Motor Assistive Glove

Senior Design Project

In partial completion of
Bachelors of Science

Contents

Chapter 1

Introduction

1.1 Abstract

Every year thousands of people suffering from neuro-trauma, will lose functionality in their motor nervous system. The condition, referred to as peripheral neuropathy, leaves patients fingers weak, desensitized, and with limited mobility. The **Motor Assistive Glove** Project is an investigation of a method to provide immediate increase strength, as well as rehabilitate the patient's own ability to move their fingers. This project asses current exercises in rehabilitation, and looks to construct a glove that is capable of contracting and relaxing a user's hand, based on user input, in order to assist and heal.

1.2 Motivation

Sufferers of neuro-trauma often must spend months, if not years, recovering from any residual degeneration. A large portion of patients must undergo extensive physical rehabilitation to regain the control over the peripheral nervous system. At best every day tasks become painful and difficult to perform, in some cases impossible. This drastically degrades the quality of life that patients have, and limiting their overall personal freedom.

Modern advances in the robotics, electrical, and computer engineering has brought about a new era of affordable, and quality sensors and micro-controllers. Furthermore advances such as 3D printing make manufacturing of prototypes a cheaper and opens a window for more complex designs to be born. By applying methods and processes from these disciplines of engineering there is space for technology to help patients recover faster, from the comfort of their own home, and a more affordable price.

1.3 Objective

This project aims to apply and produce a novel solution to a problem that presents an ailment to thousands of patients a year. While existing research exists in the field, it does not present any tangible results that could be applied in the realm of medical rehabilitation. A tremendous amount of effort has been applied to the subject, the limited variety of prototypes built and theoretical design produced, each presented many shortcomings or missed realizations.

Te final objective of this group is to design a glove that will be able to actuate human fingers from the subtle inputs of a patient, and will operate safely, and quickly. We aim to restore piece of mind, strength, and allow patietns get a grasp on recovery.

Chapter 2

Methods

2.1 System Overview

The system overview seen in figure 2.1 will be developed more during spring quarter but this is the basic concept.

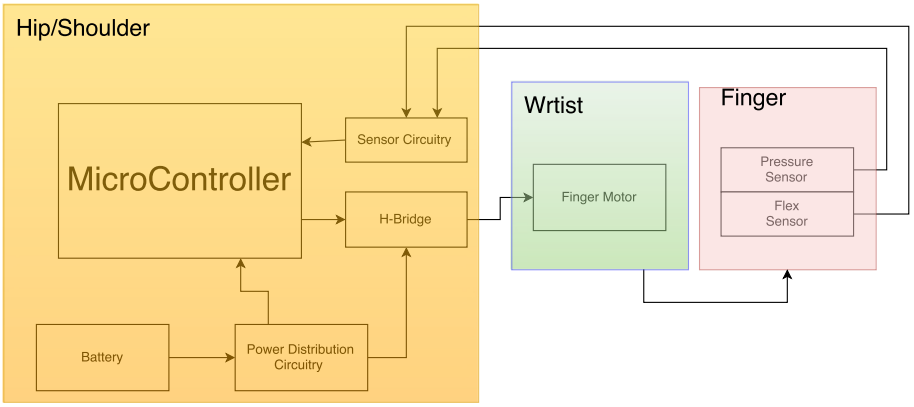

Figure 2.1: this is the system overview. A top level understanding of our system.

2.2 Mechanical Design

2.2.1 Actuation Ideas

There were several ideas that we considered pursuing when we thought about actuating the glove. Since this is an exoskeleton, the best way we found to actuate the fingers is by using a string to pull the finger into a curling position. To return the finger to its resting position we first considered pulling the finger in the opposite direction. However, this would require using a a second motor

to put the finger in extension from the curled position. The problem with the second motor is that it would take up valuable real estate on the wrist where the motors will be mounted. It also poses another problem, cost. If each finger required two motors, you are effectively doubling the motor cost and putting more pressure on your battery life.

After considering these costs we settled on a single motor pulling the finger into the curled position and then using a spring to return it to its resting position. After deciding this we had to decide how we would pull the string. This consisted of two main ideas, a linear actuator and a motor with spooling mechanism. The linear actuator would be the easiest to fit into the mechanical design due to its size and the nature of its actuation. We would not have to change the direction of the actuation of the unit which would reduce time spent on the mechanical design of the unit. The down side to a linear actuator is the time it takes for the movement as well as the price. While looking into linear actuators we found that to get the distance and speed of actuation needed for our project we would be looking at a price one hundred and above. Since we are trying to make a semi low cost product we decided to go with a motor that would use a spooling technique which would cost around twenty dollars per motor which is much more in the price range that we desired.

The spooling technique we settled on was using a spool, which can be seen in figure 2.2, that would reel in the string which would curl the finger while it is being pulled. We designed the spool in Solidworks and then 3-D printed it out of PLA material. The string will run through spool and be held in by the motor shaft. As the motor shaft spins the string will be pulled into the grooves of the spool and the string will be pulled. A picture of the actuation unit is shown in figure 2.3

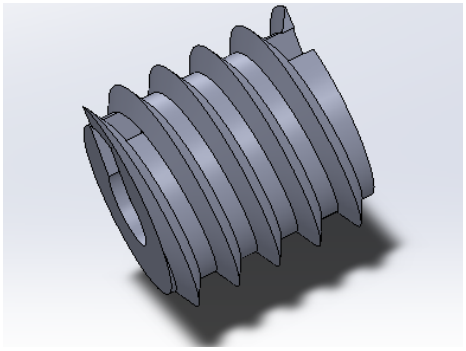

Figure 2.2: Spool to curl finger

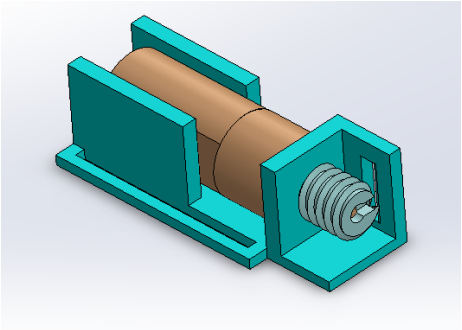

Figure 2.3: Actuation unit will turn spool which will reel in the string around the spool

2.2.2 Mechanical Structure: Hand prototype

After figuring our how to actuate the finger we need a structure in which the string would pull on. To do this we thought of the idea of three rings that would be sewed into a glove. The string would pull on the rings and as they are being pulled, the users fingers will act as the joints of the mechanical structure. Resisting the fingers curl will be a spring over the top of the finger as the finger curls the spring will resist it more. When the user releases pressure from the sensors mounted within the glove, the spool will release and the resistance from the springs and will bring the finger back into its resting position. The three rings have a hole in the bottom to run the string through and then a ring on top for the spring to run through which will return it to the resting position. The spring on top will be mounted onto a platform on top of the hand. The three rings can be seen below in figure 2.4.

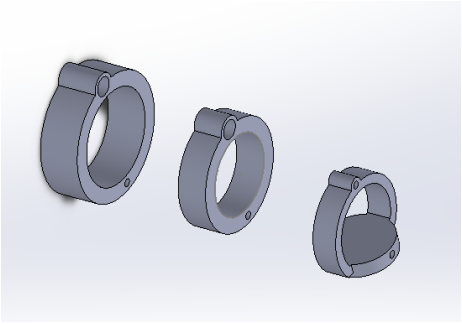

Figure 2.4: These are the rings that will be integrated into the glove in order to pull the finger into a curled position

The rings seen in figure 2.4 are the physical structure and the tip of the finger will make contact with the flat surface on the smallest ring. This surface is flat in order to have the pressure sensor placed on it. The pressure sensor needed a flat surface to act on or it could break or not give us a good reading. the holes on the top of these rings are used to run the spring through so that the finger will return to its original position. These springs will then run into a spring mount over the top of the hand. To design this piece we printed several iterations and adjusted the contour until it fell comfortable to the user when it was strapped to the top of the glove/hand. This design can be seen in figure 2.5.

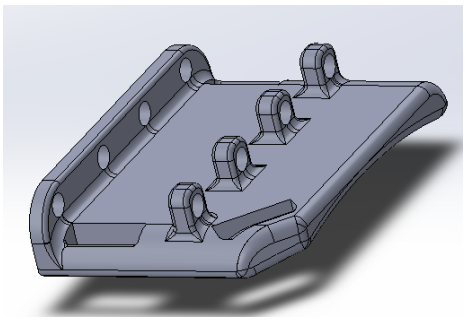

Figure 2.5: The spring mount will be strapped to the top of the hand

2.2.3 Mechanical Structure: Wrist and Forearm

This section will be built and elaborated upon in Spring quarter

2.2.4 Glove Design: Prototype

By the end of winter quarter, which is halfway through the project in total, we wanted to achieve a functioning prototype that used a rough software and hardware design to control a prototype of our glove. This turned out successful since our glove actuated based on a pressure recorded on our pressure sensor. The design on a user's hand can be seen in figure 2.6. For this prototype we sewed the rings into the glove and fastened the sensor to the tip of the tip ring of the rings. The motor and spool mount as well as the spring mount were fastened to the wrist and top of the hand using Velcro. This design will be refined and cleaned up for the final iteration of the project.

2.2.5 Final Actuation Design

The above section went into the details of the original actuation ideas that put together our first prototype. However, after the prototype was formed flaws began to surface almost immediately. The two main problems that occurred

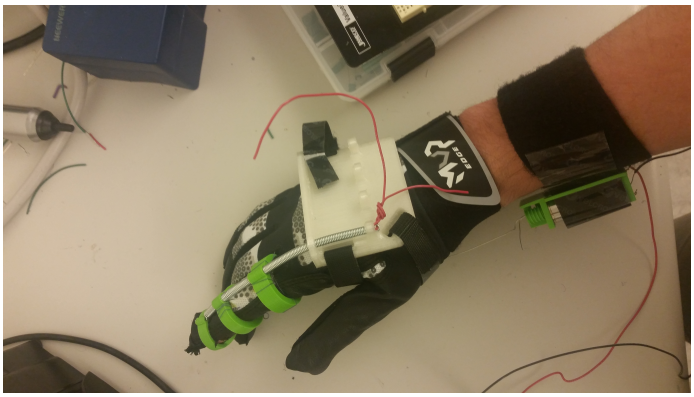

Figure 2.6: The glove will be strapped onto the users hand. As the motor pulls the finger will be curled.

were in the spool and the spring back method that would return the finger to its resting position.

The first problem was the spring back method that would return the finger to its resting position. In short, the spring would not return the finger to its resting position cleanly or consistently enough so a major change was in store. Immediately after the prototype was assembled, the idea to use elastic instead of a spring came to mind. The spring will not give much force when it is deflected normal to its direction. The elastic will provide the elastic force we need as the finger is curled and none when at resting position. This idea was implemented and worked great once we adjusted the resting tension. The elastic was attached to the tip of the finger to a new redesigned DP thimble, which will be discussed later, and to a new redesigned elastic band mount attached to the top of the hand. This elastic design can be seen in figure 2.7, where the red elastic band is mounted to the tip of the finger. As the finger is curled this elastic band is stretched by the cable be pulled. As the cable is released the elastic does the work of returning the finger to its resting position.

Figure 2.7: Elastic actuation system where elastic will return the finger to its resting position

The second problem that was fixed was the spooling mechanism. In the prototype the spool was friction fit with the shaft of the motor and the cable. In the final design we actually used a set screw that would go through the spool and pin the cable to the motor shaft. In order to do this we had to change spool to have a longer shaft at the bottom in which the set screw would be driven through. Instead of 3D printing he set screw hole we drilled a hole smaller than the set screw and then fastened the set screw through this hole by hand. The solidworks of the spool can be seen in figure 2.8. The integration of the set screw in the motor mount can be seen in figure 2.9.

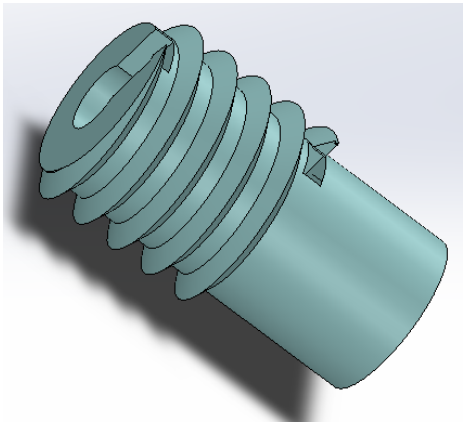

Figure 2.8: Final spool design with elongated shaft to fit the set screw to the motor shaft

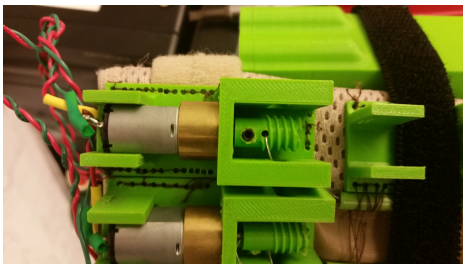

Figure 2.9: Final spool design integrated into motor mount with set screw attaching cable to the motor shaft

With these too major actuation problems solved we had a fully functioning actuation unit. The rest of the actuation design stayed the same meaning that the cable would still be spooled by a motor and would curl the finger.

2.2.6 Mechanical Structure: Final Iteration

The final mechanical structure was similar to the prototype design in concept except all of the parts were resigned to be slimmer, integrate the sensors better, and to fit the new requirements of the change from spring to elastic actuation. MP and PP rings were just slimmed down and the spring holes at the top of the rings were removed because we were no longer using the spring spring back technique. The major change that occurred involved the elastic mount on the top of the hand and the DP ring design which was changed to a thimble like structure.

DP Thimble Design

With our original DP ring design the finger would often be pushed out of the front of the ring making it so the user would no longer have consistent contact with the force sensitive resistor placed in the structure. In the end we updated the DP ring to be a thimble design. This gave a great surface for the sensor to be placed in and locked the users finger to the sensor. Now when the finger curled, it would not lose contact with the pressure sensor. The solidworks model of this thimble can be seen in two angles in figure 2.10 and 2.11. The loop on top of the thimbles is for the elastic band to wrap around.

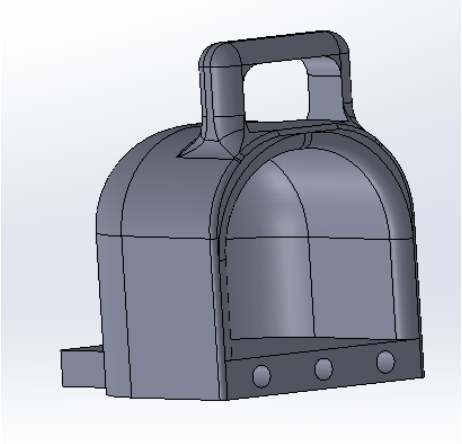

Figure 2.10: DP thimble joint: tip of the finger would rest in this component

Another major change to the ring design was where the leads of the pressure sensor were placed. We decided to run the leads out of the front of the DP thimble so that they would not kink or break. We would then run the wires that were attached to the leads with angled header pins over the top of the fingers. This can be seen in figure 2.12 and effectively stopped the leads from breaking.

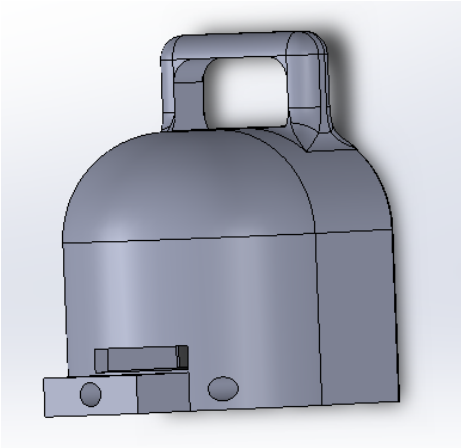

Figure 2.11: DP thimble joint: notice the tip extension where the leads of the sensor will rest

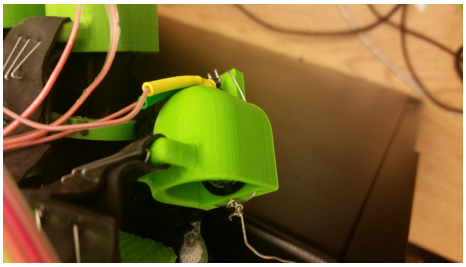

Figure 2.12: This is a figure of the DP thimble joint with the leads of the sensors being attached in the front with wires running over the top of the finger

Top of Hand Elastic Mount

Since we were now running elastic on top of the finger we needed to redesign the top of hand spring mount to now integrate elastic instead of springs. The design was also slimmed down so that it did not take up as much room. The thumb was now its own separate unit that was sewed lower than the top of the hand mount but was of the same design. The elastic mount can be seen in figure 2.13.

2.2.7 Glove Design: Final

This section will be written in the final report in spring quarter

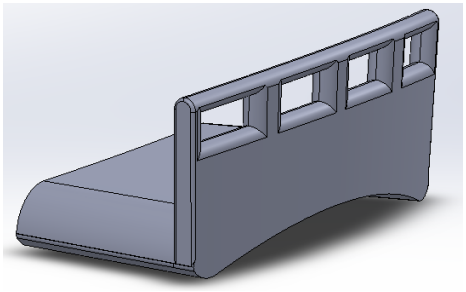

Figure 2.13: This component is mounted on top of the hand and mounts the elastic that is run to the tip of the finger

2.2.8 Mathematical Model

In order to get a better understanding of how much torque we need to pull the finger down to the curled position we created a mathematical model of our finger. This will give us a rough estimate of what specs we need to hit with our motor. After this we used our motor to try and pull our finger. There were some problems with this but it was fixed with the post width modulation cycle which is discussed in the software section. The motor specifications are discussed more in detail in the power budget section later in this paper. The mathematical model derivation is discussed in length and derived in the appendix.....

2.3 Electrical Design

2.3.1 Sensors

Flex Sensor

In order to actuate our hand we need to be able to detect motion of the hand and output a signal accordingly. We researched several types of sensors in order to accomplish this. To detect the flex of the finger we found several piezoresistive flex sensors that change resistance based on the angle. We decided on the SEN-10264 2.2" flex sensor; unfortunately, we found that the resistance values given in the datasheet were not accurate to the sensor. We decided to retake sensor measurements ourselves. In the figure 2.14 we plotted the resistance of the sensor based on the angle of the flex sensor.

The output resistance is fairly linear, which is desirable for our implementation. The graph also shows that that the resistance can exceed the flat resistance, which we must account for in our circuits. This should not be an issue in application though, since fingers do not bend backwards and our design does not actuate in this direction.

Angle(degrees)	Resistance (kΩ)
-30	23
-20	25
-10	27
0	30
10	40
20	51
30	60
40	64
50	70
60	81
70	90
80	95
90	100
100	115
110	122
120	133
130	140
140	150
150	155

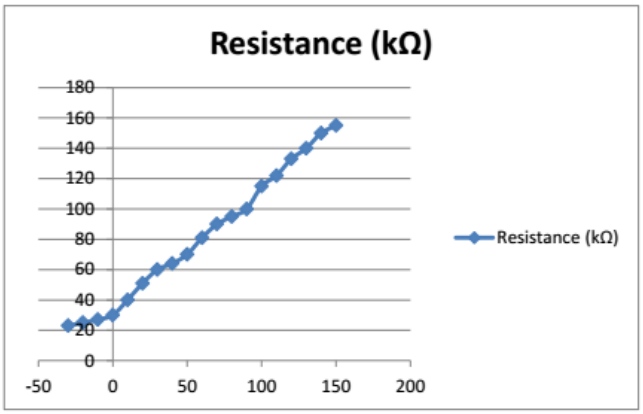

Figure 2.14: Resistance of flex sensor based on how much it is flexed.

Force Sensor

For detecting pressure output from a finger we researched both piezoresistive force sensors and capacitive touch sensors. We liked that the capacitive sensors would be extremely inexpensive to implement, but we decided on piezoresistive because it could give us a much better output range than the capacitive sensors. Once we decided on piezoresistive, we had to find a sensor that met our force requirements. Since the design is meant for stroke patients we had to account for their max force output range. In our research we found the average maximum force output for stroke patients to be 10.5kg plus or minus 9.5kg, so we need the bottom of this maximum range to be in the active range of our circuit. We found a piezoresistive force sensor that meets these requirements from Interlink Electronics. We then took measurements of the resistance versus weight by applying different weights to the sensor, the graph can be seen in figure 2.15.

Weight(g)	Rsensor(Ω)
10	2500000
20	2500000
50	60000
100	15000
200	6300
500	2500
max press	250

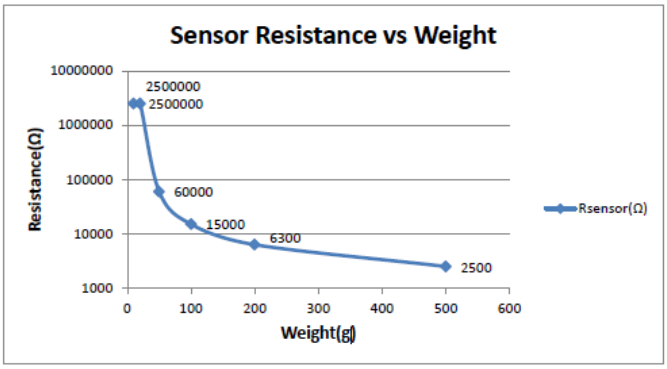

Figure 2.15: Resistance output for the pressure sensor.

As we observed the resistance to weight curve looks like an exponential, with the most sensitive region being within about .05kg and .25kg. These values are all within the range of the lowest maximum, which is the primary target for this project. We had to make some adjustments to our mechanical design to incorporate these sensors. Since the force sensor requires a flat surface below it to accurately measure force we had to adjust the ring at the DP joint to accommodate this. We solved this issue by making the bottom of the ring a flat circular surface and added the spring on top to ensure the necessary tension upwards to apply the force downwards.

2.3.2 Circuits

Flex Sensor

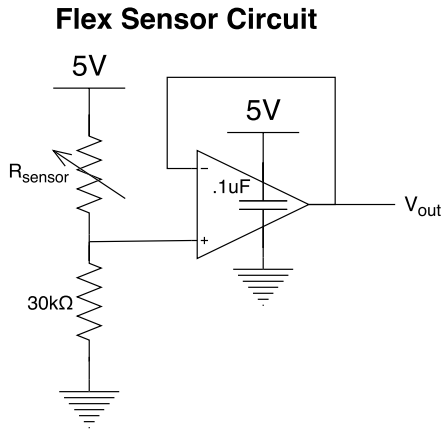

Figure 2.16: Circuit that conditions the signal for the flex sensor.

Angle(degrees)	Voltage (V)
-30	2.8
-20	2.74
-10	2.7
0	2.54
10	2.26
20	1.99
30	1.82
40	1.56
50	1.4
60	1.3
70	1.22
80	1.16
90	1.1
100	1.06
110	0.96
120	0.86
130	0.82
140	0.74
150	0.7

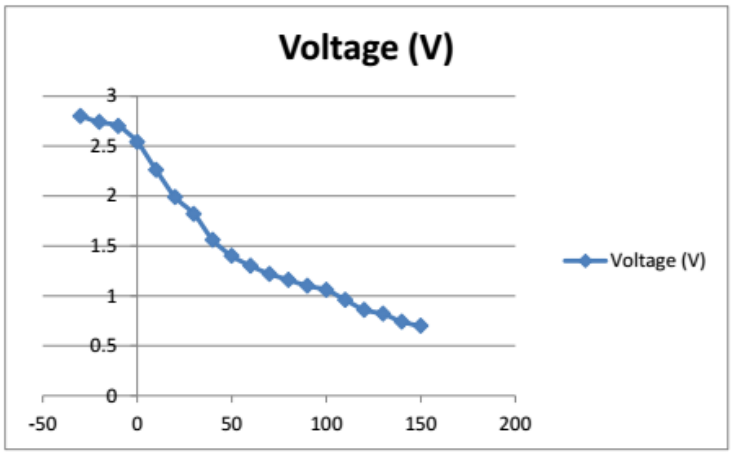

Figure 2.17: output voltage of flex sensor

The flex circuit we implemented is displayed in figure 2.16. With the resistor at the top of the voltage divider we use the minimum resistance to set the max output voltage of 3.3V. We then sent the output through a unity gain buffer in order to prevent any outside impedance from affecting the circuit. We then took measurements for the output of this circuit based on the angle of the flex sensor and graphed them in the figure above. The output of this circuit gives a good linear curve, but the output has a max at only 2.8V. This error is caused because the circuit achieves a max voltage when the flex sensor is bent completely in the wrong direction. While this can never happen in an actual application of the glove, we must still ensure the circuit output is always below the 3.3V max. In order to get better results we are likely going to remake this circuit with a 3.3V rail on our op-amp, this will clip the output and prevent it from ever exceeding the 3.3V max input for the PIC32.

Force Sensor

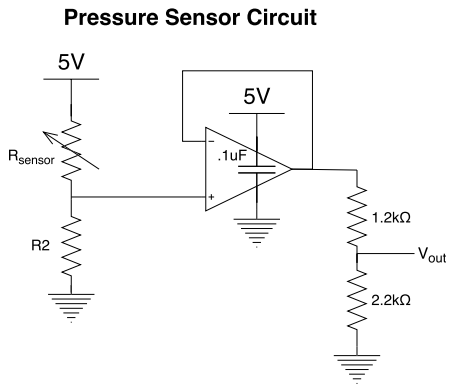

Figure 2.18: This is the circuit the conditions the signal from the pressure sensors.

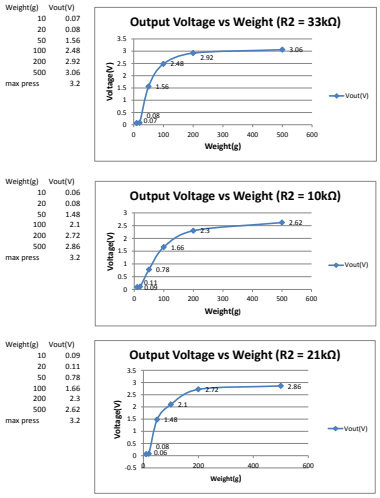

Figure 2.19: This is the circuit the conditions the signal from the pressure sensors.

The pressure sensor circuit, shown in figure 2.18, is fairly similar to the flex circuit. The main difference is that to get an ideal range the input to our unity gain buffer ranges from 0V to 5V, we then send this through a voltage divider to step it down to 3.3V. This allows us to adjust the range where the circuit is most sensitive. We took output measurements from the circuit as we adjusted the weight placed on the force sensor, and then repeated this with multiple R2 values to find the ideal range, ultimately deciding on setting R2 equal to 33kOHM.

2.3.3 Power Budget

In order to determine the battery that we will implement in our design we first need to determine our power consumption needs. To do this we must determine the voltage we will be running our hardware at and the maximum current requirements of each of our components. Our motor driver requires a voltage of at least 8V to be powered, so our battery options were either 9.9V or 12V. We found several good, cheap rechargeable batteries that supply 9.9V and decided to go with this as our voltage supply. We then went through and calculated the max current consumption of each hardware component based on their datasheets, and what rail they would be powered at. We see that the majority of our power consumption is from running our motor; however, when the finger is not in motion there is no current draw. This is because the torque of the motor is enough to offset the pull of the spring, as discussed in our mechanical design portion. This means that the motor is only running a small portion of the time, and an even smaller portion of that time is the motor stalling. So assuming each motor is only running about 5% of the time, we get an average maximum current draw of 288.5mA as shown in figure below **??**.

Part	Number of parts	Supply Voltage(V)	Supply current(mA)	Total(mA)	Total(mA) Assuming motor on 5% of time
Mini motor	5	10	800	4000	200
DRV8814 Dc motor driver	3	10	4.5	13.5	13.5
UNO32	1	10	75	75	75
MCP6004 Sensor Circuit	5	3.3	5	25	25
				4088.5	288.5

Battery Options	Voltage(V)	Battery Life (mAH)	Weight(g)	Size LxHxW (mm)	Price	Battery Life(hours)
Turnigy LiFe	9.9	1500	115	102x29x23	$6.99	5.199306759
HobbyKing Life	9.9	1500	118.8	102x28x23	$8.67	5.199306759
ZIPPY 30C LiFePo4	9.9	2100	218	137x45x17	$15.52	7.279029463
ZIPPY LiFePo4	9.9	1800	136	95x28x30	$11.98	6.239168111
Tenergy NiMH	12	5000	721	225x48x25	$81	17.33102253
Tenergy NiCD	12	1300	386	168x48x24	$29.99	4.506065858

Figure 2.20: power budget

Using these calculated values we can now research batteries based on our voltage and current needs. We found multiple batteries that supply 9.9V with different battery lives. Using our calculated current consumption and the battery life given in milli-amp hours we calculated the different battery life we would get

out of each. We decided on the ZIPPY 30C LiFePo4, as it gives us a minimum average of over 7 hours of functionality. The battery is also rechargeable, fairly safe in human applications, and the size and weight are acceptable for our design specifications.

2.4 Software Design

After incorporating the sensors and the mechanical design, there is a need to be able to control the system. After demonstrating that we could actuate a finger which the design we had, and we could also detect the appropriate sensitivity on the finger pressure and flex, we turned o controlling the actuation in response to the user input. In this section we deribe the decisions we made for the devices implemented and the moethods by which we incorporated the control.

2.4.1 PIC32MX320F128H

The first decision we had to make was to choose a micro-controller that would meet our specifications. Or decision stemmed from the teams familiarity of the hardware, as well as the micro-controller also meeting all of the specifications that we needed, containing all of the necessary Analog-To-Digital(ADC) control pins that we needed to be able to read the varying sensor data.

In conjunction to all of these benefits there was also a robust, ready-to-use, library written for the the PIC32 by Professor Dr. Gabriel Elkaim, and Max Dunne. that we were able to use in order to streamline the software implementation, making it significantly easier to start programming the controller system.

We looked to make sure that 4 main components were available and functions to our disposal:
Pulse-width Modulation(PWM), was a necessary component that was implemented to drive the motors by PWM the motors through an H-bridge. A library was provided to modulate the duty cycle of the signal and output it through a pin on the board.
Timers. timers were a necessary component to make sure that our sensor reading were made at specific times, so that we could respond to user input quickly.
Digital IO, was necessary to determine the direction in which the motors would run, a simple function was implemented that would change the value of a specific output, based on the direction enabled in software.
Analog To Digital Conversion (ADC) was necessary to read the varying values from the sensors, his allowed us to implement and events and services state-machine, allowing us to not have to deal with raw sensor values in the main state-machine code.

2.4.2 Sensor and Actuator Software and Services

Sensors Event and Services

Once we knew we had a function ADC library we took to implementing a sensor service which would sample the sensors, compare the current readings with previous readings, and post events when specific events were detected (i.e. pressed, stabilized, etc.)

In Table 2.1 we call the finger "stable" if the finger is pressing the pressure

Sensor	Event Name	Description
Flex	`FLEX_NO_EVENT`	No change sensed by the flex sensor in since the last reading.
	`FLEX_finger`	Event: finger (i.e. `THUMB`, `INDEX`, ...) has reached max value.
	`MOVING_finger`	Event: finger is between flexing and relaxed states.
	`RELAXED_finger`	Event: finger has reached the minimum flex value.
Pressure	`PRESS_NO_EVENT`	No change sensed by the pressure sensor since the last reading.
	`PRESS_finger`	Event: finger has gone from "stable" to "pressed"
	`STABLE_finger`	Event: finger has gone from "un\pressed" to "stable"
	`PRESS_finger`	Event: finger has gone from "stable" to "unpressed"

Table 2.1: List of Sensor Events based on Sensor and Description

sensor but not enough to indicate that the user wants to move the finger. This is because the way the sensor is fixed in the glove there will always be some non-zero reading that when the user simply wants to hold the position of the finger fixed.

Motor Control

To drive the motor we wrote a support library that would set the correct pins high for each finger and would also produce the necessary PWM and Direction signal. The library is also expandable for when multiple fingers are incorporated, by designating which PWM and direction channel you will direct the signal.

After starting with an initial PWM duty cycle of 50% it was concluded that we were not exerting enough torque from the motor. Tests need to be run to find the optimal duty cycle, but we believe we are close to the true value.

2.4.3 Controller Design and Implementation

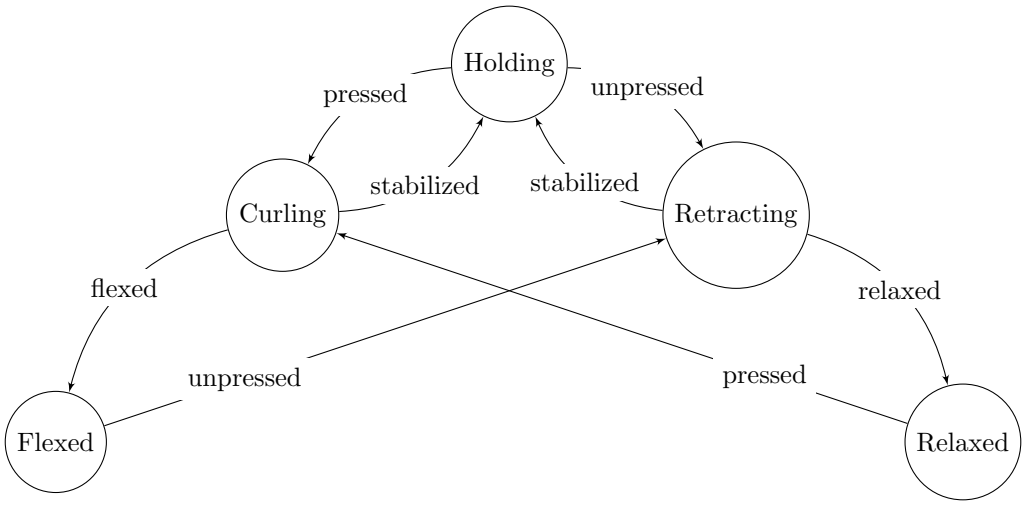

Figure 2.21: State-Machine Diagram

the system we have presented above in Figure 2.21 has proven to be working ver well. We have included states that activate as safe-guards to protect the user from accidentally over bending or over relaxing, and then sending the spool to tense the wire in the other direction.

Each of the states represents a stage in the motion of a finger, a holding state was included so that a user could hold a fixed position if they wished. The hope to integrate sensors that would also be able to sense the force applied to an object. as another safety mechanism to make sure thee user over tension their finger, and potentially hurt themselves. Currently the system can operate as expected tensioning and relaxing finger based on subtle user input

Chapter 3

Conclusion

Our preliminary efforts looking promising, more on this section will reveal itself next quarter when the finalized design is implemented and completed/

Chapter 4

Future Work

Our goals for next quarter are to adjust the circuitry and create the PCB design of all the hardware. We must then replicate our design for multiple fingers and make necessary adjustments to the mechanical design to account for the thumb.

Chapter 5

Acknowledgments

We are designing an incredibly complex system. Aside from the tremendous effort needed to come up with the systems design and lay out, the prototyping phase can become quite expensive and cumbersome.

Thanks to the mentorship and wisedom passed onto us from Dr. Patrick Mantey. Additionaly we would like to thank Patrick Ayers, your guidance, experience and insight have helped us immensely.

We would also like to thank CITRIS, and the University of California, Santa Cruz, for providing space and support for our development. We look to change the world for the better and are delighted to have you lending a hand.

Also our gratitude extends to Dr. Gabriel Hugh Elkaim and Max Dunne, for letting us incorporate their software libraries for the PIC32, and allowing us streamline the process of writing software.

Chapter 6

Team Members

Victor Ardulov

Contact:

- victor@ardulov.com
- vardulov@ucsc.edu

Responsibilities: Software engineering, state-machine design, controller implementation, miscellaneous mechanical design.

Devin Cody

Contact:

- dcody@ucsc.edu

Responsibilities: Electrical engineering, PCB design, power budgeting, sensor integration, electronics implementation.

Kyle Cordes

Contact:

- kjcordes@ucsc.edu

Responsibilities: Software implementation, mechanical design, electronics integration, mechatronic system design

Chapter 7

Appendices

Appendix A: Forward Kinematics and Mathematical Modeling

One of the important steps in designing a a mechanical model was understanding the kinematics of motion. To do this we needed to be able to describe the motion of the finger tip, henceforth referred to as the end-effector. Generally, the equation of a position of an effector as function of the angle between itself and its previous effoctor, is reduced to a matrix, that is called the Translation matrix. For any given effector in a system i and it previous effector $i-1$ we define the translation in 3 dimensions to be:

$$T_{i-1}^{i} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In this model:

- α represents the angle between 2 effectors with reference to the z (in and out of the page)
- θ represents the angle between 2 effectors with reference to the x axis (along the axis of the link connecting 2 effectors)
- **a** represents the distance in the z direction between 2 effectors
- **d** represents the length of the link connecting 2 effectors, or ,more generally, the distance between 2 effectors.

However due to the constraints we have applied to our systems, we will represent our systems with values of a and α exclusively equal to 0

From this we also derived that that motion of the end effector for an arbitraty n^{th} order link system can be defined as the product for translation matrices:

$$T_0^n = \Pi_{i=0}^{n-1} T_i^{i+1}$$

While these facts are well extablished in the domain of robotic manipulation, it is clear to see that this is only partially applicable to our situation.

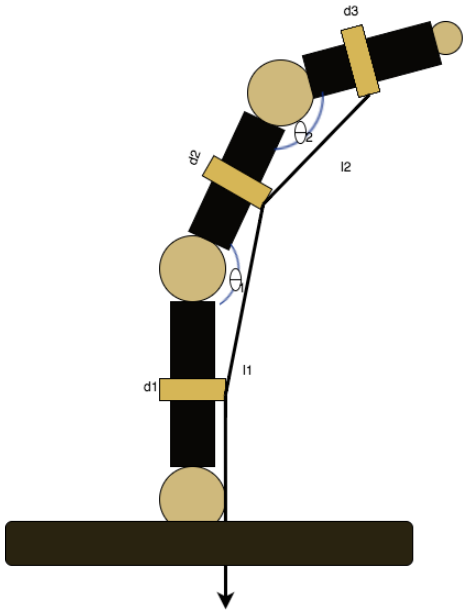

Figure 7.1: Caption

The issue arises in that while we can simulate the motion of the end effector using angle to define motion, our model uses a cable to actuate motion. This meant that we would need to compute the function $\Theta_i^{i+1}(l_1)$, as the angle between 2 links and l_i represents the length of the cable connecting 2 links.

What we came to realize is the we can generalize the system as a triangle with sides of length $\frac{d_i}{2}$, $\frac{d_{i+1}}{2}$, l_i from this we derive the following relationship from Law of Cosines:

$$(\frac{d_i}{2})^2 + 2(\frac{d_i}{2})(\frac{d_{i+1}}{2})\cos(\Theta) + (\frac{d_{i+1}}{2})^2 = l_i^2$$

Therefore

$$\Theta = \cos^{-1}\left(\frac{l_i^2 - (\frac{d_i}{2})^2 - (\frac{d_{i+1}}{2})^2}{2(\frac{d_i}{2})(\frac{d_{i+1}}{2})}\right)$$