

The University of Maine

DigitalCommons@UMaine

Honors College

Spring 2019

A Shoulder Mechanisms for Assisting Upper Arm Function With Distally Located Actuators

Michael Jones
University of Maine

Follow this and additional works at: <https://digitalcommons.library.umaine.edu/honors>



Part of the [Mechanical Engineering Commons](#)

Recommended Citation

Jones, Michael, "A Shoulder Mechanisms for Assisting Upper Arm Function With Distally Located Actuators" (2019). *Honors College*. 511.

<https://digitalcommons.library.umaine.edu/honors/511>

This Honors Thesis is brought to you for free and open access by DigitalCommons@UMaine. It has been accepted for inclusion in Honors College by an authorized administrator of DigitalCommons@UMaine. For more information, please contact um.library.technical.services@maine.edu.

A SHOULDER MECHANISM FOR ASSISTING UPPER ARM FUNCTION WITH
DISTALLY LOCATED ACTUATORS

by

Michael Jones

A Thesis Submitted in Partial Fulfillment
of the Requirements for a Degree with Honors
(Mechanical Engineering)

The Honors College

University of Maine

May 2019

Advisory Committee:

Babak Hejrati, Assistant Professor of Mechanical Engineering, Advisor
Vincent Caccese, Professor of Mechanical Engineering
Andrew Goupee, Assistant Professor of Mechanical Engineering
Mohsen Shahinpoor, Professor of Mechanical Engineering
Stefano Tijerina, Lecturer in Management

ABSTRACT

This thesis presents a new design for a shoulder assistive device based on a modified double parallelogram linkage (DPL). The DPL allows for active support of the arm motion in the sagittal plane, while enabling the use of a distally located motor that can be mounted around the user's waist to improve the weight distribution. The development of the DPL provides an unobtrusive mechanism for assisting the movement of the shoulder joint through a wide range of motion. This design contains three degrees-of-freedom (DOFs) and a rigid structure for supporting the arm. The modified DPL uses a cable-driven system to transfer the torque of the motor mounted on the user's back through the links to the arm. The proposed design assists with the flexion/extension of the arm, while allowing the adduction/abduction and internal/external rotations to be unconstrained. A kinematic analysis of the cable system and linkage interaction is presented, and a prototype is fabricated to verify the proposed concept.

TABLE OF CONTENTS

LIST OF FIGURES	iv
LIST OF TABLES	v
PREPARATION OF THESIS	1
Introduction	1
Literature Review	5
DESIGN PROCEDURE	7
Preliminary Design.....	7
Shoulder Rail.....	9
Distal Motor Cable Driven DPL Kinematics	11
DPL Version 1.....	16
DPL Version 2.....	18
DPL Version 3.....	19
DPL Version 4.....	20
DPL Version 5.....	20
DPL Version 6.....	26
DPL Version 7.....	27
EXPERIMENTS AND RESULTS:.....	31
Motor Functionality.....	31
Workspace Analysis.....	34
CONCLUSION AND FUTURE WORK:	36
REFERENCES	37
APPENDICES	38
APPENDIX A.....	39
APPENDIX B	40
APPENDIX C	51
AUTHOR’S BIOGRAPHY	61

LIST OF FIGURES

Figure 1: Planes of the human body.	2
Figure 2: Motions of the shoulder joint that an exoskeleton needs to account for in its design.	3
Figure 3: Rendering of shoulder rail concept mounted to backpack on person.	10
Figure 4: Three-views of the double parallelogram mechanism on the shoulder, (a) top view with the left arm at an angle, (b) back view and, (c) side view.	12
Figure 5: Cable and pulley mechanism schematics.	12
Figure 6: Tension created when the motor rotates the input pulley.	14
Figure 7: Slack created when the pulley at point B rotates.	14
Figure 8: Modified cable and pulley mechanism with cable constrained at B.	15
Figure 9: Cable is kept in tension when the modified pulley system bends at B.	16
Figure 10: Geometric diagram to find the cable length on the RHS of B.	16
Figure 11: First DPL prototype.	17
Figure 12: First DPL prototype input (right) and output (left).	17
Figure 13: Second DPL prototype.	18
Figure 14: Simplified cable guide.	19
Figure 15: Crossbar output shaft with cable passing through center, (a) crossbar vertically aligned, (b) crossbar at some off-axis angle.	20
Figure 16: Version 5 of the DPL mechanism.	22
Figure 17: Output pulley with the cable in groove.	22
Figure 18: Output pulley cable bend point, (a) DPL at right angle, (b) DPL in open position, (c) DPL in closed position.	22
Figure 19: Actuation of DPL in throughout the internal/external rotation, (a) 90° position, (b) open position, (c) closed position.	24
Figure 20: Experimental results of input to output rotation of DPL at 90° angle.	24
Figure 21: Experimental results of input to output rotation of DPL at fully open position.	25
Figure 22: Experimental results of input to output rotation of DPL at fully closed position.	25
Figure 23: DPL mechanism version 6.	27
Figure 24: Input pulley is mounted out of plane from the DPL and a second cable transfers the torque from the motor.	27
Figure 25: Rendering of the exoskeleton mounted on the user.	29
Figure 26: Input cable mounts directly to the motor shaft.	29
Figure 27: Final design for distal motor DPL.	30
Figure 28: Three-views of the double parallelogram mechanism on the shoulder, (a) top view with the left arm at an angle, (b) back view and, (c) side view.	30
Figure 29: Input cable guide aligns the cable along the vertical axis of DPL.	30
Figure 30: Experimental setup for testing exoskeleton rotation.	31

Figure 31: Rotation angles of the motor input shaft, the output shaft, and the dummy's arm.	33
Figure 32: The DH parameter diagram of the exoskeleton prototype in operational configuration.	34
Figure 33: Workspace of DPL end effector	35

LIST OF TABLES

Table 1: Table of values for the DH parameters.	35
--	----

PREPARATION OF THESIS

Introduction

Robotic exoskeletons have been used to assist people with limited range of motion in performing daily tasks while they are worn by their users. Exoskeletons can take different forms such as full-body, lower-limb, and upper-limb exoskeletons depending on their use case scenarios. Upper-limb exoskeletons are particularly useful for their assistance with daily activities such as eating, lifting objects, brushing teeth, etc. These types of exoskeletons can be designed to assist with the shoulder, elbow, and hand mobility. This thesis focuses on the design of a shoulder exoskeleton for upper-arm mobility. The scope of the project is to design a proof-of-concept prototype that can move the arm in the sagittal plane of motion. The planes of motion of the human body are shown in Fig. 1, and the motions of the shoulder joint are shown in Fig. 2. The motion on the sagittal is the motion in which the arm moves during walking and reaching tasks, so it is desired to have a system that is capable of swinging the arm as well as assisting with lifting the arm. For this project, the designed prototype will be capable of partially assisting the user in the sagittal plan motion and will not provide complete power for movements. While a control system and implementation will be necessary in the future, the goal of this project is to focus on proving a proof of concept for an exoskeleton and presenting the design of necessary mechanical components to enable the desired kinematics. The kinematic analysis of the proposed design will be also presented in this thesis. This thesis covers a comprehensive literature review of the existing shoulder

exoskeletons, the preliminary design, prototyping, and testing of a novel assistive shoulder exoskeleton.

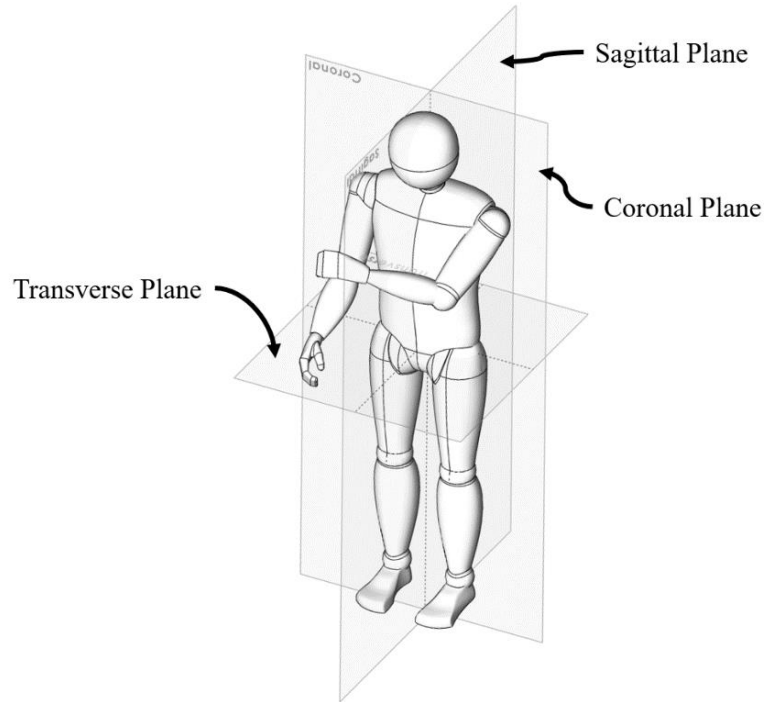


Figure 1: Planes of the human body.

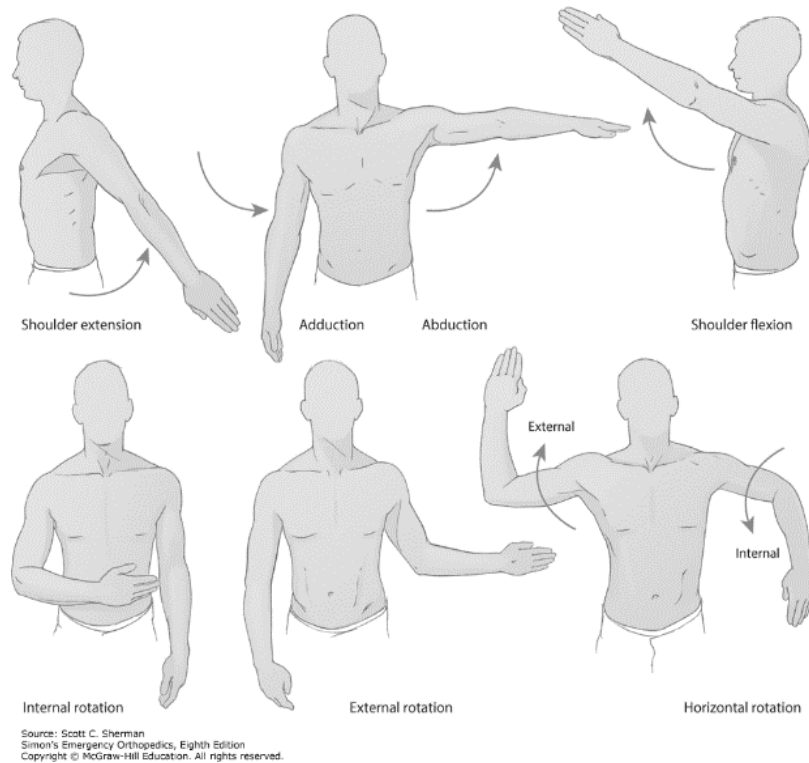


Figure 2: Motions of the shoulder joint that an exoskeleton needs to account for in its design.

The challenge with designing an exoskeleton for the arm is to determine the sufficient number of degrees of freedom (DOFs) to provide an adequate assistance, while keeping the weight of the mechanism low to avoid fatigue on the user. A DOF of an exoskeleton is a motion that allows the body to move in a particular way. The shoulder joint is a ball-and-socket connection that has three DOFs, which means that an exoskeleton, which allows the shoulder to move in all natural DOFs, must have at least three DOFs. An exoskeleton can have more DOFs than the part of the body it is supporting, which results in more freedom of movement and excessive complexity. An exoskeleton that matches the number of DOFs of the body part will be as mechanically simple as possible but be more rigid than if it had an excessive number of DOFs. The two types of DOFs are active and passive. An active DOF is powered by the system and uses

some type of actuator to move the body. A passive DOF is a motion that the exoskeleton allows, and it only depends on the user to move it. It is important to consider the correct number and types of DOFs into an exoskeleton design to achieve the desired motion. Without the correct layout, an exoskeleton can experience inefficiency, backlash, and undesired back driving.

Back driving is a phenomenon that can occur in exoskeletons that either have too many DOFs that are not aligned properly or in exoskeletons that do not have enough freedom of movement. Back driving is when a motion of the exoskeleton or the user results in the unintended motion of the other. For example, if a shoulder exoskeleton has four DOFs that are not aligned properly, when the person moves their arm in a certain way, the exoskeleton could be forced to move in unintended ways. This motion can be described as a “floppy” exoskeleton. This is an example of the body motion back driving the exoskeleton. In the opposite fashion, an exoskeleton can be built in such a way that when an active DOF of the exoskeleton is actuated with the motor or when the user moves, it causes unintended motion on the user. This can be harmful to the person if the exoskeleton moves unexpectedly, which could hurt the user. An exoskeleton that is prone to back driving has to be avoided for the purpose of keeping users safe, comfortable, and not damaging the exoskeleton.

This thesis project begins with a literature review, goes through the design and prototyping process for finding the optimal shoulder exoskeleton configuration, and finishes with testing of the final prototype. This thesis will document all of the steps taken throughout this process. An additional component of this thesis is a research paper accepted in the 41th IEEE International Engineering in Medicine and Biology Conference

(EMBC 2019). This paper was also accepted for presentation at EMBC 2019 that will be held in Berlin, Germany, July 2019.

Literature Review

More recently, the importance of arm swing for gait rehabilitation has been presented by Hejrati et al. [1], [2], which capitalizes on the design of a wearable assistive device to induce arm swing. The challenge with designing an exoskeleton for the arm is to determine the sufficient number DOFs to provide adequate assistance, while keeping the weight of the mechanism low to avoid user fatigue.

Several designs exist for overcoming this problem such as soft cable-driven exoskeletons, which rely on the user's own body to provide the structure. A soft exoskeleton has the benefit of moving the joint without causing joint misalignment because the cables use the skeletal structure of the body for support [3], [4]. Dinh et al. [4] attempted to mitigate slacking and backlash of the cable-driven systems by pre-tensioning the cables and using a nonlinear adaptive controller to actuate the system. Barns et al. [2] proposed a design which uses a pulley-belt mechanism and a cable-driven haptic paddle to transfer the torque generated by two distal motors located around the user's waist to their arms.

A spherical mechanism provides a rigid structure that closely follows the shoulder's movement and can fully support the movement of the arm [5], [6]. Hsieh et al. [6] used linear stepper motors instead of DC motors. This actuator change allowed them to use two slider-crank mechanisms and two spherical mechanisms to design a shoulder

joint with 2 active and 4 passive DOFs. Although this design was unobtrusive, compact, and lightweight, the spherical mechanism is a complex solution and requires many components to follow the shoulder movements.

A design by Liu et al. [7] can provide assistance up to 30% of a human's arm weight during flexion/extension. The design has one powered and two passive DOFs, therefore it is relatively light and weighs only 5.1 kg. A similar mechanism by Sui et al. [8] provides 4 active and 1 passive DOFs and weighs 8.4 kg for both arms. These designs focus on reducing the weight of the mechanism, but only provide a fraction of the lifting force that a healthy person could exert. A design by Ebrahimi et al. [9] has 3 active and 9 passive DOFs and can generate up to 40 N-m of torque for shoulder flexion/extension and up to 24 N-m of torque for elbow flexion/extension. A tradeoff between these design concepts would be necessary to satisfy both capability and weight requirements.

This paper presents a modification to the double parallelogram linkage (DPL), which is used in previous studies [10], [11] to allow unconstrained internal/external rotations and use cables to transfer the torque through a rigid mechanism. The proposed mechanism uses a DPL to provide 3 DOFs on the shoulder: 1 active DOF for assisting arm flexion/extension and 2 passive DOFs for other rotations. The system is intended to be used as an assistive mechanism to augment the user's arm strength during lifting tasks. Therefore, its generated torque will be low to keep the weight down. The conceptual design, geometric analysis, and prototyping of the mechanism are presented in this thesis.

DESIGN PROCEDURE

Preliminary Design

The preliminary design followed the design principal of using a DPL system to provide the structure for the exoskeleton. This idea was primarily influenced by the design by Bai et al. [10]. The purpose of using such a design is to prevent collision with the user, while also providing a structurally rigid platform for supporting the shoulder. It is a 3-DOF system that allows for passive internal/external rotation and powered flexion/extension and abduction/adduction rotations. The design by Bai et al. [10] use two motors, one that mounts to the user's back and the other one mounts on the outside of the arm. The benefit of this setup is that it has two powered DOFs and enough range of motion to allow for normal arm movement. A standout feature of the DPL mechanism is that the motor, which controls flexion/extension, moves with the user's arm and is always aligned with the plane of the elbow joint. This was seen as favorable as the mechanism will assist the motion in the plane of the elbow joint instead of only the sagittal plane. For daily tasks such as eating and reaching, this capability is desirable.

The DPL mechanism with two powered DOFs has some downsides such as adding weight and having more inertia on the shoulder due to the motor mounted to the arm. The entire weight of a shoulder exoskeleton is important to keep as low as possible since the user has to carry the weight on their body. If the weight of the mechanism is mounted to a backpack, then the back and legs have to take up the weight. For this reason, low weight is a critical design point for the exoskeletons. The scope of this thesis calls for a powered DOF in the sagittal plane. This means that the extra weight of an

abduction/adduction motor can be left out of the design. If the motor mounted to the user's back is removed and replaced with a passive DOF, then the weight of the device will rest on the user's arm, further impeding their range of motion. This effect could negate the benefit of a shoulder exoskeleton by making the user work more to support the weight of the system. To solve this problem, the motors would ideally be moved off the arms and have their weights supported by the backpack. By doing this, the overall weight of the system is reduced, and the weight felt by the arms is significantly decreased.

The other issue with mounting a motor on the user's arm is that as the arm moves, the inertia of the motor will make it harder for the exoskeleton to follow the motion of the arm. As the weight of the system rotates, the moment arm created by the distance between the back mount and the motor on the arm will create a significant torque that the user will feel on their body. If they move their arm too fast, they might not be capable of handling the torque generated by a heavy motor on their arm. To prevent this, the motor should be mounted closer to the center of gravity of the user to reduce the torque transferred to the user. By looking at the two issues associated with the DPL presented by Bai et al. [10], it is noticeable that by moving the weight of the exoskeleton closer to the user's body and off their arm, it will be more comfortable and less fatiguing. The new design should follow this design philosophy and work towards reducing the weight of the system carried by the arm.

Shoulder Rail

The shoulder rail concept is the first design explored in the design process and is intended to reduce complexity of the overall system and allow for smooth operation. It uses a circular rail that wraps around the shoulder that connects from the back of the shoulder to the side of the arm as shown in Fig. 3. This design is intended to use only one motor to power the motion of the shoulder instead of multiple motors. The motor should be mounted to the back of the user to reduce the weight felt by the arm. The rails that wrap around the shoulder have a slider that follows the internal/external rotation of the arm and has a cuff that connects to the upper-arm to move the user's arm. This slider would passively follow the arm as the user moves their arm around. The motor would connect on the user's back in a comfortable position and use cables to transfer the torque of the motor to the arm. The rail would be a semi-circular curve that carries a slider around the pivot point of the shoulder. Since the rail is stationary and a slider runs along it, the rail would be stationary at all times. The problem with the rails always being in the same position is that when the user raises their arm, their arm could collide with the rails in some positions. There was no configuration found that would allow for free range of motion at all times. The cable drive showed potential in this design and was explored in detail.

By using a cable to transfer the torque to the output slider of the rail system, the torque of the motor could be transferred to another point. The cable can be set up like a pulley system that has an input and output pulleys and the rotation of one matches the rotation of the other. By using a pulley in the rail design, one of the pulleys would be moving relative to the other pulley so that the distance would always be changing. The

problem with this approach is that a pulley system that uses a cable relies on tension in the cable to create rotation. As the distances between the pulleys change, the cable will not remain tensioned, so the cable will not transmit the torque. A cable and pulley system also requires the points where the pulleys are mounted to be rigid. As the cable pulls on the cable, it creates a force that will pull the pulley in the direction of the cable force if the system is not rigid. These problems resulted in the rail design not being further explored but the cable and pulley system proved to be a potential solution for mounting the motor on the user's back.

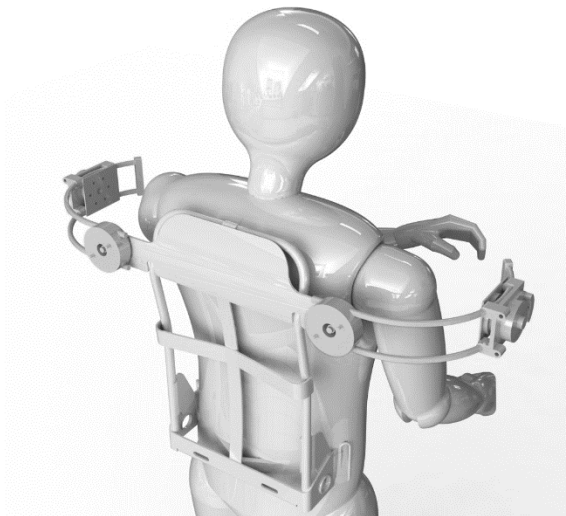


Figure 3: Rendering of shoulder rail concept mounted to backpack on person.

Distal Motor Cable Driven DPL Kinematics

The design of the final shoulder mechanism design uses a combination of a DPL and a modified cable drive. The DPL shown in Fig. 4 allows for internal/external rotation (left arm in Fig. 4 (a)) by aligning the output link with the outside of the arm. The input shaft mounts on the torso and is aligned with the axis of abduction/adduction rotation of the shoulder and provides a passive DOF. The output shaft aligns with the axis of flexion/extension rotation and provides the active DOF. The sizing of the mechanism to the user is determined by D_1 and D_2 and can be adjusted to fit a wider variety of sizes. By combining the DPL with the modified pulley mechanism, the torque from the input shaft mounted onto the torso can be transferred to the output shaft connected to the user's arm. The input axis is along the vertical axis while the output axis is horizontal. In this configuration, the cable is twisted 90° as it goes from the input to the output. The power for the mechanism comes from a distally located stepper motor that uses a cable-driven system to transfer the torque to the output shaft. As shown in Fig. 4, two cables transfer the torque from the motor to the input shaft at point A and from the input shaft to the output shaft at point C. The input shaft is mounted on the user's back perpendicular to the transverse plane. The input shaft is the interaction point between the two cables used in the system: the motor cable and the DPL cable. The torque is transferred from the input shaft to the output shaft similar to a simple cable drive mechanism with two pulleys as seen in Fig. 5. This diagram simplifies the mechanism and allows for analysis of the power transfer. Point B is a passive DOF hinge joint allowing link 1 and link 2 to actuate freely based on the user's motion. Point C corresponds to the powered flexion/extension motion in the arm. The pulleys are connected with a continuous cable loop. It is desired

for B to be a free center-of-rotation while A powers the rotation of C. The issues with a simple pulley layout are that cable tensioning and slacking occur during the motion. Cable slacking and cable tensioning happen simultaneously when the mechanism actuates at B to allow for internal/external rotation. For the purposes of examining them, the cable tensioning and slacking are explained separately.

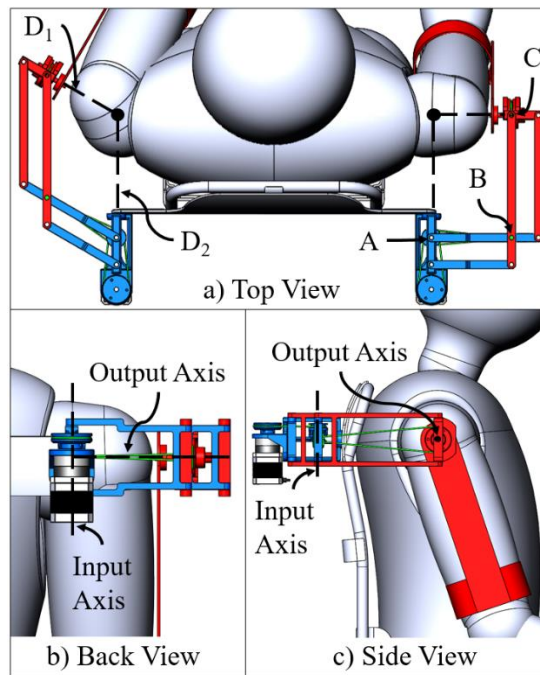


Figure 4: Three-views of the double parallelogram mechanism on the shoulder, (a) top view with the left arm at an angle, (b) back view and, (c) side view.

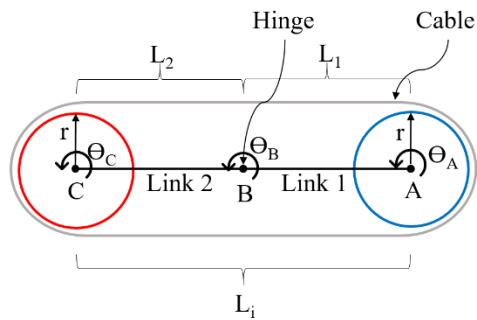


Figure 5: Cable and pulley mechanism schematics.

Cable tensioning occurs when the motor initially starts to rotate the input pulley, or the mechanism is actuated at B. Fig. 6 shows a scenario where A is rotated by the motor, B is free to rotate, and C is locked. This case simulates the scenario that the user wants assistance from the motor to raise their arm in flexion direction without moving their arm in internal/external rotation. Point C is locked because during the motion the output pulley, which is connected to the user's arm, experiences significant resistance. When the pulley at A is rotated counter-clockwise, tension is created in the cable on the bottom. Since point B is the only place with free rotation, this tension will result in an upwards movement of point B as shown in Fig. 7. The movement and subsequent rotation of point B is an undesired outcome of the motor actuation.

Due to the tensioning of the cable, the opposite side of the cable becomes slack as soon as tensioning occurs. Fig. 7 shows the result of the motor rotation. The final distance L_f between the pulleys in Fig. 7 is shorter than the initial distance L_i between the pulleys in Fig. 6. Since the distance between the pulleys decreases, the cable will not stay in contact with the pulleys and will no longer stay tensioned.

The same pulley setup can be kept, and a modification is made by constraining the cable at B so that the cable passes through the axis of rotation at B. This modification can be seen in Fig. 8. The cable is constrained at B by using a low friction guide hole to prevent the cables from escaping. By passing the cables over the axis of B, the tension will be maintained in the cable and slack will not be created. No torque will be generated around point B from the tension force in the cable because the moment arm of the cable is reduced to zero about B. The input pulley can still transfer torque to the output pulley

similar to the simple pulley configuration because the pulleys are connected by a continuous loop of cable.

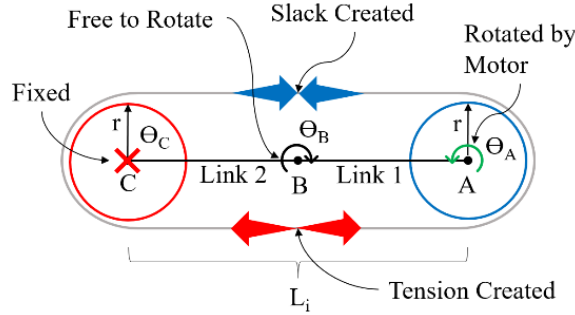


Figure 6: Tension created when the motor rotates the input pulley.

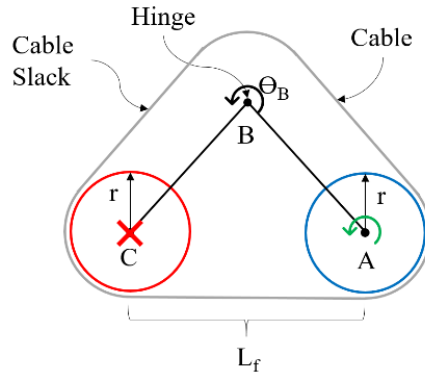


Figure 7: Slack created when the pulley at point B rotates.

This modification separates and isolates the torque transfer from A to C from the rotation at B. When the user wants to rotate their arm across their body in internal/external rotation, θ_B changes but the cable only bends at point B and does not create tension or slack as shown in Fig. 9.

The total length of the cable on each side of point B is constant and can be seen by comparing Fig. 8 with Fig. 9. At an angle of $\theta_B = \pi$ shown in Fig. 8, the total length of the cable on the right-hand side (RHS) of B noted by m is given by:

$$m = C_1 + C_2 + C_3 \quad (1)$$

At some other arbitrary angles of θ_B shown in Fig. 9, the total length of cable on the RHS of B noted by n is given by:

$$n = C_4 + C_5 + C_6 \quad (2)$$

The values for m and n can be found by calculating the total length of the cable using the triangles shown in Fig. 10, where $x = \sqrt{L_1^2 - r^2}$, $y = r \sin(\theta)$, and $\sin(\theta) = r/L_1$. These relations are based on a small angle approximation for θ to assume y is a straight line. Also, $C_1 = C_4 = \pi * r$.

$$m = n = 2 * \left(\sqrt{L_1^2 - r^2} + \frac{r^2}{L_1} \right) + \pi * r \quad (3)$$

By constraining the cable in at the point of rotation where no torque is desired, the tension of the cable will be isolated from the rotation of point B. The same analysis as in (1)-(3) can be done to find the length of cable on the left-hand side (LHS) of B. For any angle θ_B , the total length of the cable (both on the LHS and RHS of B) remains constant, therefore no tension is created.

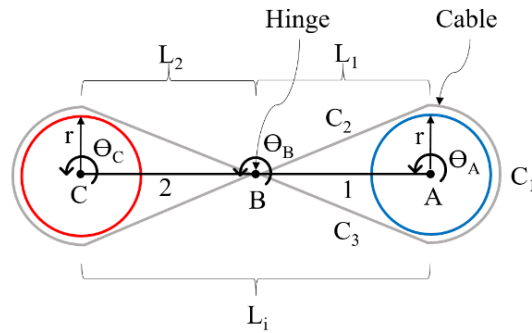


Figure 8: Modified cable and pulley mechanism with cable constrained at B.

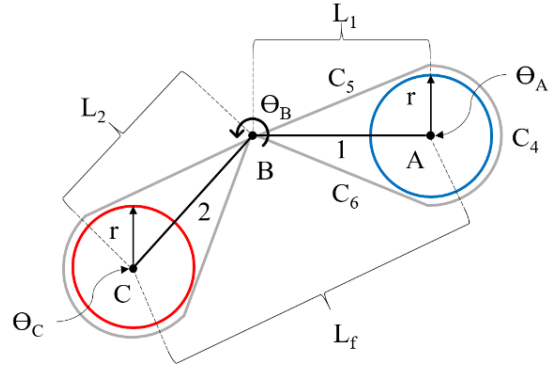


Figure 9: Cable is kept in tension when the modified pulley system bends at B.

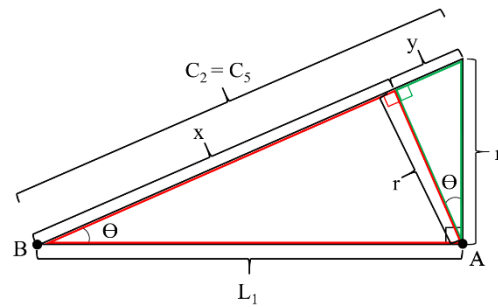


Figure 10: Geometric diagram to find the cable length on the RHS of B.

DPL Version 1

A prototype of this initial design concept was modeled and constructed using 3D printing which allows for rapid prototyping and fast alterations. The design uses a cable to provide the flexibility and a chain and sprocket to transfer the torque to the shafts without slipping. The preliminary design is shown in Fig. 11. The cable passes through the links from the input (right) to the output (left) and it is constrained in the middle by the red guide, which is aligned with the shared point of rotation between the two parallelograms shown in Fig. 8 as point B. The links are bent at the common axis as was done in the DPL made by Bai et al. [10], and this angle was intended to be tailored to the size of the system in future prototypes.

This prototype was successful because it showed that the cable was a good way to transfer torque through the DPL. When the input and output axes, shown as black dotted lines in Fig. 4, were at a right angle to each other, the cable worked well to rotate the output shaft. The output shaft has a sprocket connecting the cable by a chain shown in Fig. 12. The cable allows the links of the DPL to move without affecting the cable tensioning so the cable could always transfer torque. There were several problems that occurred with the first prototype. While the chains could transfer a large amount of torque to the shafts without slipping, the chain could not bend with the DPL. A flexible solution had to be found that could transmit the torque of the cable to the shaft, while allowing the mechanism to open and close.

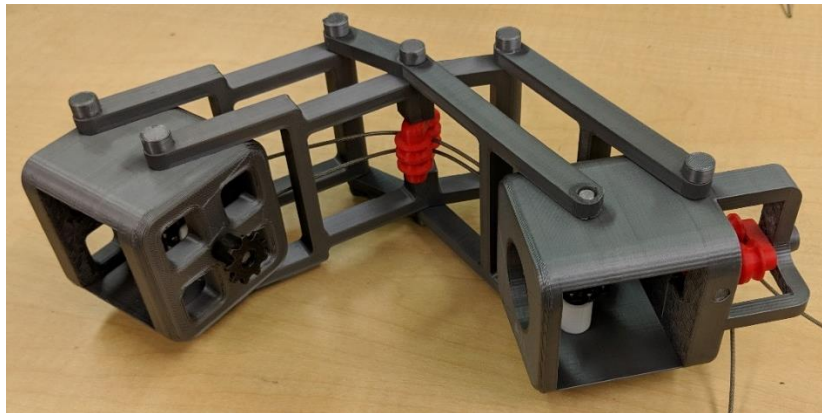


Figure 11: First DPL prototype.

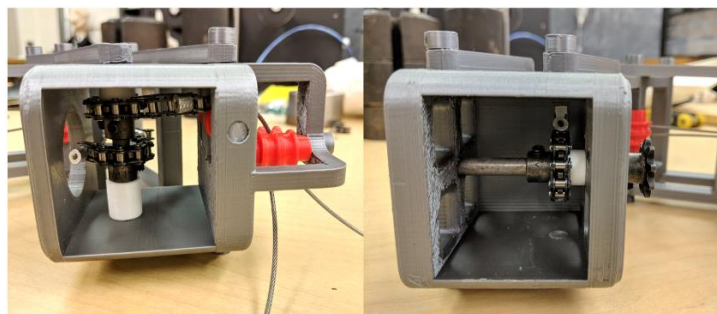


Figure 12: First DPL prototype input (right) and output (left).

DPL Version 2

The second version of the DPL focused on incremental upgrades to simplify the construction and optimize the shape. The length of the links was increased to allow for the system to fit on a larger body size as seen in Fig. 13. Testing was done to find the remote center of rotation and it was determined that with the bent links, the remote center of rotation would move as the links actuated. Because the input and out axes were aligned with the attachment points of the links, it was unnecessary to bend the links. A change was made to make the links straight and the center of rotation became constant.

The central cable guide was simplified to a single part with low friction guide holes instead of a roller as can be seen in Fig. 14. This was done to more precisely align the cable with the axis because a single part could have tighter tolerances. This improvement reduced cable movement and gave more consistent cable tension. The links were improved by removing the connecting support from between the top and bottom links, where the cable would pass through, so that there would not be any collisions between the cable and the links.

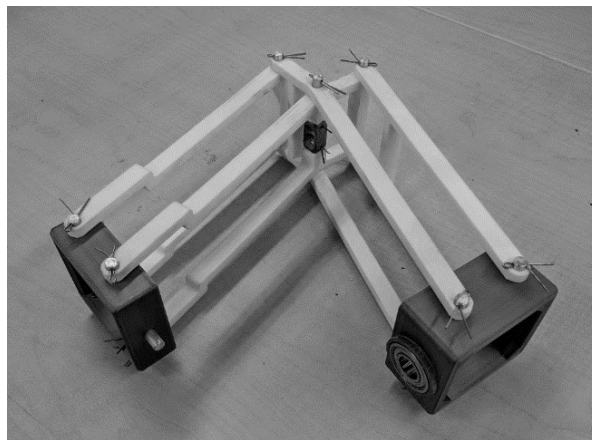


Figure 13: Second DPL prototype.

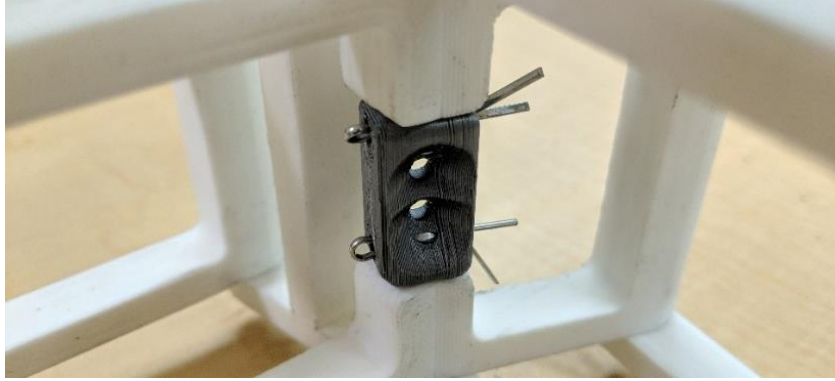


Figure 14: Simplified cable guide.

DPL Version 3

The third iteration of the design focused on improving the kinematics of the output shaft. The output shaft is the part of the mechanism that attaches to the user's arm and is driven by the motor. It is important that the output shaft is rotated by the motor directly and not effected by any other sources such as movement in the links. The chain used in the previous design had issues when the link was not at a right angle to the output shaft because the chain could not bend at the axis, where the link attached to the output enclosure. This caused tensioning and would lock up the system, and the chain could not rotate on the gear. A new design was implemented, which used a 3D-printed shaft with a crossbar that the cable would pass through as shown in Fig. 15. The crossbar has a hole through it that allows the cable to pass through it. As the input motor spins, the cable rotates the output shaft by pulling on the lever arm created by the crossbar as shown in Fig. 15. The problem with this design is that as the crossbar moves out of vertical alignment with the DPL axis (Point C on Fig. 4(a)), the cable is no longer in axis with the output enclosure pivot point and the links cannot rotate freely. If the crossbar is at an angle, the cable will prevent the DPL from moving with the user. The second problem

with this design is that as the crossbar rotates, the effective lever arm shortens as seen in Fig. 15 by the change from L_1 to L_2 . This means that as the motor rotates the output shaft, the mechanism will lose torque to raise the user's arm. This design also has a maximum theoretical rotation of 180 degrees. The shape and size of the crossbar was experimented with to try to find an optimal configuration.

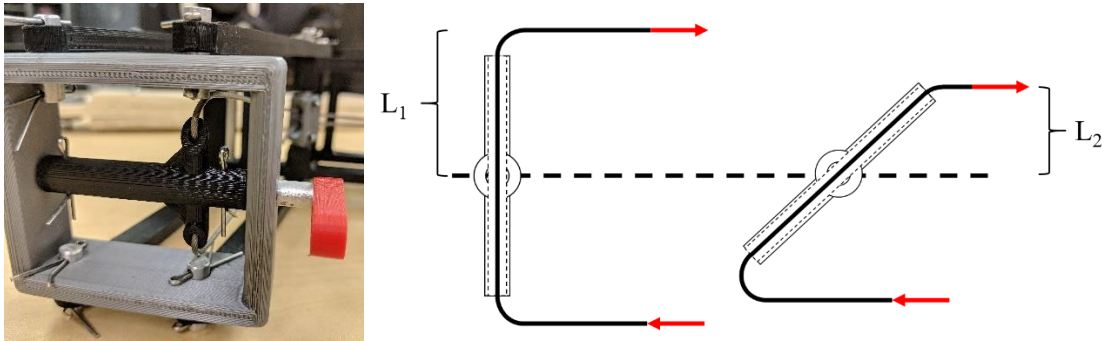


Figure 15: Crossbar output shaft with cable passing through center, (a) crossbar vertically aligned, (b) crossbar at some off-axis angle.

DPL Version 4

Version 4 of the DPL mechanism experimented with several design of the output cross-bar. It was found that the crossbar design was limited to 180 degrees of rotation due to the cable only being able to pull one side to a maximum angle of 90 degrees to either side. All other configurations had a similar issue, so this design was determined to not be suitable for the intended application.

DPL Version 5

Version 5 of the DPL with distal motor significantly improved the method for transferring torque to the output shaft by using the cable. Based on the previous designs, it was clear that a chain and gear would not be the ideal solution because of the inherent

stiffness in the chain. The crossbar design would not work because it changed its geometry based on the angle of the output shaft. The solution needed to allow the cable to have constant tension at any angle, rotate the output shaft, and not cause any backlash, which would move the shaft unintentionally. An output shaft design was constructed that fit all of these criteria and the resulting prototype is shown in Fig. 16.

The basis for the improved output pulley uses a smooth disk that the cable wraps around with a groove inside of the disk that the cable can wrap around as shown in Fig. 17. As one side of the cable is pulled on by the motor, it will cause the disk to rotate. The groove cutout bends the cable beyond its minimum operating bend radius, which effectively locks the cable in place inside the disk so that all tension causes a rotation of the disk. In Fig. 17, if the cable on the top is pulled to the right, the tension will cause the disk to rotate clockwise. This disk system can transmit much more torque without slipping than wrapping the cable around a pulley without the groove. Compared to a set screw, this grooved design has more range of motion because it can rotate more than 90 degrees to each side. This enables the design to transmit a large amount of torque by using the cable bend radius to lock it in place and have a large range of motion. The cable wraps around the disk, and there are no guides on the disk so that the cable can slide off the disk to the side at any point along the disk. This is so that at any angle of the output shaft rotation the DPL can actuate, and the cable will bend along the vertical axis of the output enclosure. To keep the cable from slipping off the disk shown in Fig. 17, there is a half-circle guide built into the casing that attaches to the output shaft. This creates a point that aligns with the vertical axis that constrains the cable while allowing it to rotate the output shaft, which attaches to the arm without unintended tensioning. As the DPL links

actuate, the cable bends at a single point as seen in Fig. 18. This output shaft design has multiple benefits: it does not cause tensioning, the cable can rotate the output at any DPL angle, and it has a higher torque capacity than a standard pulley setup.

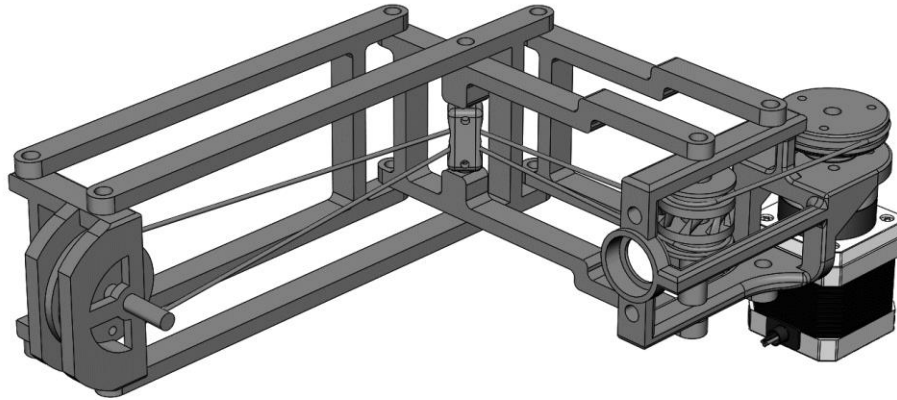


Figure 16: Version 5 of the DPL mechanism.

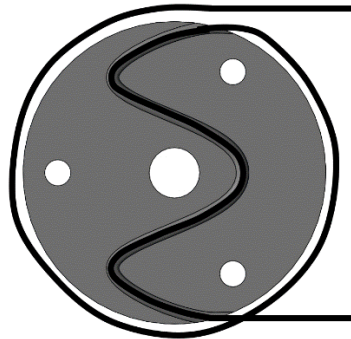


Figure 17: Output pulley with the cable in groove.

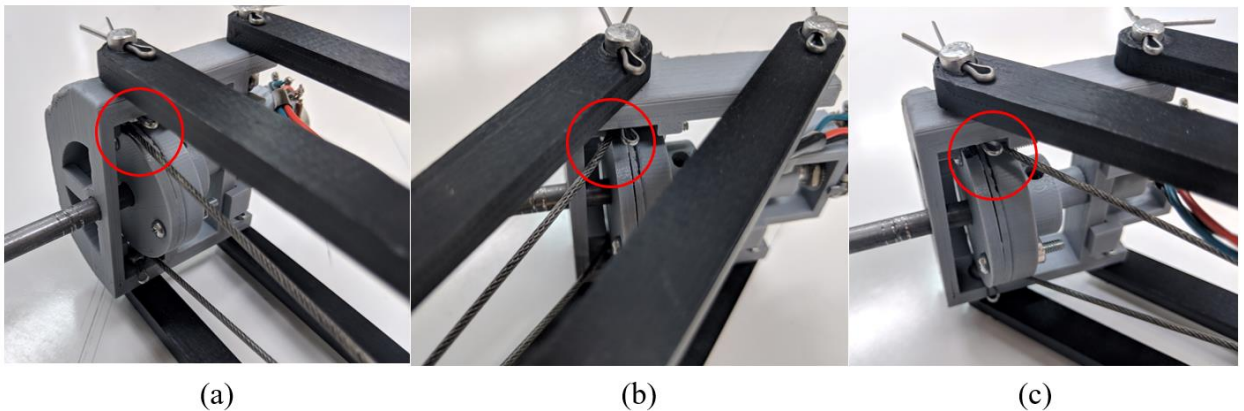


Figure 18: Output pulley cable bend point, (a) DPL at right angle, (b) DPL in open position, (c) DPL in closed position.

The input enclosure for this version of the design used an input pulley that was mounted in the transverse plane which is the same plane that the DPL actuates in. A test was performed for this configuration in three positions: 90 degrees, fully open, and fully closed. The three positions of the DPL are shown in Fig. 19. The test measured the rotation of the output shaft compared to the input shaft to measure how well the output shaft follows the input shaft. This is important to see how well the cable is able to transmit motion while the DPL is at different angles. The results of the test are shown in Figs. 20-22. The ideal response is a line with a slope given by the ratio between the diameters of the input and output pulleys. The output shaft should respond directly with the rotation of the motor that connects to the input shaft. If the motion is ideal, then the actual rotation will follow the ideal line. The results in Fig. 20 and 21 have the experimental results lagging behind the ideal response as shown by the horizontal distance from the ideal response to the actual response. This means that there is a lag between the input and output pulleys due to the tensioning of the cable. If the cable does not have enough tension, it will need to become tensioned by the motor before it can rotate the output shaft. In Fig. 22, there is no lag in the system because the DPL is closed, and the bend radius of the cable causes the response to be more direct. The experimental response of the output shaft is above the ideal response, showing that the output shaft rotated before the motor turned on. This means that when the DPL closes, corresponding to an external rotation, the cable is rotating the output shaft due to the tension created and causing backlash. This is not a desirable result as it means that the DPL is not isolated from the rotation of the motor. It was determined that the cause of these issues came from the input pulley being in the transverse plane with the DPL. Since both the DPL and input

pulley were in the same plane, they were affecting each other unintentionally. To fix this, the input shaft needs to be turned out-of-plane with the DPL actuation.

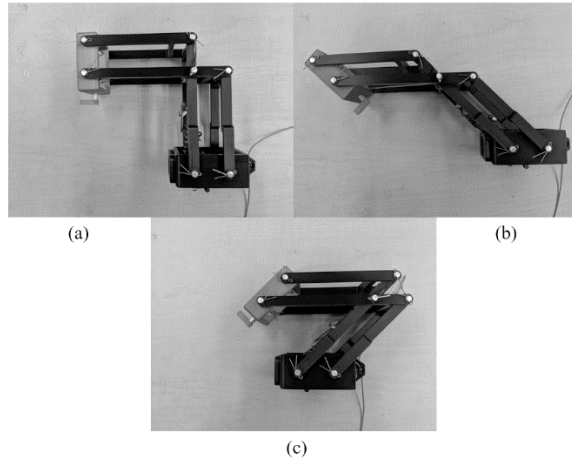


Figure 19: Actuation of DPL in throughout the internal/external rotation, (a) 90° position, (b) open position, (c) closed position.

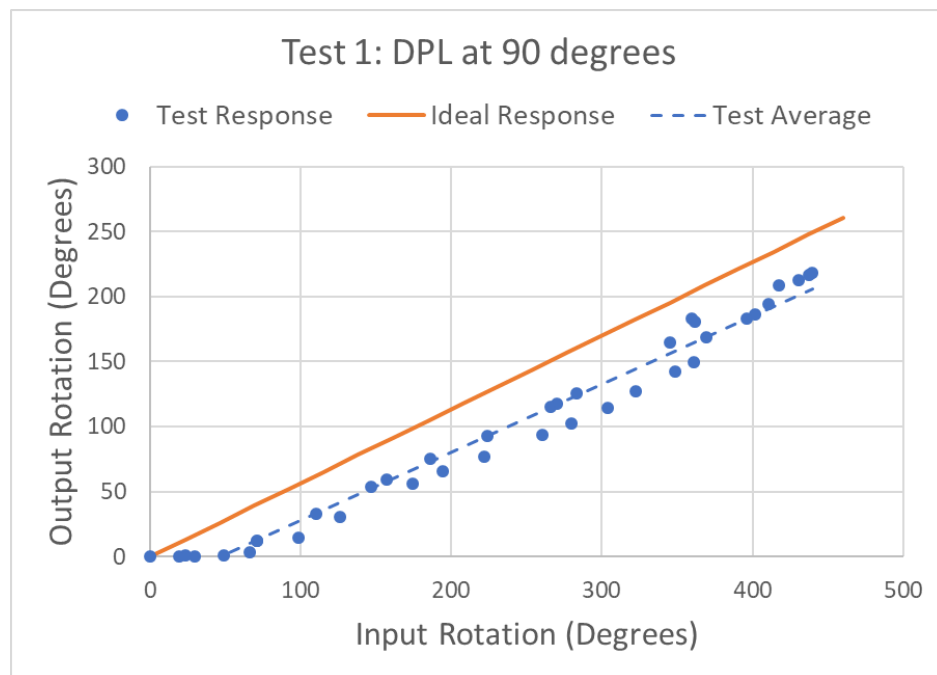


Figure 20: Experimental results of input to output rotation of DPL at 90° angle.

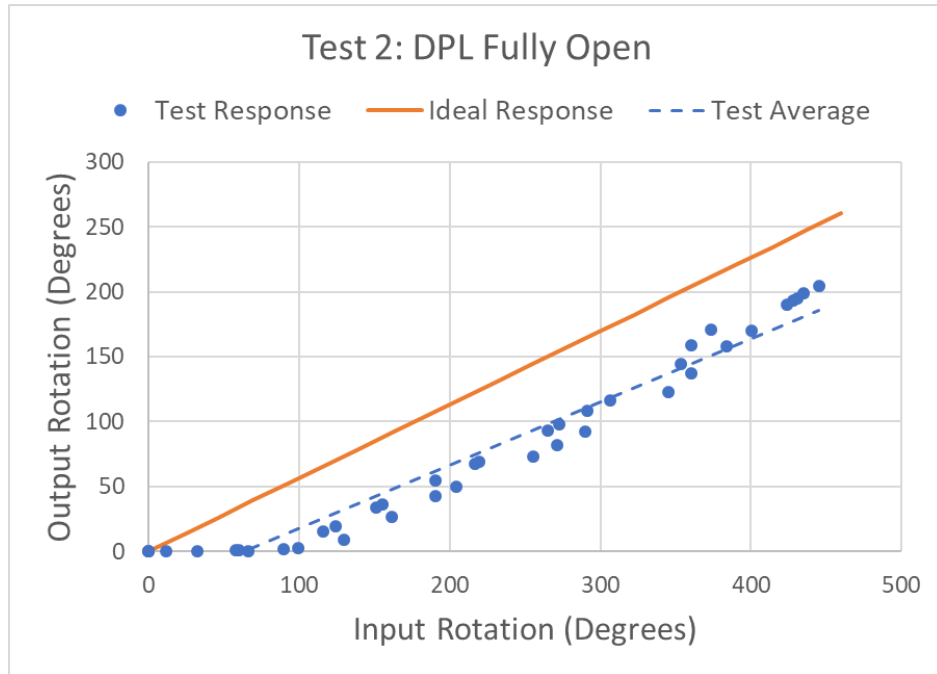


Figure 21: Experimental results of input to output rotation of DPL at fully open position.

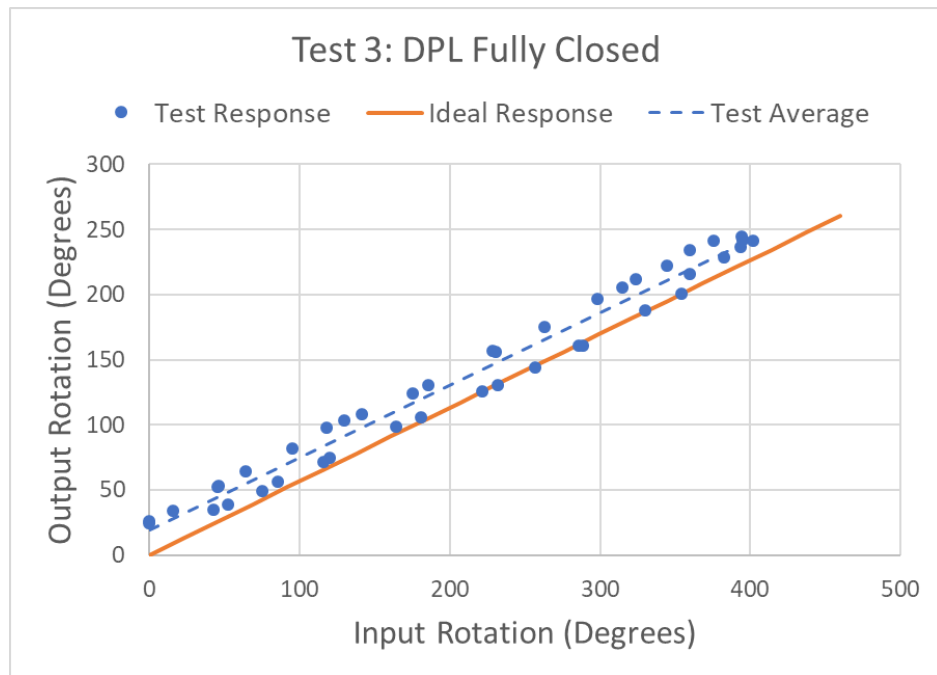


Figure 22: Experimental results of input to output rotation of DPL at fully closed position.

DPL Version 6

Based on the test data, the design of the input enclosure needed to be changed to have the pulley out-of-plane with the transverse plane to isolate the motion of the DPL from the motor and cable. The input enclosure was changed to use the same design as the output so that the pulley would be in the coronal plane and not affect the motion of the DPL as shown in Fig 23. The motor was mounted vertically and had a second cable that would transmit the torque to the main pulley as shown in Fig. 24. By changing the plane of the input pulley, the DPL did not experience any backlash and the motion of the cable was isolated from all other motions. The design was also made modular so that the motor mount could be swapped out for any other motor mounts, which made switching motors easier. The adapter that connects to the backpack was also made detachable so that any other type of adapters could be used instead. This iteration had a problem with the friction of the double cable layout. Any bend in the cable causes friction because the cable needs to be in tension and it rubs against the plastic guide. Because there were two cables, there was extra friction in the system that caused slipping of the secondary cable connected to the motor. By having two cables, it also made it very difficult to align them. An improvement needed to be made to remove the second cable and the extra complexity.

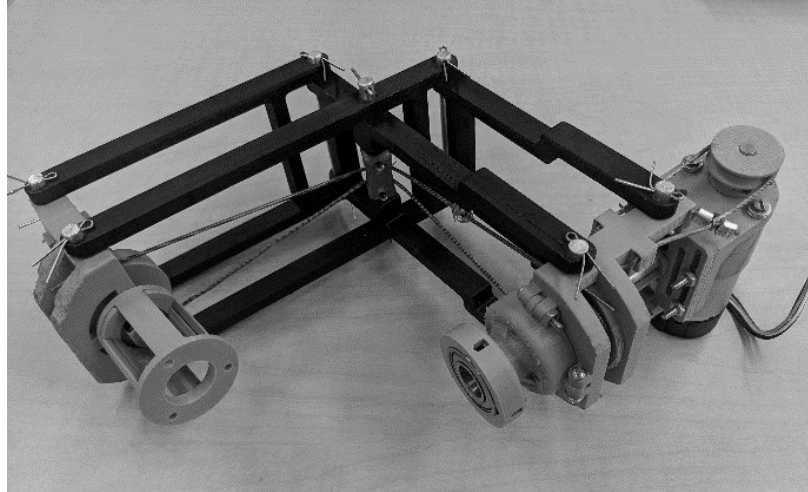


Figure 23: DPL mechanism version 6.

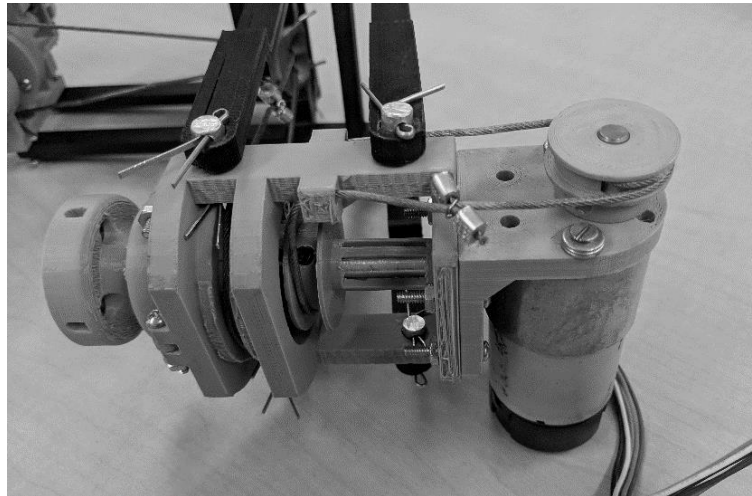


Figure 24: Input pulley is mounted out of plane from the DPL and a second cable transfers the torque from the motor.

DPL Version 7

Version 7 of the mechanism is the most recent and final design of the DPL made for this thesis. It makes several improvements over the previous iteration for simplifying the input enclosure. The new design mounts the motor horizontally closer to the user's back in an effort to make the system more compact. The final design is shown mounted on the user in Fig. 25. The secondary cable is eliminated by putting the large input pulley

directly on the motor, shown in Fig. 26, and removing the complexity of the semicircle guide and replacing it with a passive hole guide similar to the central cable guide seen in Fig. 14. By simplifying the construction of the input, it makes the design more robust and reduces friction. The new design is shown in Fig. 27. For testing purposes, a potentiometer is built into the system at the motor shaft and output shaft to measure the rotation and determine how responsive the cable is. The mechanism is mounted on a backpack and aligned with the user's shoulder as shown in Fig. 28. The cable is constrained by two cables guides at point A and B. This design simplifies how the cable aligns with the axes and reduces the friction of the cable and the cable guide, shown in Fig. 29.

This design improves upon the previous iteration by simplifying the construction and the mounting point of the motor also provides a gravity compensation mechanism by mounting on the opposite side of the backpack mounting point. Fig. 28(b) shows the motor on the left side of the input axis and the DPL is on the right side of the input axis. The input axis is the point where the system mounts to the backpack and is the balance point of the mechanism. By using the motor as a gravity balance, the weight of the mechanism resting on the user's arm is reduced. Since the system only has a single motor, the coronal plane axis is a passive DOF. This means that if the motor was mounted at point C in Fig. 28(a), the user would feel the full weight of the motor. By mounting the motor on the opposite side of point A, it balances the weight that the user feels on their arm and the entire weight of the system is transferred into the backpack which is closer to the body's center of gravity. Another benefit of mounting the motor on the back is that it reduces the inertia of the system as the DPL moves in the transverse

plane. The user will not have to fight the inertia of the mechanism because the weight of the moving components will be reduced by mounting the motor on the back. This design has improved balance over the previous iteration and is a less fatiguing mechanism to wear.



Figure 25: Rendering of the exoskeleton mounted on the user.

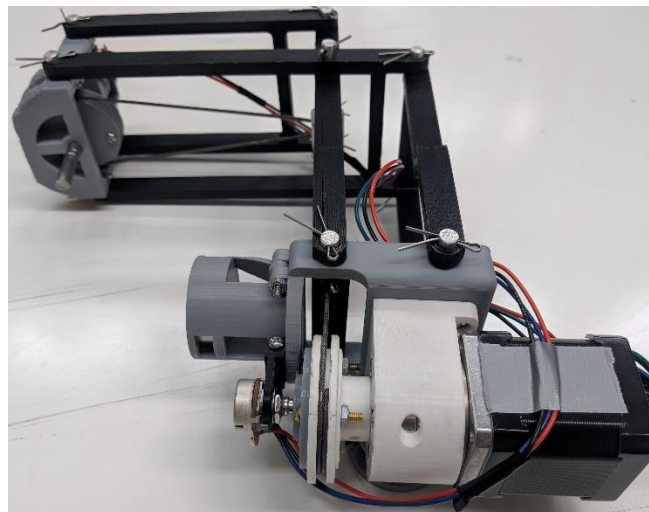


Figure 26: Input cable mounts directly to the motor shaft.



Figure 27: Final design for distal motor DPL.

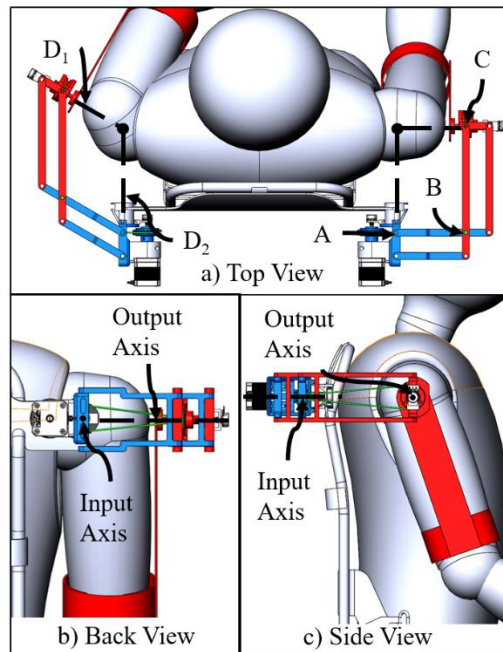


Figure 28: Three-views of the double parallelogram mechanism on the shoulder, (a) top view with the left arm at an angle, (b) back view and, (c) side view.

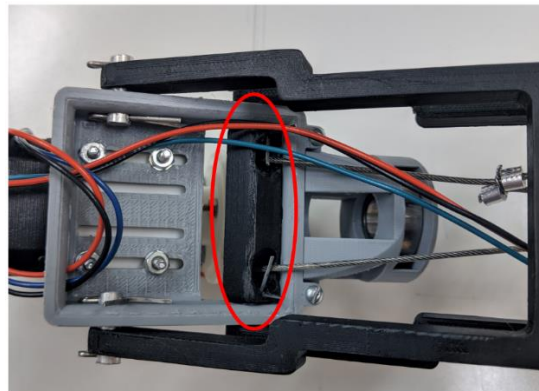


Figure 29: Input cable guide aligns the cable along the vertical axis of DPL.

EXPERIMENTS AND RESULTS:

Motor Functionality

Testing was done on Version 7 of the DPL mechanism to test the rotation of the motor, output shaft, and a dummy arm to verify the powered flexion/extension DOF. The test was performed by mounting the exoskeleton to a dummy arm that had a 3 DOF hinge to simulate the rotation of a human shoulder joint. The testing arm had a low weight so that the motor could lift it and the results would be accurate to the exact motor rotation. A potentiometer was mounted on the dummy arm at the axis of flexion/extension, which is the motion that the exoskeleton powers. The testing setup is shown in Fig. 30 with the exoskeleton connected. An adapter was made that would extend the output shaft enough to connect to the dummy arm, but this would not be present when a person wears it. An Arduino Mega 2560 was used to power the stepper motor and a second Arduino

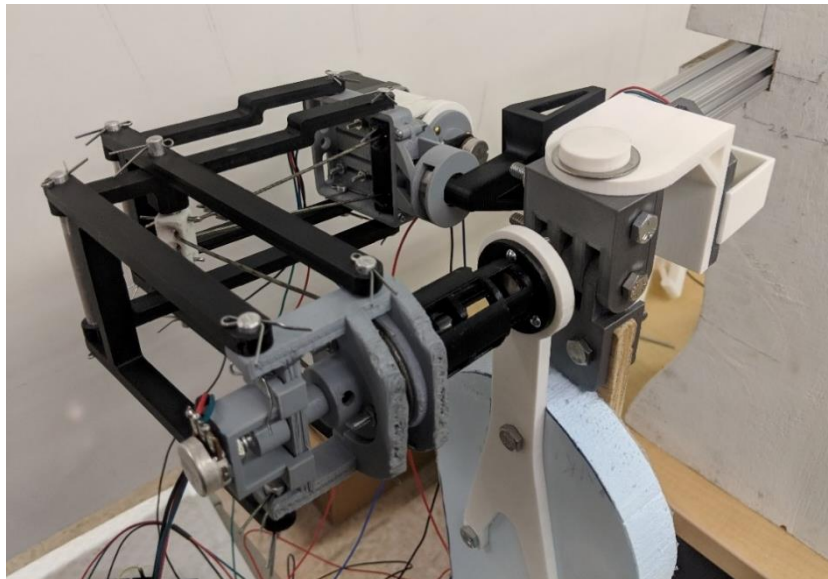


Figure 30: Experimental setup for testing exoskeleton rotation.

measured the potentiometer readings. The Arduino code spins the motor for 90 degrees back and forth to create an arm swinging motion (shown in Appendix A). A second Arduino Mega 2560 handles the measurement of the three potentiometers by using a Visual Basic (VB) interface to record the potentiometer measurements coming into the Arduino. The VB interface and the code are shown in Appendix B and the Arduino code to connect with the VB interface is shown in Appendix C. The results were averaged on a 5-point moving average to smooth out the measurement noise. The test results shown in Fig. 31. The rotation of the motor reaches the most extreme angles of rotation because it is powering the system. The output shaft rotates by the cable transferring the torque through the DPL and the rotation is slightly less than the motor. The reason for the output shaft not getting the same amount of rotation as the motor can be due to several factors including cable tensioning, tolerancing, and imperfect pulley diameters that are not perfectly circular. The dummy arm does not rotate as much as the motor and the output shaft, and this is due to tolerancing in the test setup causing the potentiometer to follow the rotation but with less overall rotation.

These factors can be mitigated by testing the prototype further and improving and design limitations. However, it is not critical to refine the design to make the rotations line up perfectly because of the intended purpose. The control system was not explored in this thesis, but a feedback control system that follows the rotation of the user's arm could be used to mitigate the effects of tolerancing in the system. By designing a control system that rotates the motor as much as needed to move the user's arm, the motor will rotate until the control system tells the motor to stop. The user's arm will cause the motor to move, whereas during testing the motor caused the arm to move. By designing a control

system that uses the user's arm to control the motor, then the small misalignments seen during testing will not matter.

This design has a passive abduction/adduction movement and allows for internal/external rotation. It was observed that the design had unintended motion during testing as the DPL fully opened. When the DPL is fully open as seen in Fig. 19(b), the input and axis become opposite each other, and their axes become close to parallel. This happens because the DPL allows the output axis to follow the arm during internal/external rotation. As the axes start to align, they effectively become the same axis. Since the output is being powered by the motor to lift the arm, but the input is free to rotate, the motor will rotate the exoskeleton about the input axis when these axes become aligned. Because the arm has significant resistance, the motor will lift up the exoskeleton instead of the arm and the arm will not move if these axes are aligned. It requires further testing to determine the severity of this effect. This effect is present on any design that aligns an active DOF with a passive DOF because the DOFs will align and become a single DOF. However, for the intended application, this effect is an extreme case and is not expected to occur during normal operation.

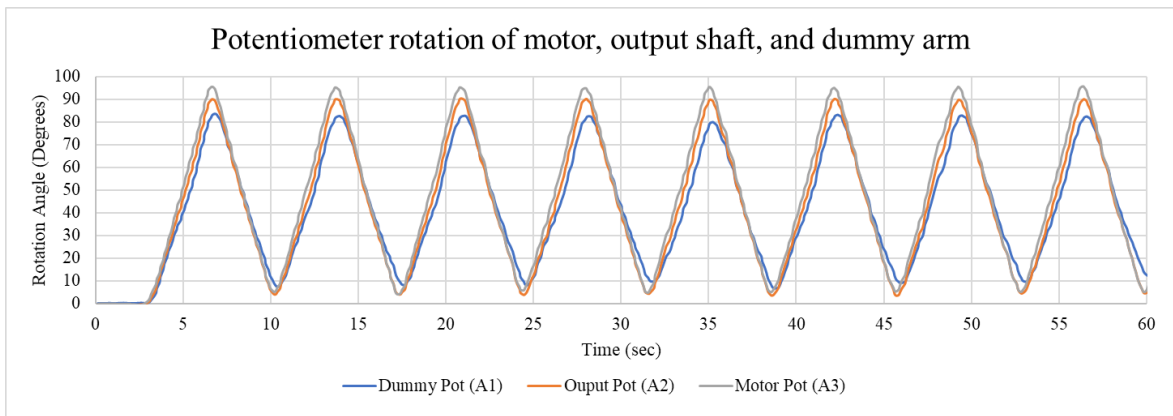


Figure 31: Rotation angles of the motor input shaft, the output shaft, and the dummy's arm.

Workspace Analysis

The range-of-motion of the design was simulated in the transverse plane to show the workspace of the DPL mechanism. As the arm rotates in internal/external rotation, the mechanism will move with the arm in accordance with the size of the links. The DPL uses links to maintain a constant center-of-rotation, so the motion of the end effector can be tracked. The DH parameters of the mechanism are shown in Fig. 32 and the values of the parameters are shown in Table 1. Only θ_2 and θ_3 are changed as they correspond with the motion of the DPL in the transverse plane. The angle limits are determined by using the CAD model of the mechanism and measuring the angles of the end-of-travel positions. The result of the workspace analysis is shown in Fig. 33 as a curve in the transverse plane which the end-effector of the exoskeleton traces as it moves in internal/external rotation. The red star shows the remote center of rotation, where θ_4 in Fig. 32 traces an arc around and the blue star that shows the position of the mounting point θ_2 in Fig. 32.

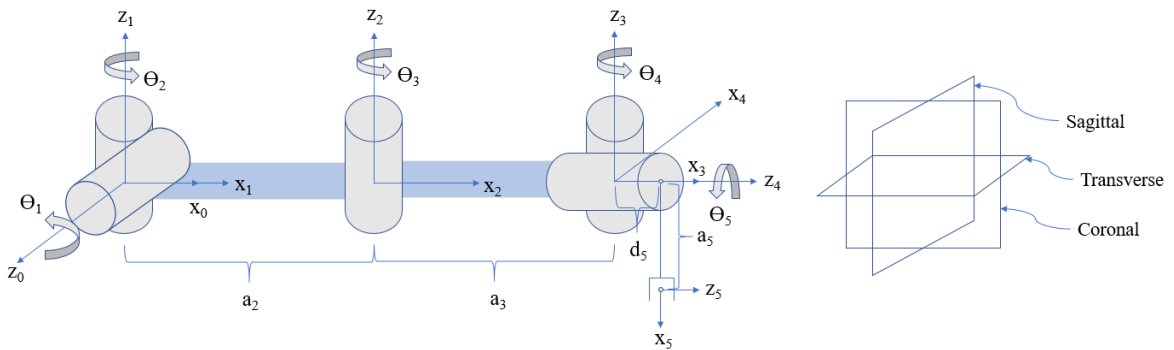


Figure 32: The DH parameter diagram of the exoskeleton prototype in operational configuration.

i	a_i	d_i	α_i	θ_i
1	0	0	$-\frac{\pi}{2}$	θ_1^*
2	$a_2 = 4.45 \text{ in}$	0	0	θ_2^*
3	$a_3 = 4.8 \text{ in}$	0	0	θ_3^*
4	0	0	$\frac{\pi}{2}$	$\theta_4^* + \frac{\pi}{2}$
5	$a_5 = 9.5 \text{ in}$	d_5	0	$\theta_5^* - \frac{\pi}{2}$

Table 1: Table of values for the DH parameters.

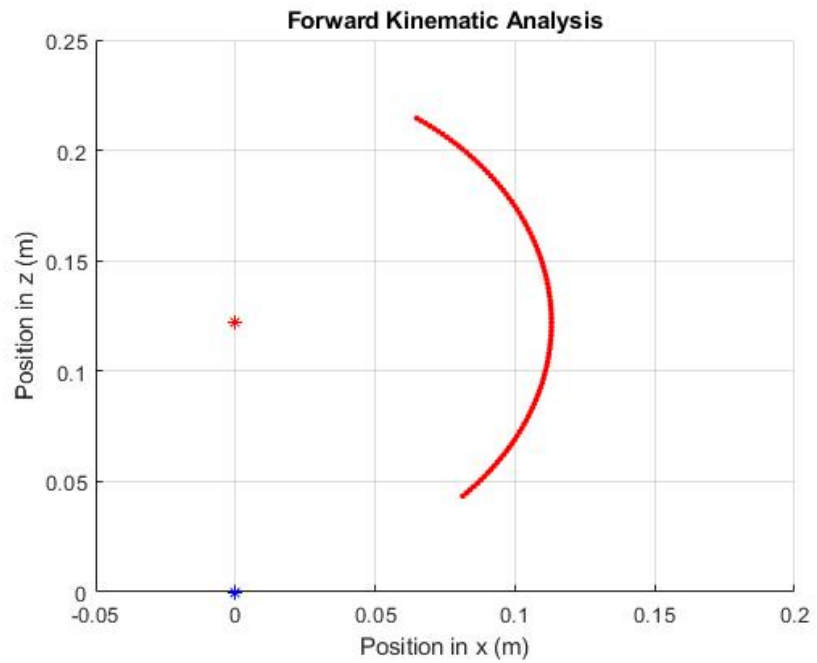


Figure 33: Workspace of DPL end effector

CONCLUSION AND FUTURE WORK:

The final prototype achieves the goal of the original design philosophy, which was to provide rotation assistance in the sagittal plane while not encumbering the user with extra weight on their arm. The newest version of the DPL design solves most of the problems of the original design and makes the system more adaptable because of the swappable motor mount and the backpack adapter. By mounting the motor distally from the point where it assists the user, the weight of the exoskeleton is brought closer to the user's center of gravity to make it less fatiguing to wear. This is achieved with a pulley system inside of a DPL mechanism. The motor provides a gravity balance that further decreases the weight on the user's arm and make the design more suited for rehabilitation because the design will not put extra weight on the arm that needs help moving.

In the future, the design will need to be refined and a motor can be selected to provide the desired torque to assist the arm. The primary refinements that should be made are sizing the links to fit a person, using a continuous cable without any break points that will impede rotation, and optimizing the pulley ratios. A prototype will need to be fabricated that can mount on a backpack and testing can be done to determine if the system can handle the torque that needs to be generated. Testing will need to be done on a person by mounting it and further refining the design. The prototype presented in this thesis gives a geometrically valid solution for mounting a distal motor to a DPL system with 3 DOFs and using cables to transfer the torque.

REFERENCES

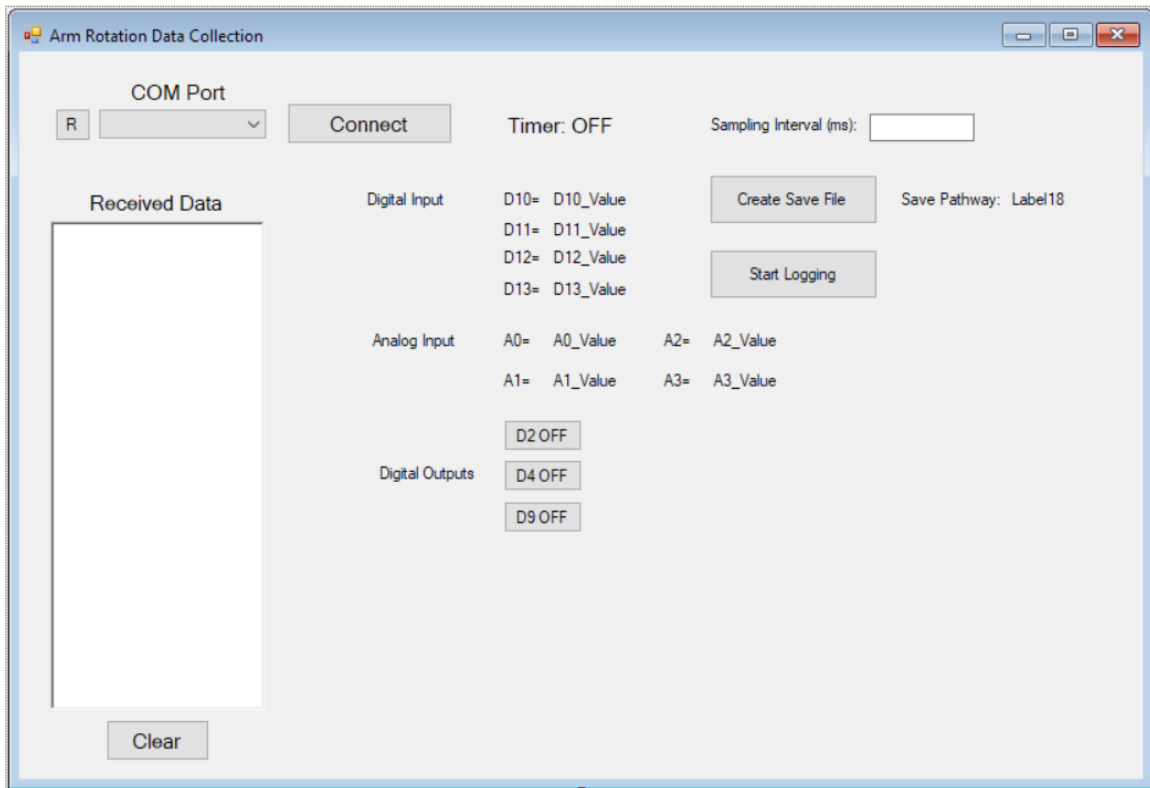
- [1] B. Hejrati, A. Merryweather, and J. J. Abbott, "Generating Arm-swing Trajectories in Real-time Using a Data-driven Model for Gait Rehabilitation with Self-selected Speed," *IEEE Transactions on Neural systems and Rehabilitation Engineering* 26, no. 1 (2018): 115-124.
- [2] O. R. Barnes, B. Hejrati, and J. J. Abbott, "An Underactuated Wearable Arm-swing Rehabilitator for Gait Training," in *2015 IEEE Int. Conf. Robotics and Automation (ICRA)*, pp. 4998-5003. IEEE, 2015.
- [3] I. Galiana, F. L. Hammond, R. D. Howe, and M. B. Popovic, "Wearable soft robotic device for post-stroke shoulder rehabilitation: Identifying misalignments," in *2012 IEEE/RSJ Int. Conf. on Intel. Rob. and Syst.*, pp. 317-322. IEEE, 2012.
- [4] B. K. Dinh, M. Xiloyannis, L. Cappello, C. W. Antuvan, S. C. Yen, and L. Masia, "Adaptive backlash compensation in upper limb soft wearable exoskeletons," *Rob. Auton. Syst.*, 92 (2017), 173-186.
- [5] H. S. Lo and S. S. Q. Xie, "Optimization of a redundant 4R robot for a shoulder exoskeleton," in *2013 IEEE/ASME Int. Conf. on Adv. Int. Mechatronics*, pp. 798-803. IEEE, 2013.
- [6] H. C. Hsieh, D. F. Chen, L. Chien, and C. C. Lan, "Design of a Parallel Actuated Exoskeleton for Adaptive and Safe Robotic Shoulder Rehabilitation," *IEEE/ASME Trans. Mechatronics* 22, no. 5 (2017): 2034-2045.
- [7] C. Liu, C. Zhu, H. Liang, M. Yoshioka, Y. Murata, and Y. Yu, "Development of a light wearable exoskeleton for upper extremity augmentation," in *2016 23rd Int. Conf. on Mechatronics and Machine Vision in Practice (M2VIP)*, pp. 1-6. IEEE, 2016.
- [8] D. Sui, J. Fan, H. Jin, X. Cai, J. Zhao, and Y. Zhu, "Design of a wearable upper-limb exoskeleton for activities assistance of daily living," in *2017 IEEE Int. Conf. on Adv. Intel. Mechatronics (AIM)*, pp. 845-850. IEEE, 2017.
- [9] A. Ebrahimi, "Stuttgart Exo-Jacket: An exoskeleton for industrial upper body applications," in *2017 10th Int. Conf. on Human Syst. Interactions (HSI)*, pp. 258-263. IEEE, 2017.
- [10] S. Bai, S. Christensen, and M. R. U. Islam, "An upper-body exoskeleton with a novel shoulder mechanism for assistive applications," in *2017 IEEE Int. Conf. on Adv. Intel. Mechatronics (AIM)*, pp. 1041-1046. IEEE, 2017.
- [11] S. Christensen and S. Bai, "Kinematic Analysis and Design of a Novel Shoulder Exoskeleton Using a Double Parallelogram Linkage," *J. Mech. Robot.* 10, no. 4 (2018): 41008.

APPENDICES

APPENDIX A

```
/*  
Wiring: Connect ENA+, DIR+, and PUL+ to the Mega's +5V pin  
        Connect PUL- to D13, ENA- to D4, and DIR- to D2  
  
Enable - HIGH locks the rotor, LOW free wheels it regardless of pulsing  
  
TB6600 Driver pins set 1,2,3 to OFF and 4,5,6 to ON  
  
*/  
  
// the setup function runs once when you press reset or power the board  
  
int PUL=2; //define Pulse pin  
int DIR=4; //define Direction pin  
int ENA=9; //define Enable Pin  
  
void setup() {  
  pinMode (PUL, OUTPUT);  
  pinMode (DIR, OUTPUT);  
  pinMode (ENA, OUTPUT);  
}  
  
void loop() {  
  for (int i=0; i<16576; i=i+1) //Forward 6400 steps  
  {  
    digitalWrite(DIR,LOW);  
    digitalWrite(ENA,HIGH);  
  
    digitalWrite(PUL,HIGH);  
    delayMicroseconds(200);  
    digitalWrite(PUL,LOW);  
    delayMicroseconds(200);  
  }  
  for (int i=0; i<16576; i=i+1) //Backward 6400 steps  
  {  
    digitalWrite(DIR,HIGH);  
    digitalWrite(ENA,HIGH); //LOW causes a pause and more forward rotation  
    digitalWrite(PUL,HIGH);  
    delayMicroseconds(200);  
    digitalWrite(PUL,LOW);  
    delayMicroseconds(200);  
  }  
}
```

APPENDIX B



Module Module1

Public filepathway As String
Public interval As Double
Public log As Boolean
Public start_time As Double

End Module

'Developed from the work of martyncurrey.com
'Developed from the work of martyncurrey.com
'Developed from the work of martyncurrey.com
,

'Arduino Wiring Assumption -
,

'D0,D1 - Not used. Reserved for communications
'A0,A1 - Implemented analog inputs
'D2,D4,D9 - Digital outputs
'D10,D11,D12,D13 - Digital inputs
'D7,D8 - Reserved for Adafruit Ultimate GPS Logger Shield use

```
'D3,D5 - PWM analog outputs
'D6 - Servo output
'
```

```
Imports System
Imports System.IO.Ports
```

```
Public Class Crosby_PC_Control
```

```
    ' Global variables
    Dim comPORT As String
    Dim receivedData As String = ""
    Dim connected As Boolean = False
    Dim count = 0
    Dim File_Opened As Integer
```

```
    Dim Analog_in_A0 As String = " "
    Dim Analog_in_A1 As String = " "
    Dim Analog_in_A2 As String = " "
    Dim Analog_in_A3 As String = " "
    Dim Digital_out_D9 As String = " "
    Dim Digital_out_D4 As String = " "
    Dim Digital_out_D2 As String = " "
    Dim Servo_D6_text As String = " "
    Dim Analog_out_D3 As String = " "
    Dim Analog_out_D5 As String = " "
    Dim Digital_in_D10 As String = " "
    Dim Digital_in_D11 As String = " "
    Dim Digital_in_D12 As String = " "
    Dim Digital_in_D13 As String = " "
    Dim Filename As String = " "
    Dim start_time As Double
    Dim current_time As Double
```

```
    ' When the program starts; make sure the timer is off (not really needed) and add the
    available COM ports to the COMport drop down list
```

```
    Private Sub Crosby_PC_Control_Load(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles MyBase.Load
```

```
        Sampling_timer.Enabled = False
        D10_Value_lbl.Text = "LOW"
        D11_Value_lbl.Text = "LOW"
        D12_Value_lbl.Text = "LOW"
        D13_Value_lbl.Text = "LOW"
        File_Opened = 0
```

```

    start_time = (DateTime.Now - New DateTime(1970, 1, 1, 0, 0, 0)).TotalSeconds
'unix time
    populateCOMport()
End Sub

```

```

'The refresh button updates the COMport list
Private Sub refreshCOM_BTN_Click(sender As Object, e As EventArgs) Handles
refreshCOM_CB_BTN.Click
    SerialPort1.Close()
    populateCOMport()
End Sub

```

```

Private Sub populateCOMport()
    comPORT = ""
    comPort_ComboBox.Items.Clear()
    For Each sp As String In My.Computer.Ports.SerialPortNames
        comPort_ComboBox.Items.Add(sp)
    Next
End Sub

```

```

Private Sub comPort_ComboBox_SelectedIndexChanged(sender As Object, e As
EventArgs) Handles comPort_ComboBox.SelectedIndexChanged
    If (comPort_ComboBox.SelectedItem <> "") Then
        comPORT = comPort_ComboBox.SelectedItem
    End If
End Sub

```

' When the Connect button is clicked; if a COM port has been selected, connect and send out a HELLO message.

' Then wait for the Arduino to respond with its own HELLO.

' When the HELLO is received we are connected; change the button text to Disconnect.

```

Private Sub connect_BTN_Click(sender As Object, e As EventArgs) Handles
connect_BTN.Click
    comPORT = comPort_ComboBox.SelectedItem
    If (connect_BTN.Text = "Connect") Then
        If (comPORT <> "") Then
            SerialPort1.Close()
            SerialPort1.PortName = comPORT
            SerialPort1.BaudRate = 9600
            SerialPort1.DataBits = 8
            SerialPort1.Parity = Parity.None
            SerialPort1.StopBits = StopBits.One
            SerialPort1.Handshake = Handshake.None
            SerialPort1.Encoding = System.Text.Encoding.Default
            SerialPort1.ReadTimeout = 10000

```



```

SerialPort1.Open()

'See if the Arduino is there
count = 0
SerialPort1.WriteLine("<HELLO>")
connect_BTN.Text = "Connecting..."
connecting_Timer.Enabled = True
Else
    MsgBox("Select a COM port first")
End If
Else
    'connect_BTN.Text = "Dis-connect"
    'close the connection a reset the button and timer label
    Sampling_timer.Enabled = False
    Timer_LBL.Text = "Timer: OFF"
    SerialPort1.Close()
    connected = False
    connect_BTN.Text = "Connect"
    populateCOMport()
End If

```

End Sub

```

'The connecting_Timer waits for the Arduino to say HELLO.
' If HELLO is not received in 2 seconds display an error message.
' The connecting_Timer is only used for connecting

```

```

Private Sub connecting_Timer_Tick(sender As Object, e As EventArgs) Handles
connecting_Timer.Tick

```

```

    connecting_Timer.Enabled = False
    count = count + 1

```

```

If (count <= 8) Then
    receivedData = receivedData & ReceiveSerialData()

```

```

If (Microsoft.VisualBasic.Left(receivedData, 5) = "HELLO") Then
    'if we get an HELLO from the Arduino then we are connected
    connected = True
    connect_BTN.Text = "Dis-connect"
    Sampling_timer.Enabled = True
    Timer_LBL.Text = "Timer: ON"
    receivedData = ReceiveSerialData()
    receivedData = ""
    SerialPort1.WriteLine("<START>")

```

```

Else
    'start the timer again and keep waiting for a signal from the Arduino
    connecting_Timer.Enabled = True
End If

Else
    'time out (8 * 250 = 2 seconds)
    RichTextBox1.Text &= vbCrLf & "ERROR" & vbCrLf & "Can not connect" &
vbCrLf
    connect_BTN.Text = "Connect"
    populateCOMport()
End If

End Sub

'After a connection is made the main timer waits for data from the Arduino
'Only OK1, OK2 or OK3 is available.
Private Sub Timer1_Tick(sender As Object, e As EventArgs) Handles
Sampling_timer.Tick
    receivedData = ReceiveSerialData()
    RichTextBox1.Text &= receivedData

    Dim tmp As String
    tmp = Microsoft.VisualBasic.Left(receivedData, 3)

    'If receivedData contains a "<" and a ">" then we have data
    If ((receivedData.Contains("<") And receivedData.Contains(">"))) Then
        parseData()
    End If

    'Tell the Arduino to send the analog pin values
    SerialPort1.WriteLine("<SA>")

    'Tell the Arduino to send the digital input pin values
    SerialPort1.WriteLine("<SD>")

    'Put code that needs to repeat at every data sampling here

    If log = True Then
        Dim current_time As Double = (DateTime.Now - New DateTime(1970, 1, 1, 0, 0,
0, 0)).TotalSeconds
        Dim elapsed As Double = current_time - start_time
        Dim Dummy_Pot As Double = A1_Value_lbl.Text
        Dim Arm_Pot As Double = A0_Value_lbl.Text

```

```

Dim Motor_Pot As Double = A2_Value_lbl.Text
Dim Dummy_Vertical_Pot As Double = A3_Value_lbl.Text

My.Computer.FileSystem.WriteAllText(filepathway, Format(elapsed) + "," +
Format(Dummy_Pot) + "," + Format(Arm_Pot) + "," + Format(Motor_Pot) + vbCrLf,
True)

ElseIf log = False Then

End If

'End of sampling code

End Sub

Function ReceiveSerialData() As String
Dim Incoming As String
Try
Incoming = SerialPort1.ReadExisting()
If Incoming Is Nothing Then
Return "nothing" & vbCrLf
Else
Return Incoming
End If
Catch ex As TimeoutException
Return "Error: Serial Port read timed out."
End Try

End Function

'Clear the RecievedDtaa test box
Private Sub clear_BTN_Click(sender As Object, e As EventArgs) Handles
clear_BTN.Click
RichTextBox1.Text = ""
End Sub

'Toggle the digital output states

Private Sub Pin9_BTN_Click(sender As Object, e As EventArgs) Handles
Pin9_BTN.Click
If (connected) Then
If (Pin9_BTN.Text = "D9 OFF") Then
SerialPort1.WriteLine("<P009ON>")
Pin9_BTN.Text = "D9 ON"
Else

```

```

        SerialPort1.WriteLine("<P009OF>")
        Pin9_BTN.Text = "D9 OFF"
    End If
Else
    MsgBox("Not connected")
End If
End Sub

```

```

Private Sub Pin4_BTN_Click(sender As Object, e As EventArgs) Handles
Pin4_BTN.Click
    If (connected) Then
        If (Pin4_BTN.Text = "D4 OFF") Then
            SerialPort1.WriteLine("<P004ON>")
            Pin4_BTN.Text = "D4 ON"
        Else
            SerialPort1.WriteLine("<P004OF>")
            Pin4_BTN.Text = "D4 OFF"
        End If
    Else
        MsgBox("Not connected")
    End If
End Sub

```

```

Private Sub Pin6_BTN_Click(sender As Object, e As EventArgs) Handles
Pin2_BTN.Click
    If (connected) Then
        If (Pin2_BTN.Text = "D2 OFF") Then
            SerialPort1.WriteLine("<P002ON>")
            Pin2_BTN.Text = "D2 ON"
        Else
            SerialPort1.WriteLine("<P002OF>")
            Pin2_BTN.Text = "D2 OFF"
        End If
    Else
        MsgBox("Not connected")
    End If
End Sub

```

'The OK button acts as soon as the button is clicked. It does not wait for the user to release the button

'There is no of for the OK button

```

Private Sub ok_BTN_MouseDown(sender As Object, e As MouseEventArgs)
    If (connected) Then
        SerialPort1.WriteLine("<NKOK>")
    End If

```

End Sub

Function parseData()

' uses the global variable receivedData

Dim pos1 As Integer

Dim pos2 As Integer

Dim length As Integer

Dim newCommand As String

Dim done As Boolean = False

While (Not done)

pos1 = receivedData.IndexOf("<") + 1

pos2 = receivedData.IndexOf(">") + 1

'occasionally we may not get complete data and the end marker will be in front of the start marker

' for exampe "55><"

' if pos2 < pos1 then remove the first part of the string from receivedData

If (pos2 < pos1) Then

receivedData = Microsoft.VisualBasic.Mid(receivedData, pos2 + 1)

pos1 = receivedData.IndexOf("<") + 1

pos2 = receivedData.IndexOf(">") + 1

End If

If (pos1 = 0 Or pos2 = 0) Then

' we do not have both start and end markers and we are done

done = True

Else

' we have both start and end markers

length = pos2 - pos1 + 1

If (length > 0) Then

'remove the start and end markers from the command

newCommand = Mid(receivedData, pos1 + 1, length - 2)

' show the command in the text box

RichTextBox1.AppendText("Command = " & newCommand & vbCrLf)

'remove the command from receivedData

receivedData = Mid(receivedData, pos2 + 1)

'Evaluate and post the inputs to the Arduino

```

If (newCommand.Substring(0, 3) = "D10") Then
  If (newCommand.Substring(3, 1) = "0") Then
    D10_Value_lbl.Text = "LOW"
    Digital_in_D10 = "LOW"
  ElseIf (newCommand.Substring(3, 1) = "1") Then
    D10_Value_lbl.Text = "HIGH"
    Digital_in_D10 = "HIGH"
  End If
End If

```

```

If (newCommand.Substring(0, 3) = "D11") Then
  If (newCommand.Substring(3, 1) = "0") Then
    D11_Value_lbl.Text = "LOW"
    Digital_in_D11 = "LOW"
  ElseIf (newCommand.Substring(3, 1) = "1") Then
    D11_Value_lbl.Text = "HIGH"
    Digital_in_D11 = "HIGH"
  End If
End If

```

```

If (newCommand.Substring(0, 3) = "D12") Then
  If (newCommand.Substring(3, 1) = "0") Then
    D12_Value_lbl.Text = "LOW"
    Digital_in_D12 = "LOW"
  ElseIf (newCommand.Substring(3, 1) = "1") Then
    D12_Value_lbl.Text = "HIGH"
    Digital_in_D12 = "HIGH"
  End If
End If

```

```

If (newCommand.Substring(0, 3) = "D13") Then
  If (newCommand.Substring(3, 1) = "0") Then
    D13_Value_lbl.Text = "LOW"
    Digital_in_D13 = "LOW"
  ElseIf (newCommand.Substring(3, 1) = "1") Then
    D13_Value_lbl.Text = "HIGH"
    Digital_in_D13 = "HIGH"
  End If
End If

```

```

If (newCommand.Substring(0, 2) = "A0") Then
  A0_Value_lbl.Text = newCommand.Substring(2, 4)
  Analog_in_A0 = Format(Val(A0_Value_lbl.Text) * 1 * (5 / 1024),
"#####.#0")
  A0_Value_lbl.Text = Format(Val(A0_Value_lbl.Text) * 1 * (5 / 1024),
"#####.#0")

```

```

End If

If (newCommand.Substring(0, 2) = "A1") Then
    A1_Value_lbl.Text = newCommand.Substring(2, 4)
    Analog_in_A1 = Format(Val(A1_Value_lbl.Text) * 1 * (5 / 1024),
"#####.#0")
    A1_Value_lbl.Text = Analog_in_A1
End If

If (newCommand.Substring(0, 2) = "A2") Then
    A2_Value_lbl.Text = newCommand.Substring(2, 4)
    Analog_in_A2 = Format(Val(A2_Value_lbl.Text) * 1 * (5 / 1024),
"#####.#0")
    A2_Value_lbl.Text = Analog_in_A2
End If

If (newCommand.Substring(0, 2) = "A3") Then
    A3_Value_lbl.Text = newCommand.Substring(2, 4)
    Analog_in_A3 = Format(Val(A3_Value_lbl.Text) * 1 * (5 / 1024),
"#####.#0")
    A3_Value_lbl.Text = Analog_in_A3
End If

End If ' (length > 0)

End If '(pos1 = 0 Or pos2 = 0)

End While

End Function

Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click

    log = False
    Dim saveFileDialog1 As New SaveFileDialog() 'Create the dialog object
    saveFileDialog1.Filter = "Text File|*.txt|CSV file|*.csv" 'Choose a file extension
    saveFileDialog1.Title = "Name your results file" 'Choose the title of the dialog box
    saveFileDialog1.ShowDialog() 'Open the file dialog

    filepathway = saveFileDialog1.FileName 'Store the full file pathway with file name

    Label18.Text = filepathway 'Put the chosen file name on the form so I can see it

    My.Computer.FileSystem.WriteAllText(filepathway, filepathway + "," +
DateTime.Now + vbCrLf + "Seconds Elapsed (sec)" + "," + "Dummy Pot (A1)" + "," +

```

```
"Arm Pot (A2)" + "," + "Motor Pot (A2)" + "," + "Dummy Vertical Pot (A3)" + vbCrLf, True)
```

'The "TRUE" indicates that the computer should append the data if the file already exists

```
End Sub
```

```
Private Sub TextBox1_TextChanged(sender As Object, e As EventArgs) Handles TextBox1.TextChanged
```

```
    interval = TextBox1.Text  
    Sampling_timer.Interval = interval
```

```
End Sub
```

```
Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
```

```
    If Button2.Text = "Start Logging" Then
```

```
        log = True  
        Button2.Text = "Stop Logging"  
        start_time = (DateTime.Now - New DateTime(1970, 1, 1, 0, 0, 0, 0)).TotalSeconds
```

```
    ElseIf Button2.Text = "Stop Logging" Then
```

```
        log = False  
        Button2.Text = "Start Logging"
```

```
    End If
```

```
End Sub
```

```
End Class
```


APPENDIX C

```
#include <Servo.h>
Servo myservo;
Servo myservo2;
```

```
/* *****
```

```
Adapted from arduinoVBserialcontrol by Stephen Abbadessa
Previous foundation code by www.martyncurrey.com
Crosby_Machine_Controller_v4
```

Pin Assignments -

Pins

A0 -

A1 -

A2 -

A3 -

D13 - Digital Input

D12 - Digital Input

D11 - Digital Input

D10 - Digital Input

D9 - Digital Output

D8 - Reserved for Adafruit Ultimate GPS Logger Shield

D7 - Reserved for Adafruit Ultimate GPS Logger Shield

D6 - Servo Output

D5 - PWM Analog Output

D4 - Digital Output

D3 - PWM Analog Output

D2 - Digital Output

D1 - Not used. Reserved for communications

D0 - Not used. Reserved for communications

D21 - Servo2 output

USB Serial Commands

The expected commands are:

Pin HIGH / LOW

<P001ON> - P001 = Pin 1. ON = HIGH

<P001OF> - P001 = Pin 1. OF = LOW

<V003XXX> - Digital pin3 set PWM to XXX

Other.

```

<HELLO>
<FON>
<FOF>
<DON>
<DOF>

*/ //////////////////////////////////////

boolean debug = true;

// length of command is 20 chrs
// if you need longer commands then increase the size of numChars
const byte numChars = 20;
char receivedChars[numChars];

boolean newData = false;

int sensorValue;

boolean feedback = true;

void setup()
{
  pinMode(13, INPUT);
  pinMode(12, INPUT);
  pinMode(11, INPUT);
  pinMode(10, INPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT); //GPS Shield Xmits on this pin
  pinMode(7, OUTPUT); //GPS Shield Receives on this pin
  pinMode(6, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(2, OUTPUT);

  myservo.attach(6);
  myservo2.attach(21);

  Serial.begin(9600);
  sendStartMessage();

  //analogReference(INTERNAL1V1);
  analogReference(DEFAULT);
  sensorValue = analogRead(A0);
  delay(100);
}

```

```

    sensorValue = analogRead(A0);
    delay(100);
    sensorValue = analogRead(A0);
    delay(100);

}

void loop()
{
    if (Serial.available() > 0) { rcvWithStartEndMarkers(); }
    if (newData) { parseData(); }
}

/*****
* sends a start up message over serial.
* Assumes serial is connected
*
* Global:
*   debug
* Local:
*
*/
void sendStartMessage()
{

    Serial.println(" ");
    Serial.println("arduinoVBserialControl Ver 1.0");
    Serial.println(" ");
    Serial.println("DON = debug on");
    Serial.println("DOF = debug off");
    Serial.println("START to reset");
    Serial.println(" ");

    if (debug) { Serial.println("Debug is on"); }
    else { Serial.println("Debug is off"); }
    Serial.println(" ");
}

/*****
* Checks receivedChars[] for commands
*
* Global:
*   receivedChars[]
*   newData;
*
* Local:

```

```

*
*/
void parseData()
{
    newData = false;
    if (debug) { Serial.println( receivedChars ); }

    // HELLO
    // If the Arduino receives "HELLO" it sends "HELLO" back
    // This is used by the VB program to show it is connected
    if (strcmp(receivedChars, "HELLO") == 0)
    {
        Serial.println("HELLO");

        //Blink the LED on pin 13 to show "HELLO" was received
        digitalWrite(13,HIGH);
        delay(100);
        digitalWrite(13,LOW);
    }

    if (strcmp(receivedChars, "START") == 0)
    {
        sendStartMessage();
    }

    // PIN
    // P001ON - P for pin. 001 is the button number. ON = on
    // P001OF - P for pin. 001 is the button number. OF = off
    if (receivedChars[0] == 'P' )
    {
        int tmp = convertToNumber( 1 ); //Returns the integer value of three digits after
poistion 1
        if ( receivedChars[4] == 'O' && receivedChars[5] == 'N' ) {
digitalWrite(tmp,HIGH); }
        if ( receivedChars[4] == 'O' && receivedChars[5] == 'F' ) {
digitalWrite(tmp,LOW); }
    } // PIN

    //PWM
    // V001220 - V for "variable Voltage = PWM". 001 is the pin number. 220 is the
duty cycle
    if (receivedChars[0] == 'V' )
    {
        int loc = convertToNumber( 1 );
        int duty = convertToNumber( 4 );
    }
}

```

```

        analogWrite(loc, duty);
    } // PWM

//Servo

// S001220 - 001 is the pin number. 220 is the duty cycle
if (receivedChars[0] == 'S' )
{
    int loc = convertToNumber( 1 );
    int duty = convertToNumber( 4 );
    if (loc==6)
    {
        myservo.write(duty);
    }
    if (loc==21)
    {
        myservo2.write(duty);
    }
}

// Send Analog Port Values when asked

if ( receivedChars[0] == 'S' && receivedChars[1] == 'A')
{
    Serial.print("<A0");
    sensorValue=analogRead(A0);
    if ((sensorValue<1000) && (sensorValue>=100))
    {
        Serial.print("0");
    }
    if ((sensorValue<100) && (sensorValue>=10))
    {
        Serial.print("00");
    }
    if ((sensorValue<10) && (sensorValue>0))
    {
        Serial.print("000");
    }
    if ((sensorValue==0))
    {
        Serial.print("0000");
    }
    Serial.print(sensorValue);
    Serial.println(">");

    Serial.print("<A1");

```

```

sensorValue=analogRead(A1);
if ((sensorValue<1000) && (sensorValue>=100))
{
  Serial.print("0");
}
if ((sensorValue<100) && (sensorValue>=10))
{
  Serial.print("00");
}
if ((sensorValue<10) && (sensorValue>0))
{
  Serial.print("000");
}
if ((sensorValue==0))
{
  Serial.print("0000");
}
Serial.print(sensorValue);
Serial.println(">");

```

```

Serial.print("<A2");
sensorValue=analogRead(A2);
if ((sensorValue<1000) && (sensorValue>=100))
{
  Serial.print("0");
}
if ((sensorValue<100) && (sensorValue>=10))
{
  Serial.print("00");
}
if ((sensorValue<10) && (sensorValue>0))
{
  Serial.print("000");
}
if ((sensorValue==0))
{
  Serial.print("0000");
}
Serial.print(sensorValue);
Serial.println(">");

```

```

Serial.print("<A3");
sensorValue=analogRead(A3);
if ((sensorValue<1000) && (sensorValue>=100))
{
  Serial.print("0");
}

```

```

}
if ((sensorValue<100) && (sensorValue>=10))
{
  Serial.print("00");
}
if ((sensorValue<10) && (sensorValue>0))
{
  Serial.print("000");
}
if ((sensorValue==0))
{
  Serial.print("0000");
}
Serial.print(sensorValue);
Serial.println(">");
}

```

// Send the digital Port Values when asked

```

if ( receivedChars[0] == 'S' && receivedChars[1] == 'D')
{
  Serial.print("<D10");
  int digitalstate = digitalRead(10);
  if (digitalstate==1)
  {
    Serial.print("1");
  }
  else
  {
    Serial.print("0");
  }
  Serial.println(">");

  Serial.print("<D11");
  digitalstate = digitalRead(11);
  if (digitalstate==1)
  {
    Serial.print("1");
  }
  else
  {
    Serial.print("0");
  }
  Serial.println(">");
}

```

```

Serial.print("<D12");
digitalstate = digitalRead(12);
if (digitalstate==1)
{
  Serial.print("1");
}
else
{
  Serial.print("0");
}
Serial.println(">");

Serial.print("<D13");
digitalstate = digitalRead(13);
if (digitalstate==1)
{
  Serial.print("1");
}
else
{
  Serial.print("0");
}
Serial.println(">");

}

// ACKNOWLEDGMENT
// FON
// FOF
if ( receivedChars[0] == 'F' )
{
  if ( receivedChars[1] =='O' && receivedChars[2] =='N' ) { feedback = true; }
  if ( receivedChars[1] =='O' && receivedChars[2] =='F' ) { feedback = false; }

  if (feedback) { Serial.println("acknowledgment is on"); }
  else { Serial.println("acknowledgment is off"); }
} // FEEDBACK

if ( receivedChars[0] == 'D' )
{
  if ( receivedChars[1] =='O' && receivedChars[2] =='N' ) { debug = true;
Serial.println("Debug is on"); }
  if ( receivedChars[1] =='O' && receivedChars[2] =='F' ) { debug = false;
Serial.println("Debug is off"); }
}

```



```

    }

}

/*****
* Takes serial input and looks for data between a start and end marker.
*
* Global:
* Updates receivedChars[] with the received data
*
* Local:
*
*/
void recvWithStartEndMarkers()
{

// function recvWithStartEndMarkers by Robin2 of the Arduino forums
// See http://forum.arduino.cc/index.php?topic=288234.0

    static boolean recvInProgress = false;
    static byte ndx = 0;
    char startMarker = '<';
    char endMarker = '>';

    char rc;

    if (Serial.available() > 0)
    {
        rc = Serial.read();

        if (recvInProgress == true)
        {
            if (rc != endMarker)
            {
                receivedChars[ndx] = rc;
                ndx++;
                if (ndx >= numChars) { ndx = numChars - 1; }
            }
            else
            {
                receivedChars[ndx] = '\0'; // terminate the string
                recvInProgress = false;
                ndx = 0;
                newData = true;
            }
        }
    }
}

```

```

    }
  }

  else if (rc == startMarker) { rcvInProgress = true; }
}

}

/*****
* converts 3 ascii characters to a numeric value
*
* Global:
* Expects receivedChars[] to contain the ascii characters
*
* Local:
* startPos is the position of the first character
*
*
*/

int convertToNumber( byte startPos)
{
  unsigned int tmp = 0;
  tmp = (receivedChars[startPos]-48) * 100;
  tmp = tmp + (receivedChars[startPos+1]-48) * 10;
  tmp = tmp + receivedChars[startPos+2]-48;
  return tmp;
}

void sendOK(int val)
{
  // The 3 command buttons wait for the OK signal
  Serial.print("OK");Serial.println(val);
}

```

AUTHOR'S BIOGRAPHY

Michael B. Jones was born in Portland, Maine on July 27, 1997. He was raised in Gray, Maine and graduated from Cheverus High School in 2015. Michael majors in Mechanical Engineering and is a member of the 3D printing club.

After graduation, Michael will start work for Pratt and Whitney in North Berwick Maine as a structural engineer.