provided a uniform framework that has been tested and vetted on other similar systems that are designed to take force or positional elements and translate them into locations or forces. Sending messages are in the format of “M L0.0 R0.0”; the “M” denotes movement in this string, followed by the “L” and “R” for the absolute position on the rotational and the linear axis as read by the absolute encoders. The encoder positions are being sent to further reduce the amount of the floating point math done on the device by finding the difference between the last point and the current point. Sending it in this format also reduces the jitter of the devices and ensures that even if there are messages lost in transmission the host interface can still reflect the movement accurately.

The messages from the host interface down to the device commanding forces can either take a format of “F R0.00 L0.00” or “V R0.00 L0.00” where “F” stands for passing a force in mm/N*m on the “R” Rotational” or the “L” linear axis respectively or “V” denoting passing through a voltage to drive the motors too. If a string is parsed without one of the 2-axis values it will then command the motors to be off until the next valid message is received.

2.5.4. Calibration Method

To calibrate the devices two axes, 2 load cells, and amplifiers were employed. For the linear axis, the load cell was placed at the end of the device. In the initial stages of a startup, the device will drive the linear rail into the load cell starting at 0 V as the target voltage and increasing this up to 1.5 V, this is where we observed a significant drop off in terms of forces torque the motor was able to provide. At each interval, the force and voltage pair is recorded as well as a printed tab-delimited to the console. In the initial stages, this data was saved and regressed using linear and polynomial models to find the best fit based on the current status of the system.

2.5.5. Data Post Processing

After the data was collected the points were put into Microsoft Excel where they were modeled. Data cleaning was done in the form of taking the absolute value and offsetting the values to have an intercept of 0 g of force measured when 0V was applied to the system to ensure consistency between collection cycles. This data was then fit to linear, and quadratic functions over the range to determine the relationship between the voltage and the forces. As a result, some of the data range was trimmed in to make the model fit linearly as well as cut down on the number of computations without cutting much of the model's accuracy.

2.6. Testing Procedures

2.6.1. Calibration accuracy

In the evaluation of the calibration regression accuracy, a test was conducted using static set force points and then the output resulting force on the load cell was measured. This allows for the creation of an accuracy curve over the working range of the device when static numbers are implemented.

2.6.2. Accuracy testing

For accuracy calibration the 2 load cells used to calibrate the device were leveraged. Tests were conducted on both axes. In likeness, one load cell was designated the input device and the other was the output force measuring device. A mode with the g-code “C G” was run that let the tester apply a force to the input load cell, the controller would measure the input and then apply force using the voltage calculated through the calibration regression. Thus, different measuring points over the working range of both axes were able to be measured.

2.7. Full system integration

2.7.1. Interfacing with Existed System

The integration of a haptic feedback controller into an existing guidewire feeding system is a significant advancement in the field of surgical procedures. The implementation of this system will require a high level of precision and accuracy to ensure successful outcomes for patients. The haptic feedback controller will be responsible for providing real-time feedback to the surgeon to help them navigate and manipulate the guidewire with greater accuracy.

To achieve this, the force sensor will be placed inside a container mounted on the actuator to provide accurate force measurements in different directions as the guidewire is fed into the tissue. The integration process will require a coordinated effort between the different parts of the system involved in the project, including those responsible for the haptic feedback controller and those responsible for the guidewire feeder.

The data flow inside the system will be crucial to ensure open-loop control and accurate feedback. The force sensor will measure the resistance forces experienced by the end effector and relay this information to the haptic feedback controller, which will then convert the data into a format that can be applied by the haptic handler to the holder. This will help to ensure that the surgeon can receive the necessary feedback to navigate the guidewire with precision and accuracy.

In summary, the integration of a haptic feedback controller into an existing guidewire feeding system represents a significant advancement in this surgical platform. The coordination between the different parts involved in the project will be critical to ensure the success of the whole system. With accurate force measurements and open-loop control, this system has the potential to minimize the risk of complications and maximize the chances of successful surgical outcomes for patients.

3. Evaluation and Validation

3.1. Mechanical Friction/realization

The largest limiting mechanical factor of this or any other mechanical controller is the presence of unwanted friction. This issue is especially important for this controller as neuro-interventional procedures generate very small levels of forces, so any friction in the system attempting to replicate these forces will be exponentially felt. The first few prototypes produced had significant plastic-on-plastic contact, resulting in very high levels of felt mechanical friction on both the rotational and linear elements. To avoid these issues in the final prototype, a specialized low-friction linear rail was implemented into the system, and model tolerances were increased. While the resulting final prototype had significantly less friction than any of the previous iterations, the small friction left was still significant enough to interfere with the low-level force generation in the controller and have an overall negative effect on the forces produced. With more specialized parts with tighter tolerances, this friction could be reduced to undetectable amounts but given the nature of our controller being a prototype and simply a proof of concept, these levels were deemed acceptable for the initial testing.

3.2. Calibration of the system

3.2.1. Linear Calibration

Data was collected throughout 15 trials to evaluate a best-fitting model to convert the target voltage set onto the device and torque output. This was done by setting the target voltage on the controller and then having it report the force from the linear load cell produced by the set voltage.

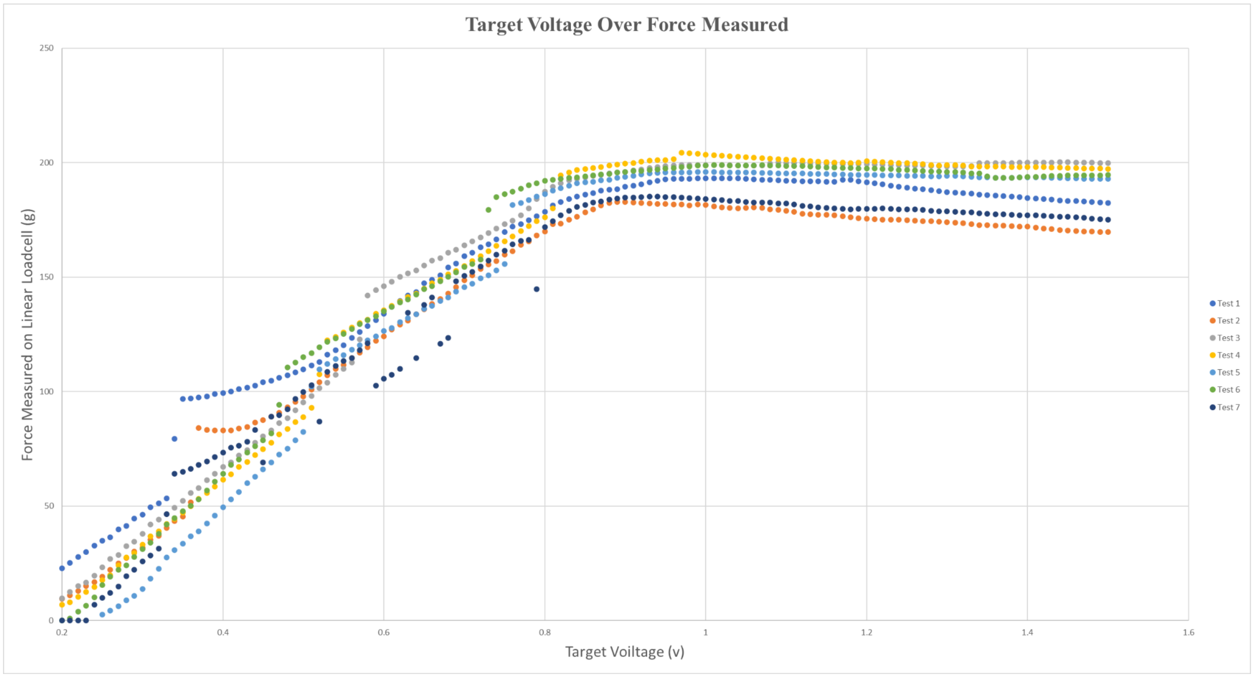


Figure 17: Linear Calibration force curve tests 1-7 plotted over target voltage

Some of the variation in the results can be attributed to the overall temperature of the system. As concurrent tests were run the force produced decreased. This is justified in the mechanical properties of the motor as the heat increases the normal resistance of the motor changes therefore different currents are produced.

From this collected data a linear approximation was produced over the range of 0.0v to 0.85v as this was our operating range of the device. In addition to the amount of force produced, flat-lining beyond that point. This relationship was the following:

$$\text{Voltage} = 0.0035382 * \text{Force} + 0.2012702$$

Over the range this equation had an R^2 value of 0.9989 thus the model was determined to be a good fit for the mean tests. This is the equation that was then implemented into the controller's firmware and used to reproduce forces based on the host machine commands.

3.2.2. Rotational Calibration

The rotational axis was calibrated in a similar fashion to that of the linear axis outlined above. A custom rotational collar was developed for the 3D-printed shaft of the motor. This then allows us to mount a load cell that was long enough such that the end piece intersected the ground, transferring the force into the load cell and allowing it to be read. The rotational data was collected over 8 runs incrementing the target voltage and measuring the resultant force in grams of force. This produced the following voltage-to-force curve:

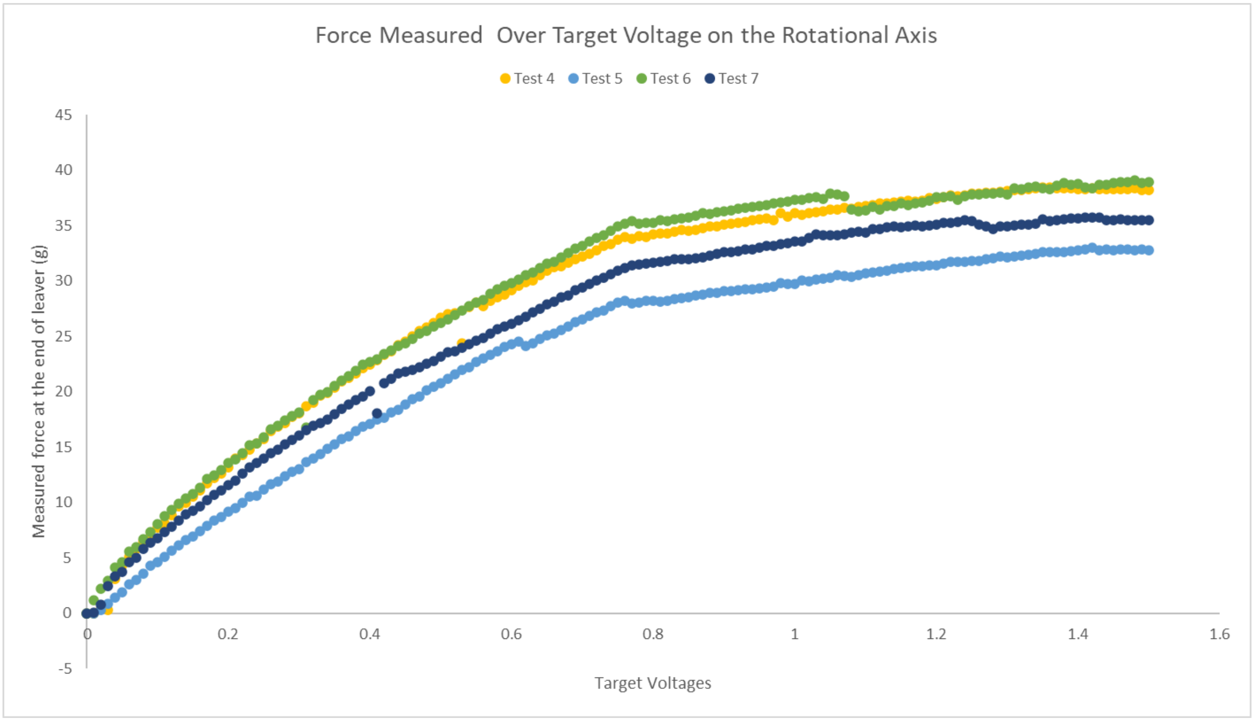


Figure 18: The Voltage to Force curve collected as a result of the rotational axis

The variation of the data can also be attributed to the thermal properties and changes to the motor over time as different voltages in turn were produced different amounts of current and therefore measurable forces consistently over the course of the test runs. The data collected was then modeled using a quadratic function as follows:

Voltage = 0.000364 (Force^2) + 0.0127*Force

This model fit the data with an R^2 value of 0.9988 and thus was evaluated to be a good fit for the voltage to force relationship of the device.

3.3. Evaluation of the force reproduction

3.3.1. Rotational results

In order to test the accuracy of the implemented equations referenced in section 3.4.2 a similar internal to the controller test was run in the firmware. The test stepped a range of 0.0 g to

40.0 g, as this was determined by the linear model to be the equivalent working range, in 0.1 g increments, and measured the applied force to the load cell after each step. This data was then gleaned from the logs to produce the following results:

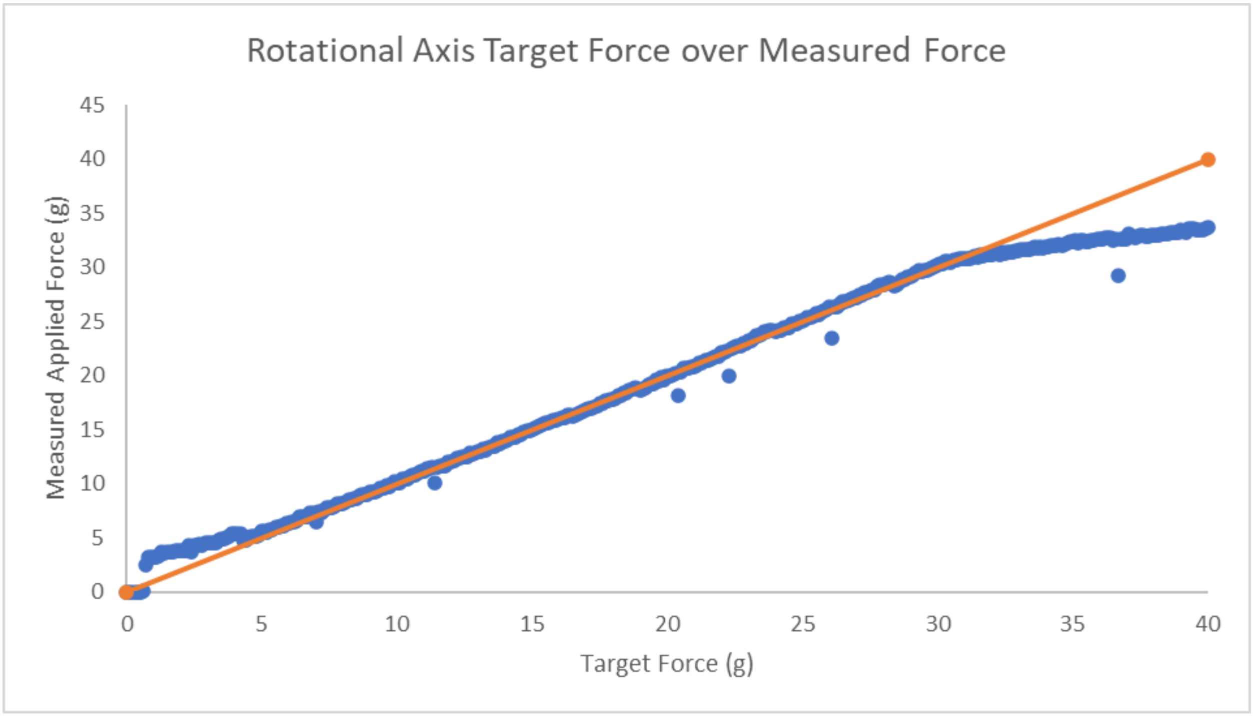


Figure 19: Measured force in grams over target force

The figure above shows that within the commendable range of 5 g to 30 g the error in reproduction is small and greatly fits the model that was applied for the axis. When looking further into the errors consider the following graph of the percent error:

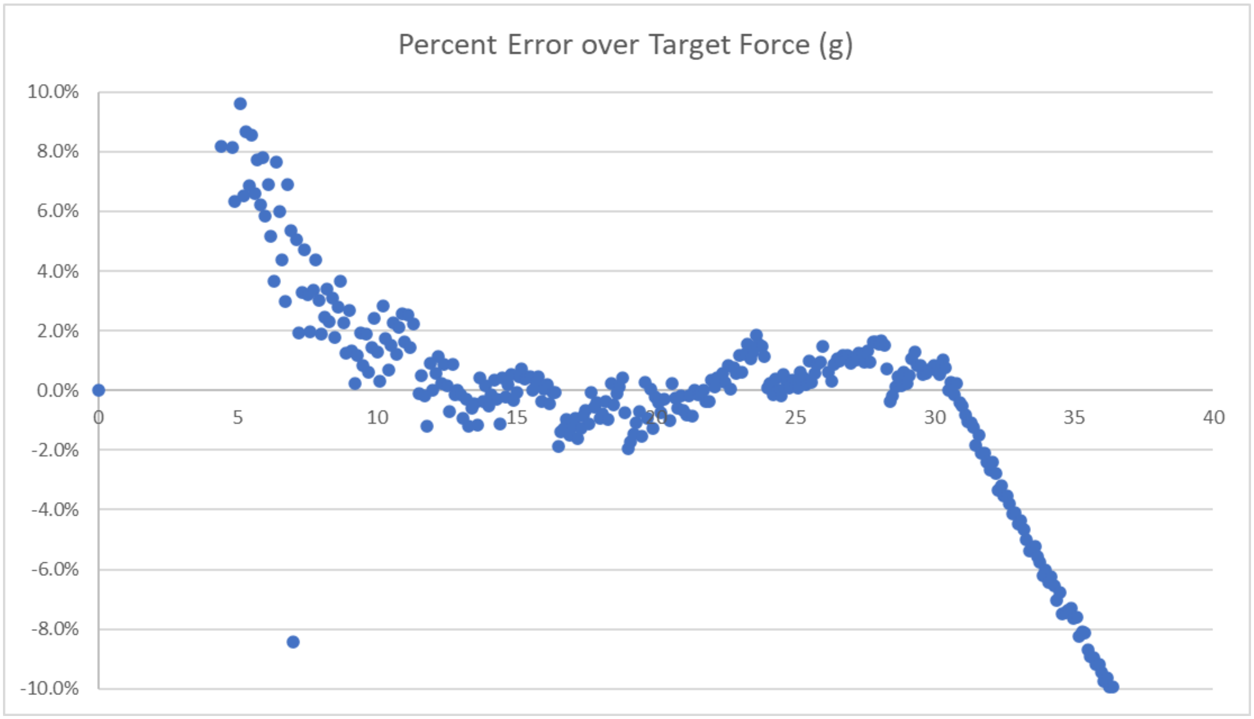


Figure 20: Percent error over target force range

This graph shows the relative error to force intended on being produced. This highlights that within the range of 10g to 30g of a target force on the system, the force was produced and measured within +/- 2%. Outside of this range, it is assumed to be mechanical properties that had influence over the system. This is the nature of the inertia of the system as well as the internal friction of the system as a whole.

3.3.2. Linear results

Additionally the rotational axis was measured in a similar fashion, a target force was applied to the subsystem and it was then measured by the load cell at the end of the device. The following data were collected on the accuracy of the device:

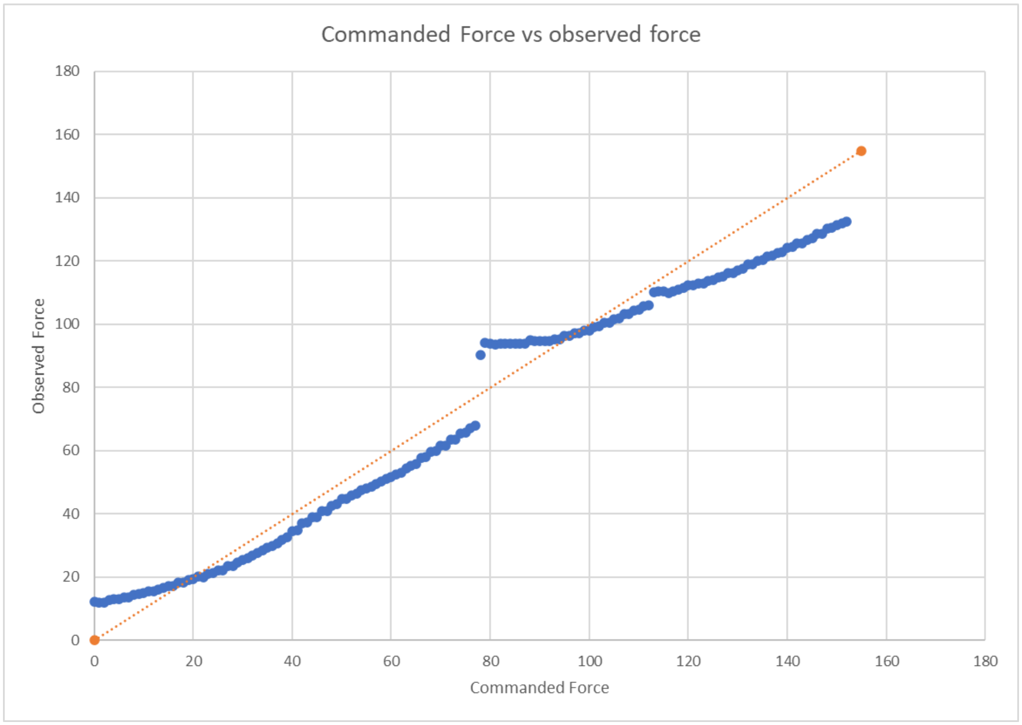


Figure 21: Commanded Force over Force measure in grams

This shows that the force reproducibility is largely linear across the system. However, there are multiple deviations in error from the ideal slope of 1 that is graphed in the figure. The deviation comes around 80 g of the force commanded where the system jumps up. Speculation has this large jump attributed to the teeth meshing together to transfer more force into the system at higher voltages. Overall the system maintains consistent linear patterns that could be modeled more over time over a broader range. In terms of the percent error, the following figure was produced:

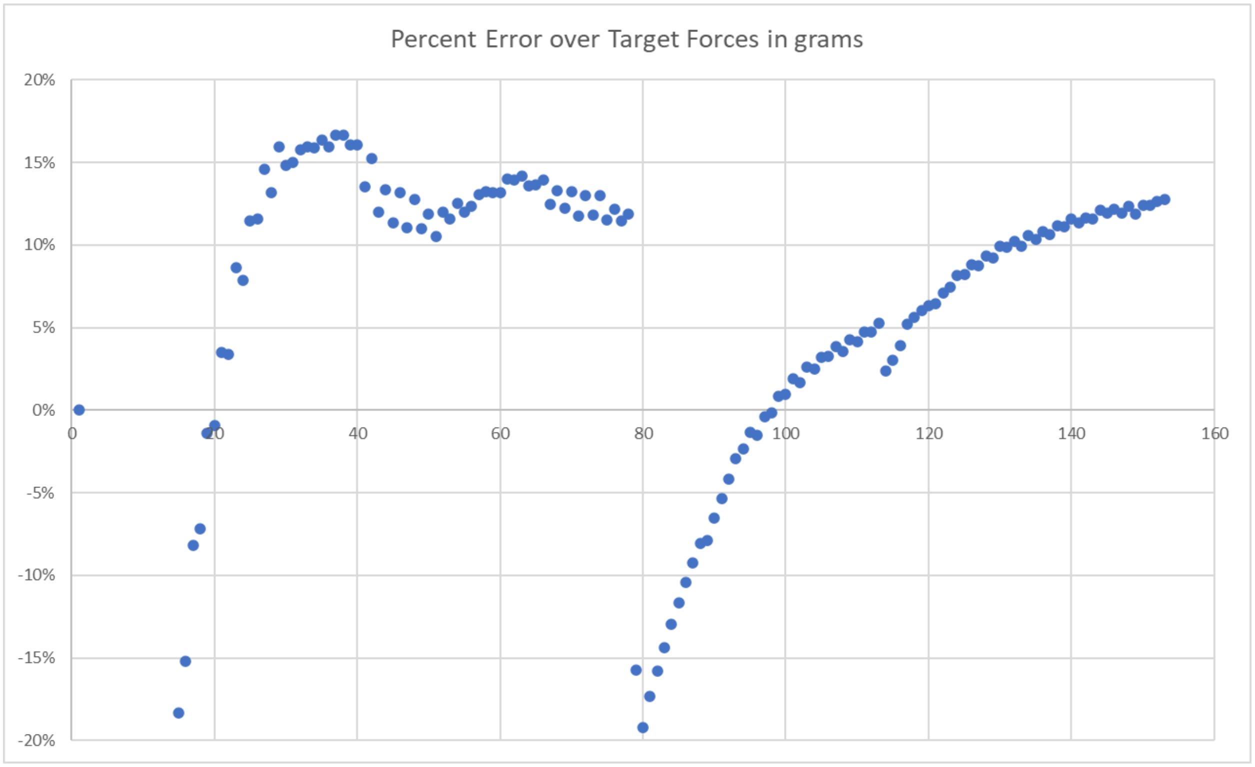


Figure 22: Percent error of the linear axis system over a range of target forces in grams

This also shows a divergent line between 0g to 80g and 80g - 150g. This could be attributed to the mechanical system. Overall the range that we were focusing on in this prototype was for micro force / small force reproduction. Over the first range, the device saw a 12% positive error on each of the target forces. In future iterations of the model, this could further be reduced by modeling the inertia of the rail system as well as reducing the overall friction of the device.

3.4. PC to Controller Performance

3.4.1. "Real-time performance"

Ensuring the accuracy and reliability of the system requires that the program's converted force performance and the load cell reading are in real-time. To test the system's performance, forces ranging from 0N to 2N were generated and recorded on the graph after being converted to grams using newton-to-grams conversion for analysis. The relationship between the converted force and the actual reading is almost linear and proportional, with a 1:1 ratio, as demonstrated by the regression line. Over 200 data points were generated repeatedly, and the resulting graph shows the average proportional relationship collected over time. The system's performance is satisfactory, as evidenced by the linear regression line, which fits the data with a high R2 value of 0.9986 as shown below:

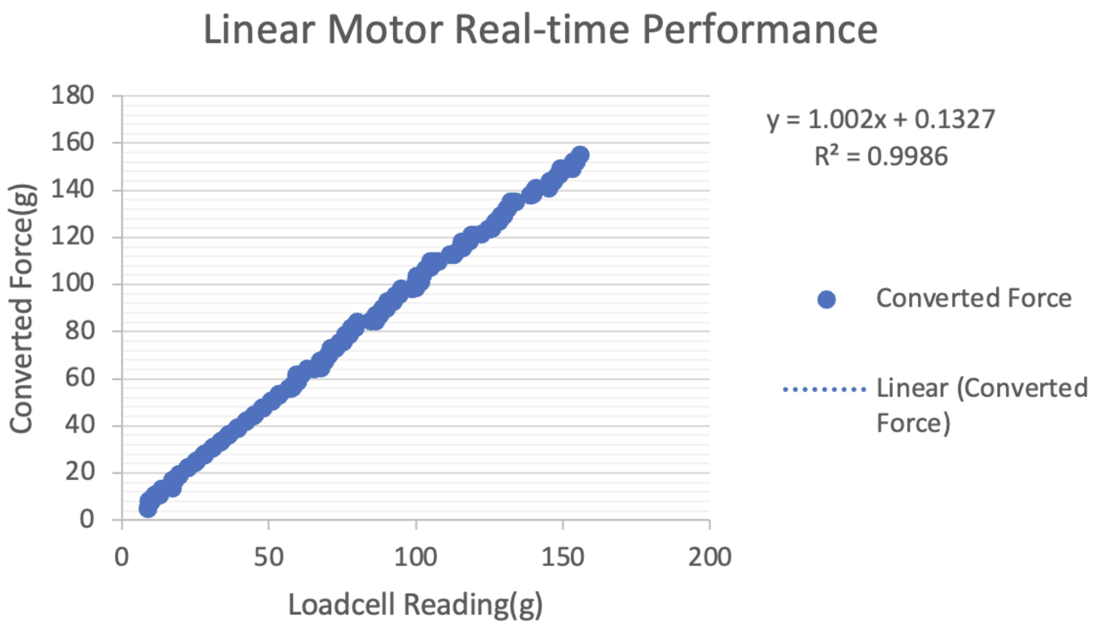


Figure 23: Averaged recorded real-time linear motor performance vs. actual

The error during the real-time response with the linear motor is shown below:

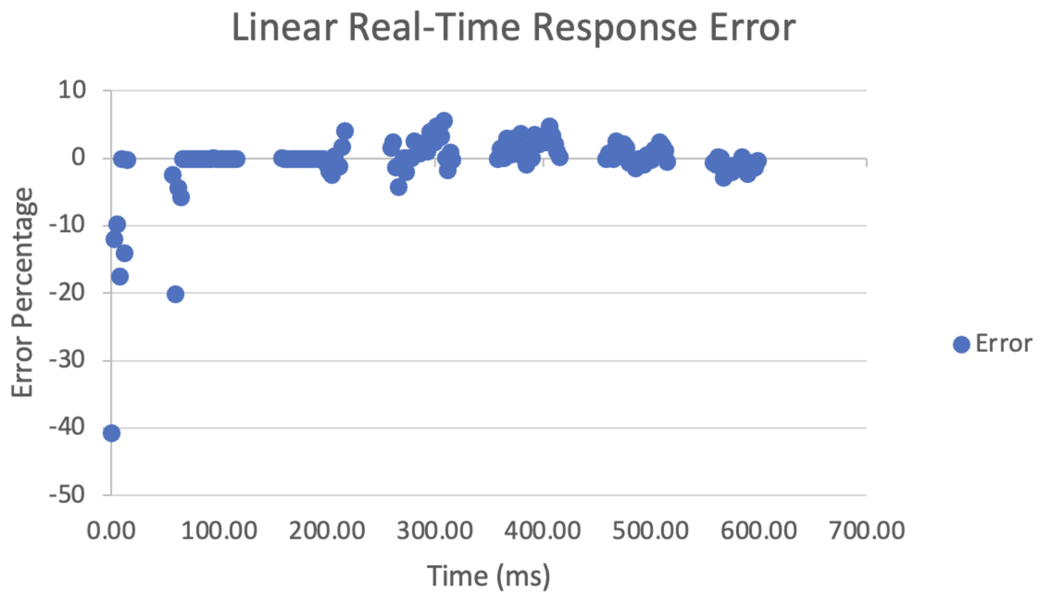


Figure 24: Averaged recorded real-time error of linear motor vs. actual

The errors in the system are observed to be clustered within a specific time interval. This clustering is due to the system's integration with ROS, where the updating rate for each cycle's reading and writing is set to 125ms, which is equivalent to 1/8th of a second as per the sleeping rate. The error percentage averaged over a series of trials conducted within a short time range is presented in the plot above.

The measurement error or deviation is the difference between the readings obtained from the load cell and the values generated by the device. This error can be caused by various factors such as sensor drift, calibration errors, or noise in the signal. The presence of any of these factors can lead to differences between the actual value of the force and the measured value. In the tests conducted above, the device was reading Newton as the linear forces input from the ATI force sensor. However, the program was required to convert these readings to grams for validation purposes, which could lead to a larger error than the actual performance during conversion.

The conversion process from Newton to grams involves a mathematical procedure that introduces additional sources of error. Any deviation in the conversion factor or the actual value of the force could result in a significant difference between the actual value and the measured value. Furthermore, other factors such as environmental conditions, equipment malfunction, or operator error could further affect the accuracy of the measurement.

Therefore, to minimize measurement error, it is essential to calibrate the device regularly, ensure that the load cell is correctly installed and aligned with the force being measured, and use appropriate techniques to handle and process the data. By employing these measures, it is possible to obtain more accurate and reliable measurements, which can enhance the performance and effectiveness of the system.

As indicated by the graph, the errors in the measurements lie between -10% and 10%. To ensure that the measurement error is minimized, regular calibration of the device is carried out to ensure that the load cell is properly installed and aligned with the force being measured. In the past, there was a significant deviation from the line, which was attributed to an incorrect conversion where the reading unit was in grams while the force sensor produced values in Newtons.

During the normal operation of the system, the guidewire handler will receive linear and rotational steps from the haptic handler, while the haptic handler will receive linear force and rotational torque and control the applied forces with converted voltages. However, it is important to note that conversion among forces and voltages can introduce data biases during the mathematical procedure. Therefore, omitting the processing of these conversions could enhance the system's performance from the calculation perspective.

It is crucial to note that the overall real performance of the system when integrated with the entire system will be faster than what is shown in the graph. This is because the graph only captures a snapshot of the performance within a particular time frame, and the system's overall performance is dependent on various factors, such as the efficiency of the software and hardware integration, the accuracy of the sensors, and the consistency of the calibration.

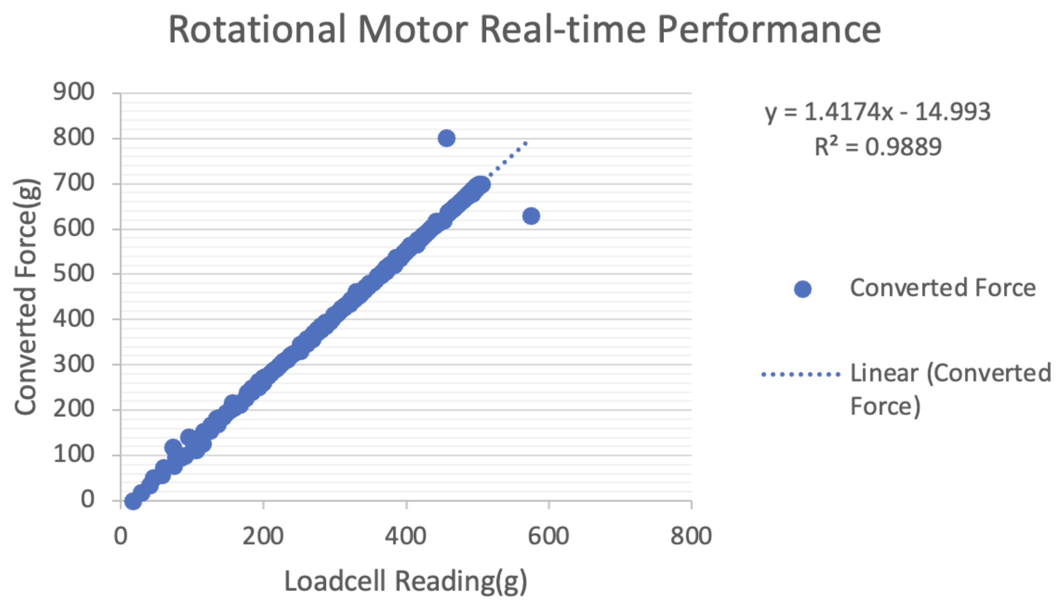


Figure 25: Recorded real-time rotational motor performance vs. actual

For the rotational motor performance, a total of 200 data points were collected over multiple trials to obtain an average representation. The forces generated were found to be in the same range as those generated by the linear motor. However, there were some differences in the measurement process. The direct reading from the ATI sensor was in N*M, whereas the calibration model was in quadratic regression, with measurements in grams. As a result, there was a slight discrepancy between the actual reading and the applied force, with a scaling factor of 0.5 more than the linear performance. Despite these minor mathematical discrepancies, the overall performance of the rotational motor still followed a linear regression with a high degree of accuracy. A statistical measure in a regression model, known as R^2, was used to determine the proportion of variance in the dependent variable that could be explained by the independent variable. The R^2 value obtained for the rotational motor performance was around 0.9889, indicating that the model was able to explain a significant amount of the variation observed in the dependent variable. This level of precision and accuracy is crucial in the context of medical devices, where even minor errors can have serious consequences for patient safety. To further evaluate the performance, the following percentage error is produced:

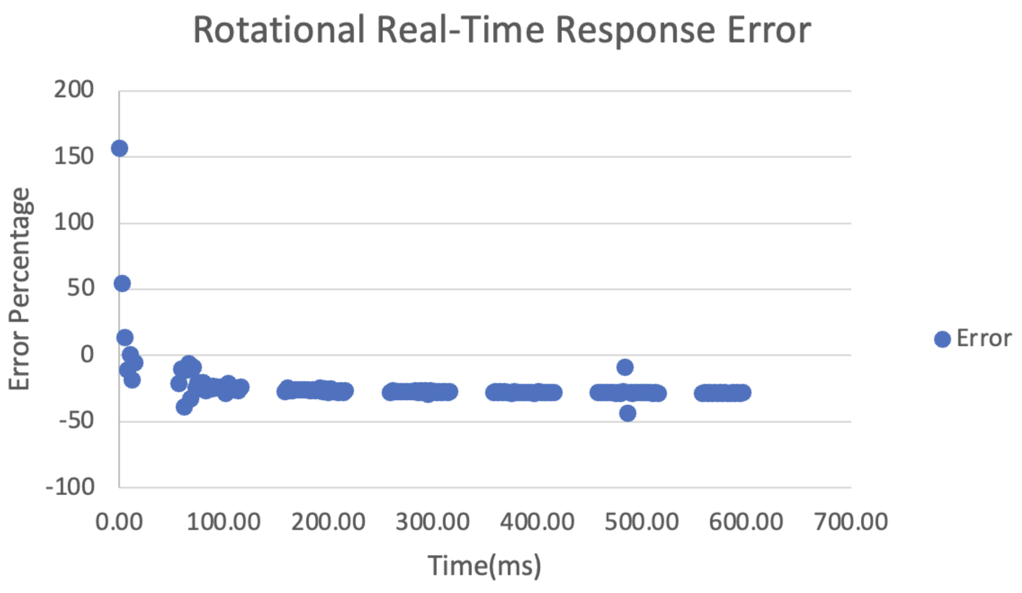


Figure 26: Averaged recorded real-time error of rotational motor vs. actual

As shown above, the error range is between -50% and 0%, which means it has an average of +/-25% from the ground truth. Similarly, mathematical conversions that are rotational use quadratic equations instead of linear ones which can lead to even larger errors if the input variables are measured inaccurately or with low precision. In this case, even small errors in the input variables can lead to significant errors in the final result because the square of the input will also square the errors, which can have serious consequences for the results. Therefore, it is crucial to carefully evaluate the accuracy and precision of the input variables and the conversion process itself to minimize the potential for an error with respect to that. In addition, it is also helpful to validate the results of mathematical conversions through experimental verification, which is how this graph is produced..

3.5. Force application rising time

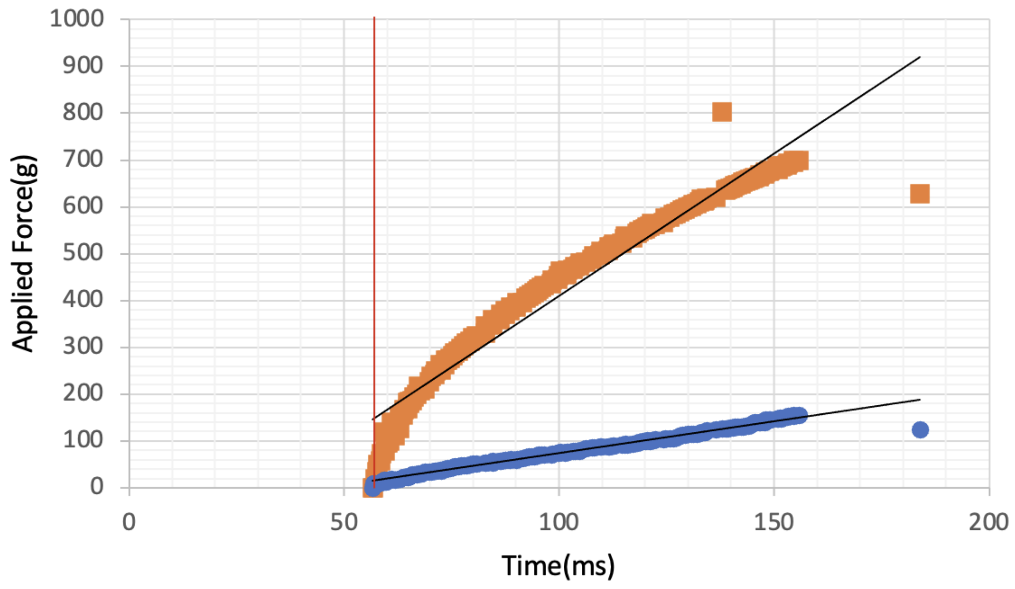


Figure 27: Linear and Rotational Forces recorded over time

As depicted in the figure above, there is no data point shown in the plot for the first 50 ms of the time period. This delay can be attributed to the response time of the system which is the time it takes for the system to provide a response to an input, which is an important aspect to consider when evaluating the functionality of real-time answering. To ensure that the system can operate in real-time, the updating rate of the ROS (Robot Operating System) has been adjusted multiple times to determine the most appropriate rate that can minimize packet loss and communication lag.

The plot shows two lines, with the orange line representing the linear performance and the blue line representing the rotational performance. The black line represents the trend line of the entire set of data. The graph indicates a short delay at the beginning, which represents the rising time for effective data. The rising time is observed to be low, at approximately 10 ms, which is an indicator of the system's ability to respond quickly to input. To ensure that the update rate is fast enough to capture the rising time, the program controlling the firmware on the haptic handler has been modified to enable load cell readings to be executed as quickly as possible. As a result, the entire system can update data at a rate of up to 115200 bits/s in one reading cycle, which is significantly faster than the default rate of 9600 bits/s.

3.5.1. Bottlenecks of the implementation

One aspect that inhibited faster real-time performance than 8hz was the processor's ability to parse the lines of G-code at a rate that fed the system to perform and apply the field-oriented control at an acceptable rate. This thus capped the receiving and motor updating rate and response times of the system.

On the ROS side, the previous 200HZ updating rate was changed to match the message handler's rate of 8HZ, which significantly reduced the lagging time of the system, since high *rospy.Rate* can lead to high CPU usage and slow down the overall performance of the system. On the other hand, if the rate is too low, it can result in messages being delayed, or even missed. In the case of the system with haptic feedback, the *rospy.Rate* was initially set to 200Hz, which led to lagging issues due to the system's slow response time. By changing the rate to 8Hz, the system was able to keep up with the message handling and reduce the delay between message publication and subscription. This optimization allowed the system to update data with up to 115200 bits/s in one reading cycle, a significant improvement over the default rate of 9600 bits/s.

By ensuring that the updating rate of the system matched the message handler's rate, we were able to minimize the potential for packet loss and lagging, ensuring that the system performed optimally.

However, it's important to note that the updating rate alone may not solve all the lagging issues in a ROS system. Other factors such as hardware limitations, and message size can also contribute to delays in the whole system implementation.

4. Conclusion

4.1. Future work

4.1.1. Smoother linear rail

As previously mentioned, the largest limiting mechanical factor of this or any other mechanical controller is the presence of unwanted friction. Given the very small levels of forces generated by neuro-interventional procedures, any mechanical friction in the system attempting to replicate these forces will dramatically affect the resulting forces of the controller. Most of the mechanical friction experienced by the final prototype is within the linear rail and gear teeth interaction of the linear motion element. The gear teeth interaction friction can be explained by the use of 3D-printed gears. While very convenient and fast, 3D printed parts have non-smooth surface finishes which can lead to unintended friction in any mechanical system utilizing 3D printed parts, especially in 3D printed gears. To remove this factor of friction, machined gears could be implemented in future designs given their smoother surface qualities and better teeth nesting. The linear rail friction could be avoided by sourcing a more expensive, lower-friction linear rail or implementing a completely different linear control method such as linear motors.

4.1.2. More poles on the motors

The brushless motors are the most significant part of this controller as they are the force generators for the haptic feedback system. In future iterations of this design, it is advised to source brushless motors with high pole counts. Using brushless motors with more poles will only positively impact this controller design as more poles mean finer motor control capabilities (lower torques) as well as smoother operating cycles. Because speed is not an issue in this system, the reduction of RPM with an increase in pole count would not negatively affect this project. Large motors with more poles would decrease the need for mechanical complications and could dramatically reduce the amount of friction generated through the mechanical controller. The only limiting factor of high pole count motors would be cost and the larger footprint, but if implemented correctly they could have a significant positive impact on the controller.

4.1.3. Current (amp) sensors on the motors (better torque control)

One fundamental bottleneck of the implementation is the reliance upon the motor resistance for running constant torque control. The algorithm for the field-oriented control used the motor resistance value to then calculate how much current the motor was getting. Thus, there has been more comprehensive literature on constant torque-controlled systems employing the use of current sensors to be able to track the changing motor conditions in real time creating a more accurate feedback loop and thus would have improved the implementation, especially over time as some of the currents are lost to heat and auditory energy loss.

4.1.4. Communication over a full duplex medium (faster processor)

Another major bottleneck in the implementation was the MCU’s ability to parse out the string GCode commands coming in in real-time. Similar systems that are tasked with parsing and handling large volumes of input often run at double the rate of that of the processor used in this project. Thus communication was limited to a rate of 8hz based on this constraint. Future implementations should take this into consideration employing a processor with a higher clock rate or a dedicated core to parsing and message handling or the use of an off-board coprocessor leveraging a faster communication protocol on the controller level to serve the data to the motor controller processor. This would greatly reduce the load and allow for a higher refresh rate.

4.1.5. Real-time force feedback

Another improvement to the system would come in the form of a sensing solution to report forces in real time. Currently, this implementation operates on the impression that the force to voltage curve on startup is the one truth that does not falter. However, for a long procedure or use of the controller, mechanical and electrical forces change. An additional sensor that would add to a closed feedback loop would serve to greatly improve the force replication accuracy of the device and decouple the reliance upon the initial truth.

4.2. General design considerations

4.2.1. Clutch placement

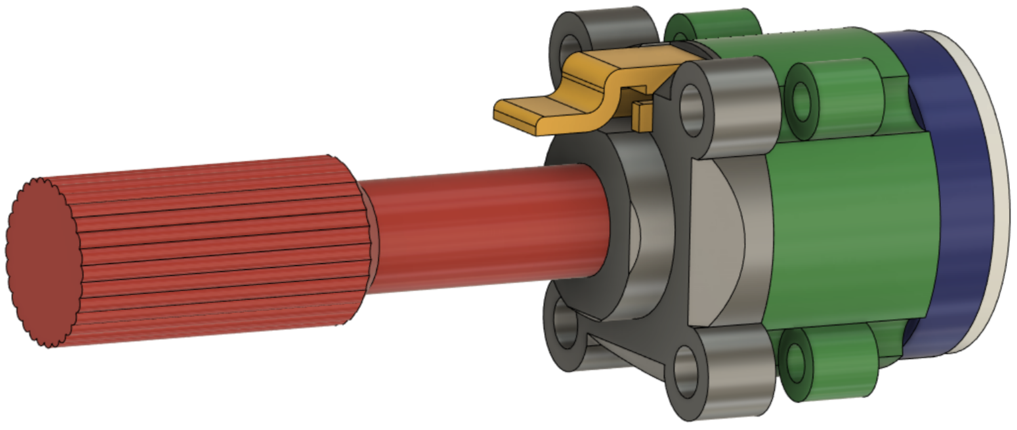


Figure 28: CAD of Clutch Button

The linear range of motion of the controller is approximately 200mm. For a surgeon to fully execute procedures requiring a larger range of linear motion a clutch button design was implemented in CAD and then 3D printed. A clutch mechanism would allow the user to turn off their controller inputs and then “jog” the controller to the desired linear position without sending new position commands to the robot, allowing for infinite motion in both directions. The switch was placed in an optimal position just in front of the control handle interface where a finger from the control hand could easily reach. The other consideration for this design is that this button is independent of the rotation of the handle such that it will always be in the same position regardless of the handle rotation. This removes the risk of the button being positioned in locations that are hard or impossible to reach during a procedure. Though this button is mechanically implemented on the controller, it was never electronically or digitally implemented as there was no robot for the team to test the clutch function on.

5. Broader Impact

5.1. Engineering ethics

This project abides fully by the ethics of engineering as the project aims to use engineering to better society as a whole and nothing else. The objective of This project is to exclusively research and develop new ways of improving patient outcomes in the hospital for very common human diseases and procedures.

5.2. Societal and global impact

This project has a broad vision to advance the field of telerobotic systems as well as further the control of robotic systems, making them more human. With multiple benefits to the surgeon as well as a greater impact on telemedicine. By allowing the operator to still have the feeling of the procedure but at a remote location, remote interventionist procedures will be able to become more common for the professionals in the field. In turn, this will allow hospitals with the proper robotic equipment to provide these operations regardless of the location of human capital resources.

5.3. Environmental impact

As global citizens it is important to consider the resources being used in the design. The prototype constructed uses 2 brushless motors, a custom PCB driver and power supply, as well as 3d printed plastic parts and a steel linear rail. It was the goal of this project to minimize the overall costs of the project in addition to the parts that were being used. For example, the design choice is to reduce switching the linear motion from a PLA plastic solution to a steel linear rail version. Thus the device will be able to stay more consistent with friction throughout the device's lifetime.

5.4. Codes and standards

When utilized correctly and with adequate training, robotic surgery can be safe and effective for carrying out most procedures. To offer a reasonable level of assurance regarding the safety and efficacy of devices for their intended uses, the FDA regulates them like any other medical device. The FDA ensures that manufacturers put in place suitable training programs for both novice and seasoned users. The company providing this controller must prove the 100% reliability of the technologies, show multiple levels of fail-safes, and failure mode analysis models to guarantee safety on the robotics side of the procedure. The controller system proposed in this paper would have to go through the same FDA regulations and approval as every other surgical device as it would act as a critical part of the surgical procedure.

5.5. Economic factors

Compared to other feedback controllers on the market, our solution provides a more specialized experience for neuro interventionalist procedures. Similar devices exist in the capacity of 3-axis controllers, and due to this versatility are much more expensive to operate. The solution proposed in this paper is also not as resource intensive as other solutions only using 2 motors to provide feedback as well. Thus, sourcing and manufacturing should benefit from the reduced number of parts.

6. References:

[1] NeuroIntervention Haptic Feedback Controller WPI MQP 2022-23 (2023) [Source code]. https://github.com/VoidedIceberg/WPI_NHFC_MQP.2022

[2] Mendes Pereira V, Cancelliere NM, Nicholson P, et alFirst-in-human, robotic-assisted neuroendovascular interventionJournal of NeuroInterventional Surgery 2020;12:338-340.

[3] Steval-GMBL02V1. STMicroelectronics. (n.d.). Retrieved March 24, 2023, from https://www.st.com/en/evaluation-tools/steval-gmbl02v1.html

[4] Amazon.com: Drones Brushless Motor Brushless Motor, for Gimbal RC ... (n.d.). Retrieved March 24, 2023, from https://www.amazon.com/DAUERHAFT-Drones-Brushless-Materials-Gimbal/dp/B08S5XMK5J