

Design & Evaluation of an Underactuated Lower Body Exoskeleton

Zackory James Biggers

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State
University in partial fulfillment of the requirements for the degree of

Master of Science

in

Mechanical Engineering

Alan T. Asbeck

Maury A. Nussbaum

Kaveh Akbari Hamed

May 6th, 2022

Blacksburg, Virginia

Keywords: exoskeleton, lower body, underactuated, pantograph, gait cycle

Copyright 2022, Zackory Biggers

Design & Evaluation of an Underactuated Lower Body Exoskeleton

Zackory James Biggers

(ABSTRACT)

An underactuated exoskeleton design for walking assistance is presented and evaluated. The exoskeleton uses one motor per leg and makes use of a pantograph to reduce the overall profile and allow the exoskeleton to closely follow the shape of the user's leg. Support is provided between the ball of the user's foot and their waist by compressing a spring in parallel with the user's leg during Stance Phase. The exoskeleton has a mass of 14.0 kg (30.8 lbs) and was tested up to a supplied spring force of 323.6 N (72.75 lbf) which equates to around 161.8 N (36.38 lbf) of assistive force at the waist. Range of motion tests showed minimal restriction at the knee and ankle, but some restriction of the hip. Human subject experiments using a simple gait detection method based on GRF at walking speeds from 0.45 m/s to 1.12 m/s (1.0 mph to 2.5 mph) were performed and showed an increase in the time between actual heel strike and predicted heel strike of approximately 0.05 seconds to 0.1 seconds. Lastly, calculations are presented examining the effect of exoskeleton assistance on the biological joint moments and optimizing the actuator design to reduce power consumption. The actual performance of the exoskeleton is compared with the calculations based on the joint angles during a typical walking cycle.

Design & Evaluation of an Underactuated Lower Body Exoskeleton

Zackory James Biggers

(GENERAL AUDIENCE ABSTRACT)

A design for an exoskeleton capable of providing walking assistance without requiring a motor for every joint is presented and evaluated. The exoskeleton uses one motor per leg and makes use of a pantograph to reduce the required size and allow the exoskeleton to closely follow the shape of the user's leg. Support is provided between the ball of the user's foot and their waist by compressing a spring attached beside the user's leg while the user's foot is on the ground. The exoskeleton weighs 14.0 kg (30.8 lbs) and was tested up to a supplied spring force of 323.6 N (72.75 lbf) which equates to around 161.8 N (36.38 lbf) of assistive force at the waist. Range of motion tests showed minimal restriction at the knee and ankle, but some at the hip. Testing with a human participant using a simple method for determining when to apply support and remove it based on the forces measured at the user's foot were performed at walking speeds of 0.45 m/s to 1.12 m/s (1.0 mph to 2.5 mph). These tests showed an increase in the time between when the heel of the foot initially hits the ground and when the exoskeleton code determined that it occurred of approximately 0.05 seconds to 0.1 seconds. Lastly, calculations are presented examining how exoskeleton assistance affects what is felt at the joints of the user and determining what spring stiffness would best reduce the power required from the motors. The actual performance of the exoskeleton is compared with the calculations based on the joint angles during normal human walking.

Acknowledgements

First I would like to thank my committee chairman, Dr. Alan Asbeck. When I reached out about available positions in the Assistive Robotics Lab (ARL) during my Sophomore year, Dr. Asbeck was quick to share with me what the lab was doing and to gauge both how I could help the lab and how the lab could help me. Working in the ARL over the past four and a half years has enabled me to grow as an engineer and a professional and I will always be grateful for the opportunities and experiences that it has allowed me to gain.

Thank you to my project partner, Evan Claessen, who has experienced all of the highs and lows of this project alongside me. His willingness to work hard and his wealth of experience from robotics to machining has been invaluable to the successful completion of this project.

I would also like to thank Dr. Maury Nussbaum and Dr. Kaveh Hamed for serving on my advisory committee.

To all current and past members of the ARL that have helped me learn new skills and knowledge, my Senior Design team that helped develop the foundation of this project, and our Undergraduate Research Assistants Madeline, Forrest, and Christian who have helped wherever necessary, thank you for all of your contributions to this project and to the engineer I have grown to be.

Finally, I would be remiss not to give thanks to God, my family, and my friends, without whom I never would have made it to where I am today. Your love and support is more appreciated than you could know.

Table of Contents

List of Figures	vii
Chapter 1: Introduction	1
1.1 Exoskeleton Architectures	1
1.2 Gait Cycle Detection/Prediction	5
1.3 A Novel Underactuated Exoskeleton	6
Chapter 2: Mechanical Design	9
2.1 Frame	10
2.2 Foot Attachment	13
2.3 Soft Goods	15
2.4 Actuator	16
Chapter 3: Electrical/Sensor Design	17
3.1 Sensors/Components	17
3.2 Controllers	20
3.2.1 ESP32	20
3.2.2 ODrive	21
Chapter 4: Software	23
4.1 ESP32 Code Structure	23
4.2 Gait Detection: Force Method	27
4.3 Gait Detection: Distance Method	29

Chapter 5: Calculations	31
5.1 Joint Moment Calculations	31
5.2 Actuator Length Calculations	40
5.3 Required Actuator/Motor Performance	42
5.4 Spring Stiffness Optimization	48
Chapter 6: Testing/Results	52
6.1 Range of Motion	52
6.2 Treadmill Walking Tests	56
6.3 Model Comparison	67
Chapter 7: Conclusion	71
7.1 Future Work	71
Appendix A : ODrive Initial Setup Values	73
Appendix B : Full Electrical Schematic*	76
Appendix C : Moment Derivation	78
Appendix D : Motor Calculation Code Flowchart	80
References	81

List of Figures

Figure 1.1: BATEX exoskeleton labeled diagrams	4
Figure 1.2: PALExo labeled diagram and gait model.....	4
Figure 1.3: Once cycle of the human walking gait [19]	5
Figure 1.4: Support force path	7
Figure 2.1: Orthographic view of CAD model and real-world underactuated exoskeleton	9
Figure 2.2: Exoskeleton frame CAD	10
Figure 2.3: Exoskeleton knee joint CAD.....	11
Figure 2.4: Pantograph example diagram	12
Figure 2.5: Labeled Overshoe.....	13
Figure 2.6: Exoskeleton soft goods (human interface)	15
Figure 2.7: Compressed and Uncompressed Actuator Comparison	16
Figure 3.1: Dead Man’s Switch	18
Figure 3.2: Exoskeleton electronics back plate.....	20
Figure 4.1: ESP32 code flowchart. Modified from [43].....	25
Figure 4.2: Exoskeleton actuator safety cover	26
Figure 4.3: Gait determination algorithm pseudocode: Force method. <i>heel_force</i> and <i>ball_force</i> are the force at the heel and ball of the foot in lbs that was read earlier and brought into this algorithm, <i>prev_heel_force</i> and <i>prev_ball_force</i> are the force values from the last reading, “heel threshold” is the value that indicates a heel strike, <i>phase</i> is the current gait cycle phase of the leg, “SWING” means the leg is in Swing Phase, “STANCE” means the leg is in Stance Phase, <i>compression</i> is what value to compress the spring to in mm, “-15 mm offset” is the 15 mm distance away from the spring to prevent accidental force during Swing, “support force compression” is the required spring compression to achieve the desired support force, <i>stance_state</i> is the condition heel during stance (either staying on the ground or lifting off of the ground), <i>heel_force_slope</i> and <i>ball_force_slope</i> is the current force value minus the previous force value at the heel and ball of the foot, <i>stance_counter</i> is the number of times in a row a heel lift was predicted, and “lift prediction threshold” is the number of heel lifts in a row that must be predicted for the leg to enter Swing.	28

Figure 4.4: Gait determination algorithm pseudocode: Distance method. <i>heel_distance</i> is the distance from the ground at the heel in mm that was read earlier and brought into this algorithm, “start stance threshold” is the distance value that indicates that heel strike is about to occur, “end stance threshold” is the distance value that indicates that the heel is lifting from Stance, <i>phase</i> is the current gait cycle phase of the leg, “SWING” means the leg is in Swing Phase, “STANCE” means the leg is in Stance Phase, <i>compression</i> is what value to compress the spring to in mm, “-15 mm offset” is the 15 mm distance away from the spring to prevent accidental force during Swing, “support force compression” is the required spring compression to achieve the desired support force.	30
Figure 5.1: Leg free body diagram	32
Figure 5.2: Support forces used for joint moment calculations	34
Figure 5.3: Stick figure code verification model. In the diagram, the letters H, K, and A denote the hip, knee, and ankle joints, respectively.	35
Figure 5.4: Stick figure movement through Stance Phase	35
Figure 5.5: Distances from each joint to the line of force between the exoskeleton hip attachment and the connection at the foot.....	36
Figure 5.6: Joint moments caused by a constant support force and biological joint moments scaled so peak ankle moments match	37
Figure 5.7: Joint moments caused by a force proportional to the GRF and biological joint moments scaled so peak ankle moments match	38
Figure 5.8: Joint moments assisted by a force proportional to the GRF and unassisted biological joint moments	39
Figure 5.9: Labeled exoskeleton frame for actuator length calculations	40
Figure 5.10: Biological joint angles and calculated actuator and GRF angles	42
Figure 5.11: Free body diagram for support ratio derivation (A is the location of the ankle joint) ...	43
Figure 5.12: Support ratio between the force supplied by the actuator and the supportive force felt at the waist	44
Figure 5.13: Required actuator and motor properties to maintain 222.4 N (50 lbf) at waist.....	47
Figure 5.14: Required motor power with different spring stiffness values	50

Figure 5.15: Peak absolute value of the required motor power resulting from different spring constants. Potential springs of the correct dimensions that are available from McMaster Carr are shown with Xs	51
Figure 6.1: Range of Motion measurements, method one	53
Figure 6.2: Range of Motion measurements, method two (Knee Flexion)	54
Figure 6.3: Range of Motion measurements, method two (Ankle Dorsiflexion)	54
Figure 6.4: Range of Motion measurements, method two (Ankle Inversion)	55
Figure 6.5: Human subject walking trials on treadmill.....	58
Figure 6.6: 1.12 m/s (2.5 mph) experimental force data with estimated and predicted heel strike	59
Figure 6.7: 1.12 m/s (2.5 mph) experimental joint angle data	60
Figure 6.8: 1.12 m/s (2.5 mph) experimental actuator data and force data for right leg	61
Figure 6.9: 1.12 m/s (2.5 mph) experimental actuator data and force data for left leg	62
Figure 6.10: 1.12 m/s (2.5 mph) experimental motor position data for both legs	63
Figure 6.11: 0.89 m/s (2.0 mph) experimental distance data for right leg.....	64
Figure 6.12: 0.89 m/s (2.0 mph) experimental distance data for left leg.....	64
Figure 6.13: 0.45 m/s (1.0 mph) experimental force data for right leg comparison when distance sensors are and are not being read	65
Figure 6.14: Sample period and detection delay time comparison for multiple walking speeds	66
Figure 6.15: Biological joint angles and calculated actuator and GRF angles compared to joint angles collected from exoskeleton and resulting actuator and GRF angles. The shaded fields around the knee and ankle exoskeleton angles show the variance of the data over 7 gait cycles.	67
Figure 6.16: Actuator and leg length based on theoretical joint angles and joint angles collected from exoskeleton	68
Figure 6.17: Required support ratio to maintain 222.4 N (50 lbf) at waist based on theoretical joint values and joint values collected from exoskeleton.....	69
Figure 6.18: Exoskeleton force comparison showing spring compression, spring force from the actuator, and force felt by the user from the exoskeleton for the theoretical model, commanded exoskeleton values, and actual exoskeleton values.....	70

Chapter 1: Introduction

Modern media habitually portrays exoskeletons as miraculous creations that can provide superpowers to their wearers. While those portrayals are currently rooted in science-fiction, the real-world development and application of exoskeletons has potential for large impact in the industrial and medical fields. An exoskeleton is an assistive device worn by a user that aims to restore or augment the natural abilities of the user. Today, exoskeletons can be split into two groups: passive and active. A passive exoskeleton does not rely on powered components to operate, instead using potential energy stored in components such as springs or elastics and redirecting it to provide energy to assist in motion. Active exoskeletons use powered components such as motors or pneumatic pumps to apply assistive forces to desired locations of the wearer. Active exoskeletons are capable of providing greater assistance to a user, but typically at the cost of being both heavier and taking up more space.

1.1 Exoskeleton Architectures

In terms of active lower-body exoskeletons, these devices can further be distinguished by the number of joints where actuation (assistive power) is provided. A fully actuated lower-body exoskeleton provides actuation at the hip, knee, and ankle joints. Some of the existing exoskeletons meeting these criteria are the EXO-H3 [1], BLEEX [2], and REX [3]. Fully actuated exoskeletons provide the greatest amount of assistance since they are capable of applying desired forces at each

joint individually. This makes them ideal for users that have little to no ability to control their own joints, such as stroke victims that experience varying degrees of paralysis.

Some exoskeletons only provide force directly to one or two targeted joints (typically the hip and knee joints). Exoskeletons that provide actuation to both the hip and knee (via motor) include the HAL-5 [4], LLEX [5], ReWalk [6], TWIICE ONE [7], and Indego [8]; a sixth exoskeleton made by Aguilar-Sierra et al. [9] uses a combination of motors and pneumatics to actuate the hip and knee. Since the ankle remains free of actuation, the user is less restricted by these exoskeletons, but also receives less overall assistance. Users unable to use their ankle would receive less benefits from these exoskeletons, while more able users would benefit from the greater range of motion.

Exoskeletons that provide actuation to a single joint exist for each different joint and can include both passive and active devices. The Active Pelvis Orthosis (APO) created by Giovacchini et al. [10] is an active hip exoskeleton that uses series elastic actuators at each hip joint to support flexion and extension of the hip. A quasi-passive knee exoskeleton developed by Dollar et al. [11] uses a motor to actively move a spring into parallel with the knee joint from heel-strike to toe-off of the gait cycle (discussed in Chapter 1.2) and remove it during the Swing Phase. This allows the spring to store the energy during the heel-strike and use it to assist the wearer as they lift their heel off the ground. B-Temia's Keeogo [12] and the Lockheed Martin ONYX [13] (derived from B-Temia's work) are active knee exoskeletons that utilize motors at the knee joint. The former is geared towards a clinical setting while the latter is aimed towards more industrial applications (first responders). The RoboKnee [14] is another active knee exoskeleton, but it differs by using series elastic actuators to apply torque across the knee joint. Wiggin et al. [15] created a purely passive ankle exoskeleton that used a novel 'smart clutch' to engage and disengage a spring in parallel

with the ankle, allowing free ankle rotation during Swing Phase and providing mechanical assistance during stance phase. Lastly, an active ankle-foot orthoses (AAFO), developed by Blaya & Herr [16], implemented a torsional spring to provide variable resistance to ankle joint throughout the gait cycle in order to assist wearers that have difficulty lifting the front of their foot (drop-foot gait). The traditional benefits of these single joint exoskeletons is that they are often less complicated, lighter weight, and have a lower profile. The downside is that they do not provide assistance to the other joints, requiring more energy/stability from the wearer than exoskeletons actuating more joints.

The last type of lower body exoskeleton that will be discussed is the underactuated exoskeleton. These exoskeletons are a middle ground between fully actuated exoskeletons and those actuating individual joints. An underactuated exoskeleton is designed to provide assistance to multiple joints while using fewer actuators than the number of joints being assisted. This allows the exoskeleton to provide greater assistance than a single joint exoskeleton, while benefiting from the increased simplicity and decreased weight of having less actuators. Two examples of Underactuated Exoskeletons would be the BATEX [17] and the PALExo [18]. The BATEX, seen in Figure 1.1, uses compliant series elastic actuators on each thigh to simulate the function of a muscle and transfer energy between the hip and knee joints during walking. The degree of actuation (or “muscle stiffness”) is determined by the measured ground reaction force (GRF).

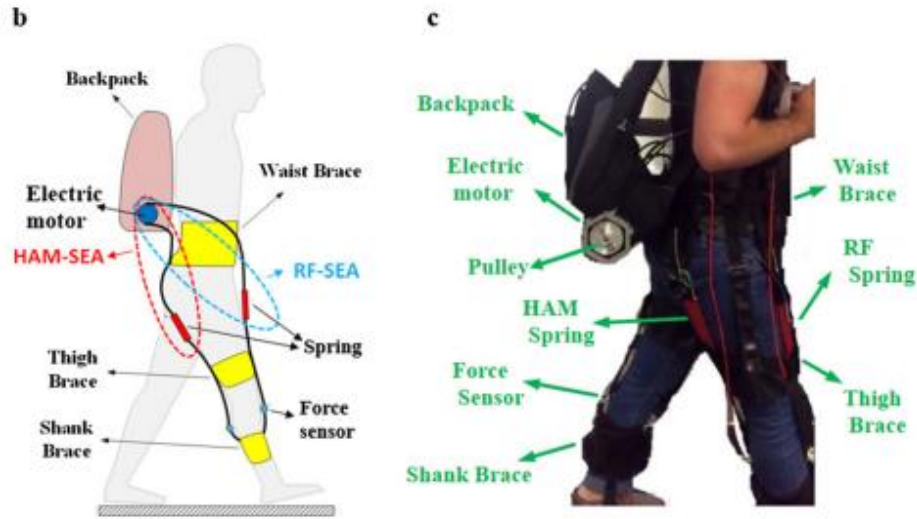


Figure 1.1: BATEX exoskeleton labeled diagrams

The PALExo, seen in Figure 1.2, is designed for the purpose of load-bearing. It uses two actuators per leg to drive flexion and extension for the hip and knee. Due to the parallel structure of the exoskeleton to the leg and passive degrees of freedom (DOF), the wearer is able to maintain their typical range of motion, allowing for increased comfort. Because the design of the PALExo does not follow the shape of the leg like most lower body exoskeletons, some actions, such as sitting in a chair, would not be possible.



Figure 1.2: PALExo labeled diagram and gait model

1.2 Gait Cycle Detection/Prediction

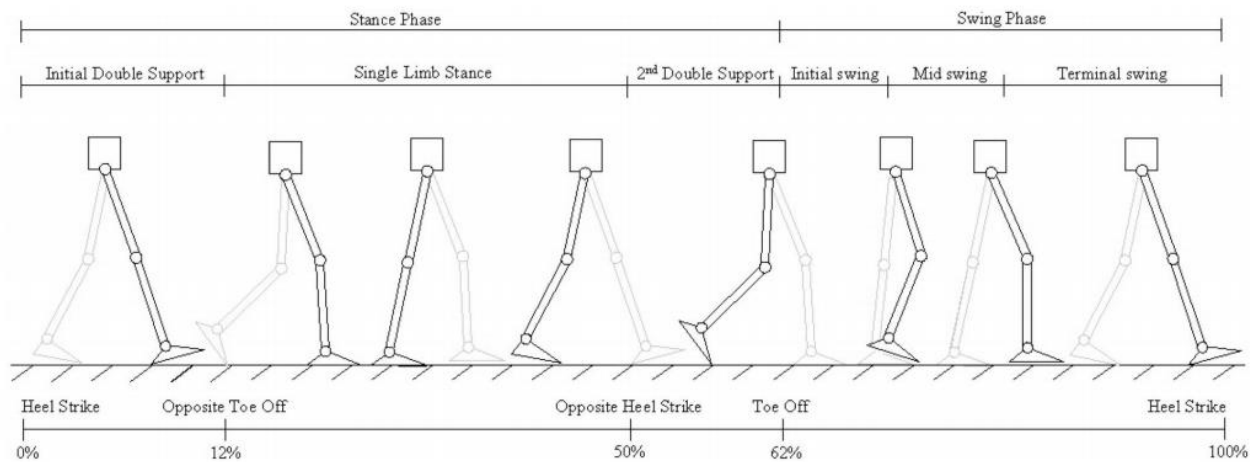


Figure 1.3: Once cycle of the human walking gait [19]

Next the motion of the legs during human walking is discussed since lower-body exoskeletons need to accommodate their motions and torques. The gait cycle is the cyclical pattern of motion performed by humans during walking, seen above in Figure 1.3. It consists of two phases, Stance and Swing. Stance Phase begins with heel strike and is the portion of walking where the foot is on the ground, ending at toe off. Swing Phase begins at toe off and is the portion of walking where your foot is off of the ground, ending at the next heel strike.

When it comes to the control of active exoskeletons, or similarly active prosthesis or walking robots, there are many approaches used to enable them to move appropriately. Some of the greatest challenges are enabling these devices to move according to the user's intention and successfully and naturally navigating over different terrains such as slopes and stairs. A common method for allowing exoskeletons and prosthesis to react to user intent is the use of myoelectric sensors which can measure the extent of muscle activation at desired muscles in the user. Ha, Varol, and Goldfarb [20] use this method to provide control to a knee prosthesis that simulates natural movement by

controlling the stiffness and damping of the knee joint when not bearing the user's weight. Chen and Wang [21] took this a step further and used two IMUs to determine the slope of the ground and used this together with myoelectric sensors to allow adaptive slope walking with a lower-limb prosthesis. A method of gait detection was achieved by Li et al. [22] on a lower extremity exoskeleton. The exoskeleton employed three analog infrared (IR) distance sensors with a range of 40 mm to 300mm. The sensors were mounted to the side of the user's foot, two at the heel on either side of the foot, and one at the ball of the foot. Using the distance data from the three sensors, information on the orientation of the foot was derived and an online detection algorithm was used to detect the significant events of the gait cycle. The methods discussed so far all use sensors for detection and use the data to allow the device to then react. While this is a viable method, increased performance could be obtained using sensors that could collect data that allows for prediction of the gait cycle. The control method for the HAL-3 exoskeleton by Kawamoto [23], uses myoelectric sensors together with GRF data to predict user intent. This control method operates by separating motion into phases and matching the joint angles proportionally to the muscle activation. From here the needed phase shift required to match user intent is found using the GRF data.

1.3 A Novel Underactuated Exoskeleton

Underactuated exoskeletons have great potential for allowing for light and simplified designs while still providing support to a large portion of the body, but they have not been widely explored yet. Typically, a larger number of actuators in an exoskeleton corresponds to an increased mass and complexity. As such, it is interesting to consider exoskeleton designs that can potentially provide a large amount of assistance to the body while minimizing the number of actuators and

hence mass and complexity. Underactuated exoskeletons can potentially provide beneficial torques to multiple joints simultaneously, maximizing the benefit to the body as compared to the exoskeleton complexity. This thesis presents the design and evaluation of an underactuated lower body exoskeleton for walking assistance that seeks to further explore the ability of underactuated exoskeleton designs to remain light weight, low profile, and employ a simple method of control. Each leg of the device is designed to provide 222.4 N (50 lbf) of assistive force directed between the ball of the user's foot and their waist as seen below in Figure 1.4.

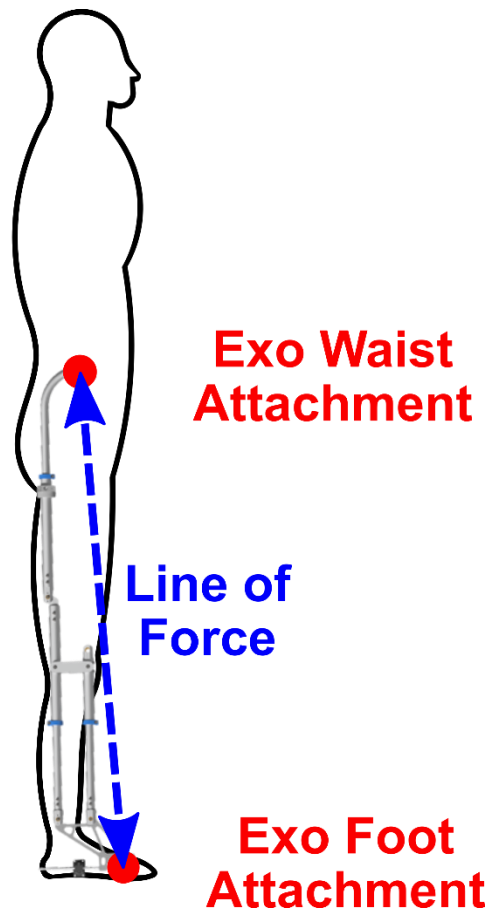


Figure 1.4: Support force path

For each leg, assistance is provided when the foot is placed on the ground by using a motor to compress a spring that is connected in parallel to the leg. When the foot is lifted from the ground, the motor decompresses the spring so no force is applied. The exoskeleton is intended for users that have full ability to walk without the exoskeleton and will not provide walking or balance assistance to those that could not walk without it. With the current version of this exoskeleton, individuals that are capable of walking but have experienced a light injury, such as a sprain, or reduced leg strength due to aging could potentially make use of this device to reduce the strain on their body.

The presented exoskeleton is a novel design that contributes to the advancement of exoskeleton research. The exoskeleton uses only one motor per leg, constituting an underactuated design, and includes a pantograph linkage to closely follow the shape of the leg and maintain a smaller profile. Additionally, calculations were performed to examine the effect of exoskeleton assistance on the biological joint moments and to explore the optimization of the actuator design to reduce power consumption. Lastly, human subject experiments were conducted with the exoskeleton. These included testing the range of motion permitted by the exoskeleton, and testing the exoskeleton's operation with simple gait detection methods based on GRF or the distance between the foot and the ground. The actual performance of the exoskeleton was compared with predictions based on the joint angles during a typical walking cycle.

Chapter 2: Mechanical Design

The mechanical design of the underactuated exoskeleton presented in this thesis focused on allowing full range of motion and following the shape of the body in order to maximize comfort during walking and reduce protrusion of the exoskeleton. Collectively these two factors would allow the exoskeleton to be accepted more readily by a user. The exoskeleton can be seen below in Figure 2.1. The exoskeleton had a mass of 14.0 kg (30.8 lbs) and was tested up to a supplied spring force of 323.6 N (72.75 lbf) which equates to around 161.8 N (36.38 lbf) of force at the waist.



Figure 2.1: Orthographic view of CAD model and real-world underactuated exoskeleton

2.1 Frame

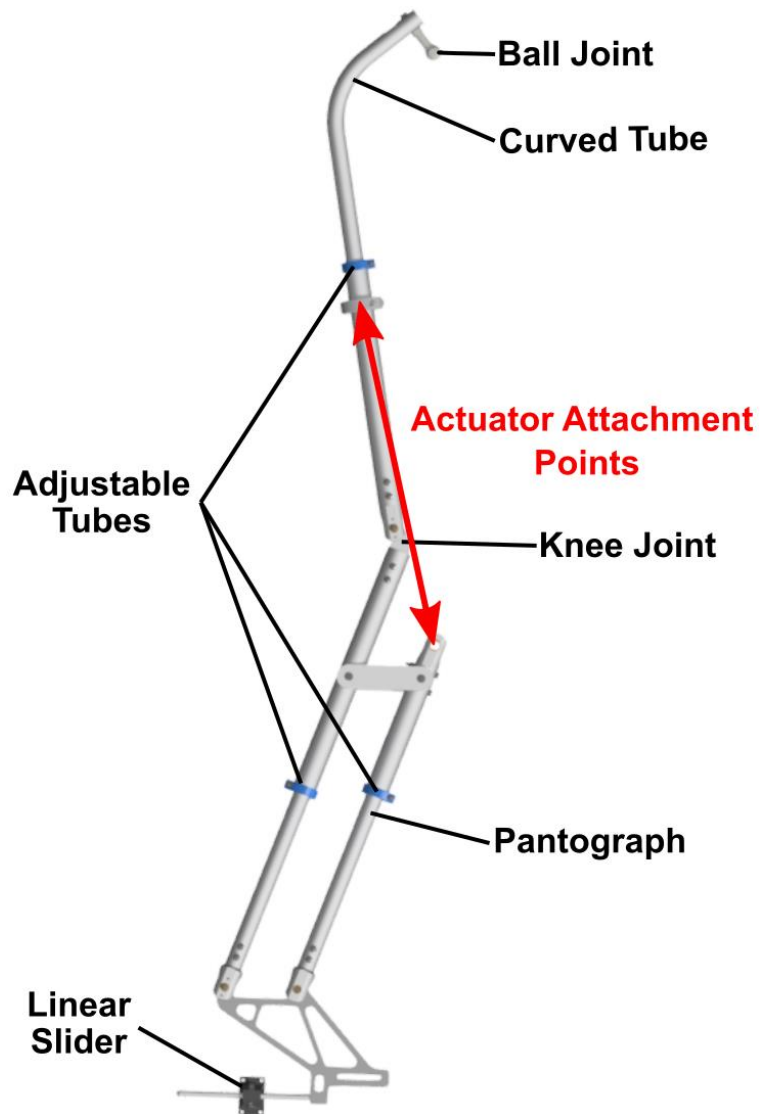


Figure 2.2: Exoskeleton frame CAD

Figure 2.2 shows the full frame of one exoskeleton leg with the key features to be discussed labeled. At the waist, a ball joint connects the top of the exoskeleton to the user, permitting free rotation in several directions. At the hip of the exoskeleton, the frame features a curved tube. Since the exoskeleton attaches to the waist belt over the iliac crest (located at the top of the hip), the

curve serves to shift the top of the frame further back, allowing it to stay in-line with the leg throughout the motion of walking. The knee joint, shown below in Figure 2.3, was designed with an angled extension on the bottom to allow for a deeper bend in order to avoid limiting the range of motion of the user's knees.

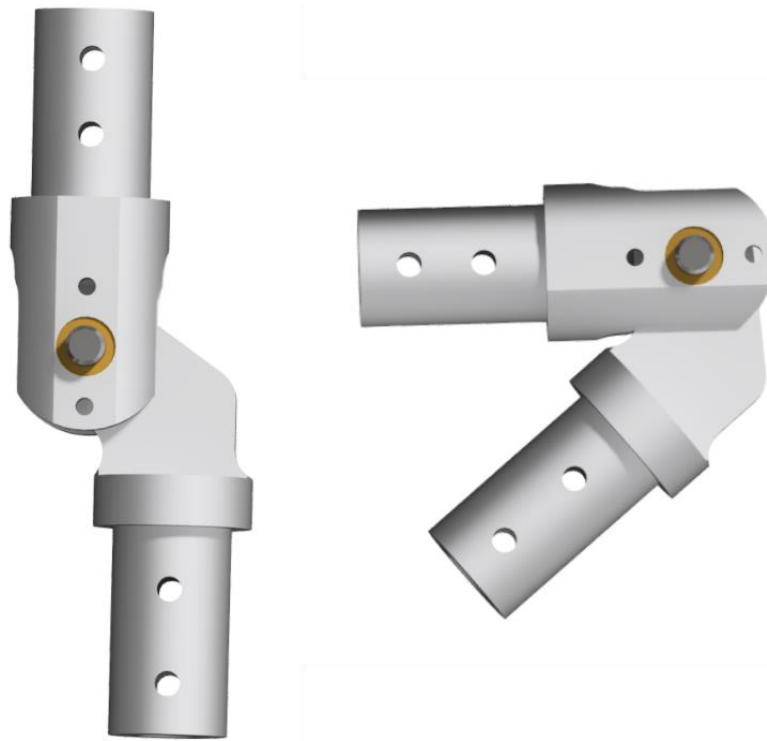


Figure 2.3: Exoskeleton knee joint CAD

A pantograph configuration is used for the bottom portion of the frame in order to reduce the necessary length of the actuator and to prevent the actuator from restricting movement of the foot when force is applied. In the exoskeleton, the pantograph includes a link that is parallel to the bottom of the foot, and a parallel bar close to the knee. These are connected with two links following the lower leg so as to form a parallelogram. Thus, the pantograph essentially copies the shape of the foot and translates it up closer to the knee. In general, pantographs take advantage of geometry by scaling up a shape and strategically attaching the end points so that the movement of

the shape and the larger projection are the same. This means that the actuator can push on the smaller shape of the leg, seen in green in Figure 2.4, and produce a force at the equivalent point of the larger shape of the leg seen in red. Specifically, the lower actuator attachment point is on the equivalent of the ball of the foot, so the forces of the exoskeleton are directed through the ball of the foot. This has the added benefit of allowing the user to lift their heel even while force is being applied at the ball of the foot.

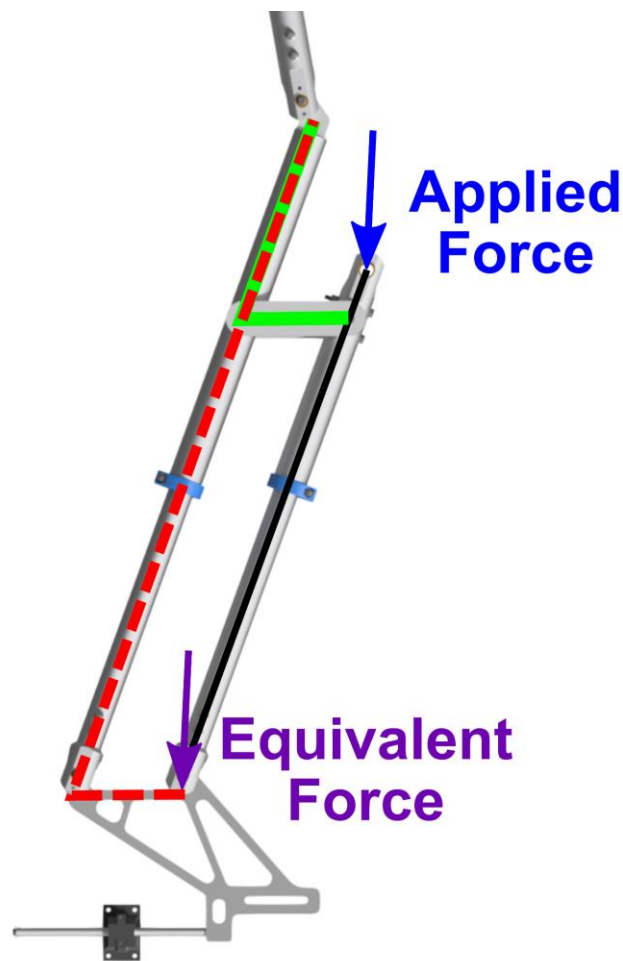


Figure 2.4: Pantograph example diagram

Both the top and bottom of the exoskeleton frame were designed using a telescoping tube and clamp system to allow for adjustability. This allows the size of the exoskeleton to be fit to different users to improve comfort and allow the exoskeleton to follow the user's body as closely as possible. The ball joint at the hip and the linear rail and slider at the foot all provide more range of motion to the user, the extent of which will be discussed in Chapter 6.1: Range of Motion.

2.2 Foot Attachment

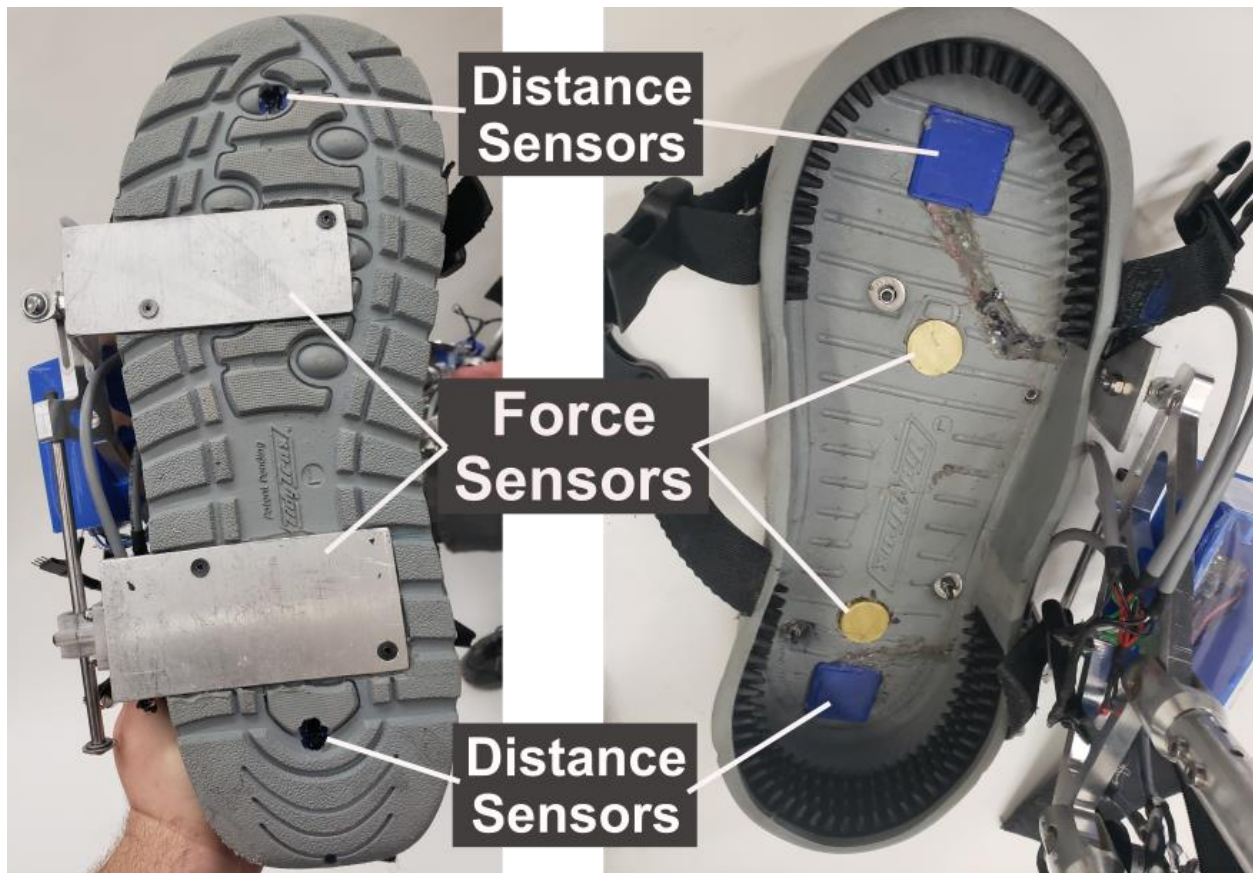


Figure 2.5: Labeled Overshoe

To attach the exoskeleton to the user's foot, Tidy Trax Shoe Covers [24] were used so that the user would be able to use their own shoes with the exoskeleton. Additional straps were adhered to the sides of the shoe covers to further secure the foot to the exoskeleton. In each shoe cover, a distance sensor and a force sensor are located at both the ball of the foot and the heel of the foot for the purpose of gait detection. The distance sensors are placed in a 3D printed case and situated over a hole that goes through the bottom of the shoe cover to allow the sensor to detect the ground. Each force sensor is sandwiched between a 3D printed fixture and a metal plate. The 3D printed fixture sticks through the top of the shoe cover and directs the force of the foot onto the center of the force sensor. The metal plates on the bottom of the shoe cover are L shaped plates which extend to the sides of the shoe cover; the plate at the ball of the foot attaches to the exoskeleton with a ball joint, while the plate at the heel of the foot attaches to the linear rail discussed in Chapter 2.1 with a bearing clamped in a 3D printed holder. Views of the top and bottom of the foot attachment can be seen above in Figure 2.5.

2.3 Soft Goods

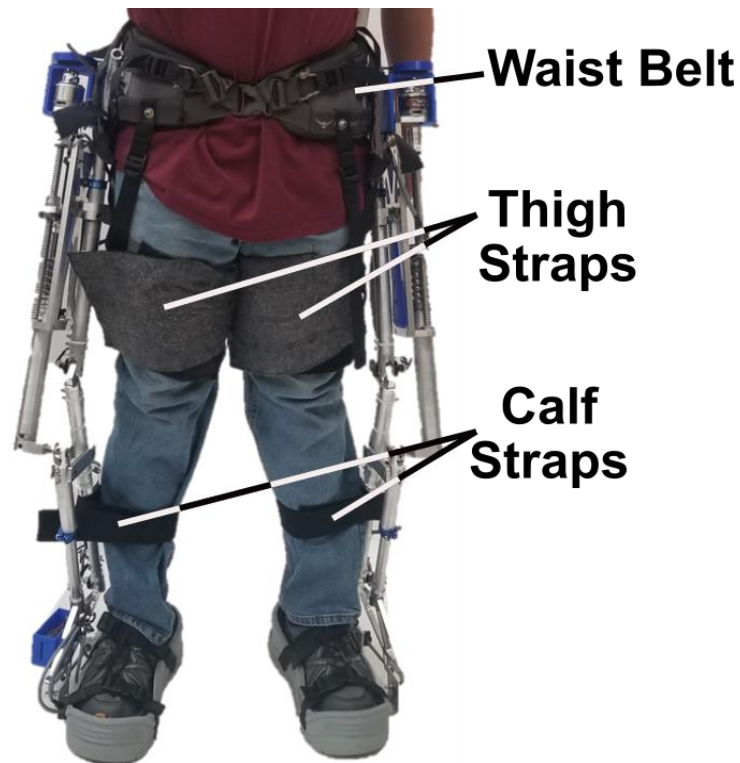


Figure 2.6: Exoskeleton soft goods (human interface)

Since a comfortable interface between the exoskeleton and the user is important for the function of the exoskeleton, soft goods were used to fix the exoskeleton to the body during use. Figure 2.6 shows each component of the soft goods. A waist belt is used to position and secure the ball joint at the top of each leg over the iliac crest, as discussed in Chapter 2.1, and prevents the exoskeleton from shifting downward on the user. The thigh straps are made from cloth and fasten around the leg in a conical shape using hook and loop straps. They do not connect directly to the exoskeleton, but just to the waist belt. When tightened, this prevents the waist belt from shifting up over the user's torso when force is applied. The calf straps provide another connection point directly between the exoskeleton and the user and help the exoskeleton to better follow the motion of the user's legs.

2.4 Actuator

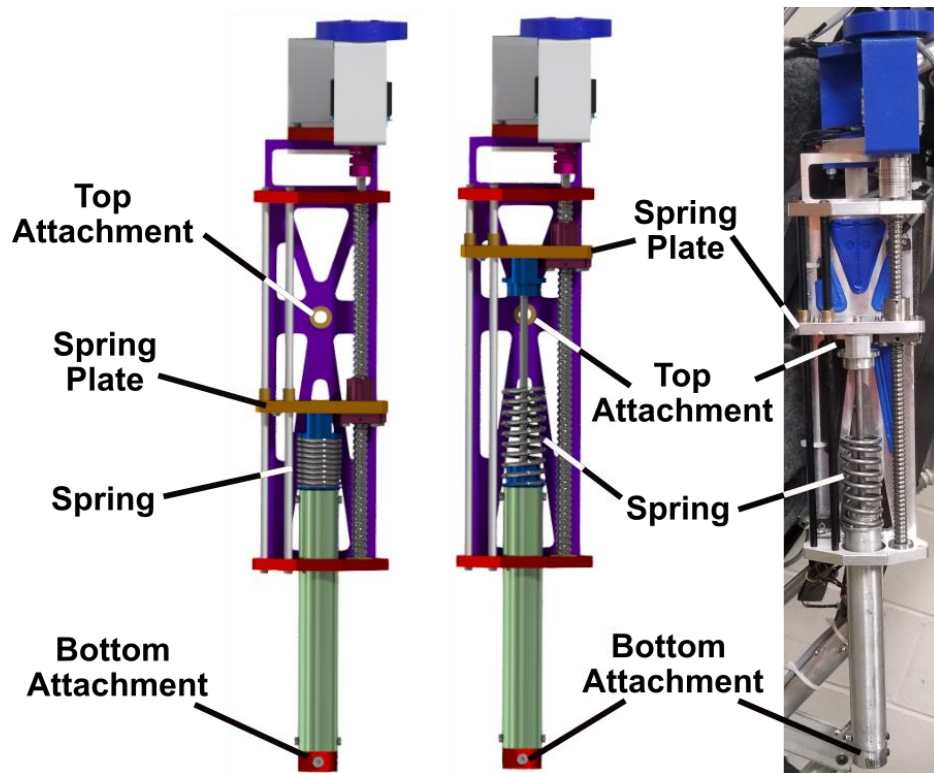


Figure 2.7: Compressed and Uncompressed Actuator Comparison

The actuator of this exoskeleton operates by compressing and decompressing a spring that is in parallel with the user's leg. A motor coupled to a ball screw controls the position of the spring plate (labeled above in Figure 2.7). When the user's foot is on the ground and a support force is desired, the spring plate is moved down to compress the spring the desired amount. When the foot is off of the ground, the spring plate moves out of the way to allow for the unimpeded motion of the leg. Since the spring plate is not rigidly coupled to the bottom attachment point of the actuator, the position of the spring plate will not prevent the user from extending their leg while walking.

Chapter 3: Electrical/Sensor Design

3.1 Sensors/Components

For the purpose of detecting and predicting the gait cycle of the user, the VL6180X Time of Flight distance sensors [25] and the FX29K0 force load cells [26] were used. The distance sensors can read from 0 mm to 100 mm and can operate at 10 Hz with a noise standard deviation of 2 mm (from the datasheet). When operated at 100 Hz, the noise standard deviation was 8.3 mm (calculated from our tests with 10 sets of 100 measurements). The force sensors can detect up to 445 N (100 lbf) and can run reliably at 119 Hz in sleep mode or 333 Hz in non-sleep mode. Both sensors operate using Inter-Integrated Circuit (I2C) protocol and since four of each sensor were used and each sensor of the same type has the same I2C address, the TCA9548A I2C Multiplexer [27] was used to allow communication with multiple sensors at the same address. To improve noise rejection on the I2C lines, the LTC4311 I2C Extender/Active Terminator [28] was used to boost the signal strength as it traveled from the sensors located on the foot, to the microcontroller (Chapter 3.2.1) located on the user's back. Additionally, while shielded cables were used throughout the exoskeleton, the I2C lines were modified to make use of ethernet or Category 5 (Cat 5) cables which have even greater noise rejection.

In order to determine the orientation of the exoskeleton and derive the needed location of the spring plate, two MAE3 Absolute Magnetic Kit Encoders [29] were used for each leg; one on the knee joint, and one on the ankle joint. The model used for this project has an analog output and can read at a sampling rate up to 2.6 kHz.

Two types of switches were used for this exoskeleton. The MS Series Small Size Switch [30] and the SV-166-1C25 Micro Limit Switch [31]. Four of the MS Series Small Size Switches were used to set the boundaries of the spring plate, one at the top of each actuator and one at the bottom. During normal operation, if either is hit, the spring plate would be returned to a safe position and the program stopped. The switch on top was also used to set the zero position of the spring plate before entering the main code. The SV-166-1C25 Micro Limit Switch was used as a dead man's switch to allow the user to cut off the exoskeleton by releasing the button. This switch was housed in a 3D printed handle to provide a comfortable grip for the user during operation and can be seen below in Figure 3.1.



Figure 3.1: Dead Man's Switch

The electrical power for the exoskeleton was supplied by two batteries: an everyday 9 Volt battery, and a Turnigy Graphene 6S Lithium Polymer (LiPo) Battery [32]. The 9 Volt was used to power the analog joint encoders, which required 5 Volts. In order to step down the voltage, a version of the LM2596 Buck Converter [33] with a light emitting diode (LED) display was used to regulate and monitor the voltage. The 6S LiPo Battery was used to power the motor controller (Chapter 3.2.2 below) and motors. A Turnigy LiPo Battery Cell Display and Balancer [34] was used to maintain equivalent charge across the LiPo battery cells and to monitor the charge of the battery. The microcontroller and I2C sensors are powered through the connection from the microcontroller to the laptop that records the data from the microcontroller.

To move the spring plate, one Turnigy Aerodrive SK3 Brushless Outrunner Motor [35] was used for each leg. The motor has a maximum loading of 60 Amps and can supply up to 1350 Watts of power. It is capable of achieving 500 rotations per minute (RPM) for each volt applied (500 KV). To appropriately control the motors, an E3 Optical Kit Encoder [36] was mounted to the shaft of each motor. This is a quadrature encoder with 1000 counts per revolution (CPR) and includes an index pulse.

3.2 Controllers

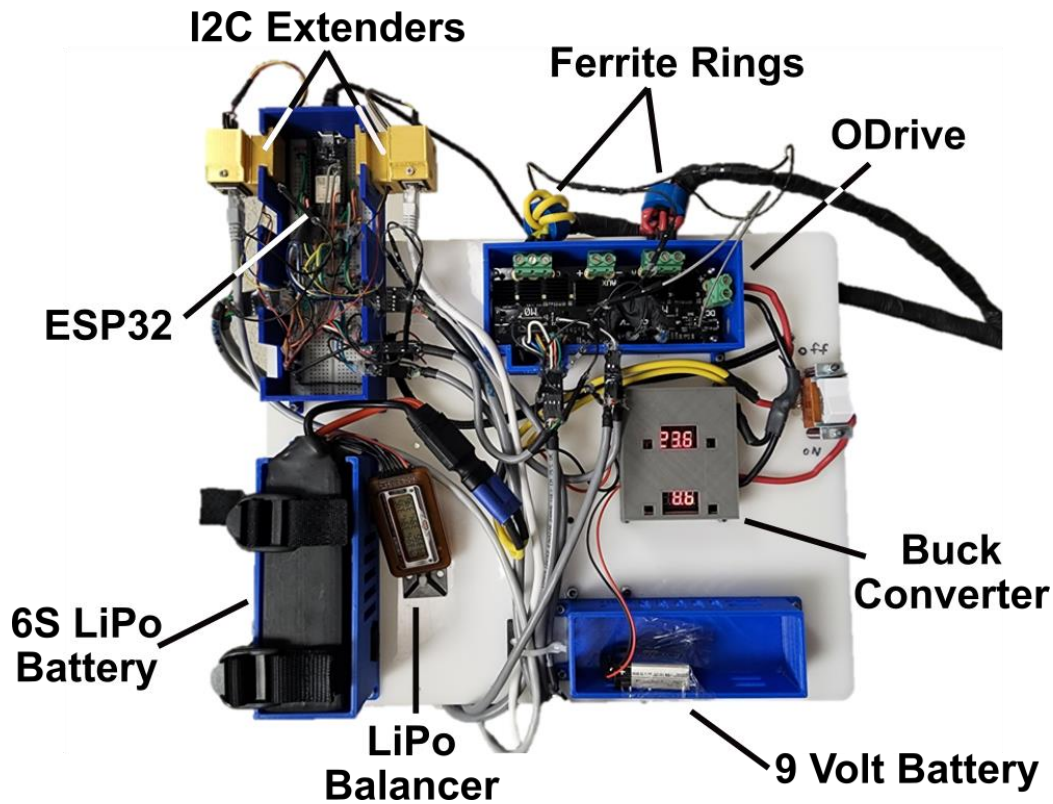


Figure 3.2: Exoskeleton electronics back plate

3.2.1 ESP32

An ESP32 Feather microcontroller [37] [38] was used for data collection, data processing, and operation of the exoskeleton. This microcontroller operates with 3.3 Volt power and logic and can run with a clock frequency up to 240 MHz. The dual core architecture of the ESP32 allows for the simultaneous operation of different codes, helping to prevent sensor readings from slowing down the operation of the exoskeleton. There are a large number of general-purpose input/output (GPIO) pins, analog input pins, a set of Serial pins that allow for universal asynchronous receiver/transmitter (UART) communication, and a set of I2C pins that are all configured by

default, with more capable of being configured. The I2C interface of the ESP32 supports both Standard mode (100 Kbit/s) and Fast mode (400 Kbit/s). The ESP32 is convenient for use because it is able to operate using the Arduino IDE, meaning many of the Arduino libraries written for the sensors being used on this exoskeleton are still usable.

3.2.2 ODrive

The 56 Volt ODrive V3.6 motor controller [39] was used for accurately controlling the movement of the two motors. Since the ODrive architecture is largely structured around Python, the initial setup and tuning can be performed using the Python utility “odrivetool” and saved to the motor controller. For the initial setup and tuning, the motors were secured in a 3D printed test stand without any load applied. The initial setup values can be seen in Appendix A. The control structure of the ODrive consists of a cascading position, velocity, and current control loop. In position control mode, all control loops are used. The position control loop is a proportional (P) loop while both the velocity and current control loops are proportional-integral (PI) loops. When tuning, only the position gain, velocity gain, and velocity integrator gain are directly adjusted to affect the motor performance [40]. The tuning values can also be seen in the initial setup values in Appendix A. In order to communicate between the ESP32 and the ODrive, UART communication was used along with the ODriveArduino library [41].

One of the greatest difficulties faced during this project was acquiring the desired performance from the ODrive and dealing with the effects of the noise caused by the motors on our other sensors. As was briefly mentioned before, shielded wires, I2C extenders, and ethernet cables were

implemented to improve noise rejection. Additionally, ferrite rings were added to the motor lines along with shielding to reduce the noise produced. The main difficulty with the ODrive was achieving the desired motor movement without reaching the current limit. Initially the measured current limit of the ODrive was around 30 amps, much below the set current limit of 60 amps. To lower the required current for movement, mechanical improvements were made to the actuators, including re-reaming holes, sanding down parts, and tightening the ball nut on the ball screw. With these improvements and the addition of lubricant to the system, which was initially avoided due to material selection, but later determined to be acceptable, the current required was greatly decreased. Finally, it was discovered that the current of the ODrive was being further limited by our initial use of a buck converter. Originally, two 4S batteries were used in series and a buck converter stepped down the voltage to a constant 25 volts, but the buck converter has a listed maximum current output of 3 amps. Once the single 6S battery was used directly, the ODrive performance improved to the expected level. This configuration was used in the final exoskeleton.

The ODrive provides multiple methods of position control. For this exoskeleton, the Trapezoidal Trajectory method was used to set a velocity limit, acceleration limit, and deceleration limit. The ODrive would then calculate the speed and acceleration necessary to allow the motor to reach the desired position as quickly as possible without exceeding the supplied limits [42]. This allowed more direct control over the current draw of each motor since the current could be decreased by reducing the acceleration limit. Values of the limits supplied are included in the initial setup values in Appendix A.

A full electrical diagram including all sensors, components, and controllers, updated from the version presented by Claessen [43], can be seen in Appendix B.

Chapter 4: Software

4.1 ESP32 Code Structure

The software used to operate the exoskeleton was a real-time code relying on software interrupts to set the sampling frequency of the sensors. The distance sensors, joint encoders, and motor encoders were set to read at 100 Hz while the force sensors were set to read at 200 Hz. As mentioned in Chapter 3.2.1, the ESP32 microcontroller has a Dual Core system; this allowed the majority of the sensor readings and gait detection to be housed on Core 0 and run simultaneously with the motor control commands on Core 1. This helped to prevent the sensor readings from slowing down the operation of the rest of the code.

When begun, the code first performed the setup for all sensors and waited for the ODrive motor calibration to complete. The ODrive must perform an offset calibration on every power up to find the index point of the motor encoder. If unsuccessful, the motor would not be operational. Once the ODrive calibration was successful, the ESP32 would zero the motors by slowly moving them until they hit the top limit switch and recording the position when that occurred. Since it was found that reading the motor position from the ODrive was very slow, the A and B signals of the motor encoders were additionally connected to the ESP32 for direct readings. The direct read position was synced with the ODrive position reading during motor zeroing.

It was found that the joint encoder data experienced a lot of noise, so a second-order Butterworth filter with a 3 Hz cutoff frequency was applied to the data. Since the initial filtered values are very

inaccurate, ten filtered readings were taken after the motor zeroing so the first readings actually used in the code would be more accurate.

After the initial setup, on Core 0, the distance sensors, force sensors, and motor encoders were read when readied by the interrupts. The distance and force sensor values were then used in one of the gait determination algorithms, discussed in Chapter 4.2 and 4.3, to find the desired spring compression. On Core 1, the joint encoders were read and filtered when readied by the interrupts and used to calculate the length of the actuator based on Equation (5.6) and Equation (5.7) in Chapter 5.2. Using the actuator length and the desired compression from the gait determination algorithm, Equation (5.10) from Chapter 5.3 was used to get the desired motor position which was sent to the motor. Before being sent to the motor, if the leg was in Swing Phase, the position was filtered with a second-order Butterworth filter with a 10 Hz cutoff frequency to reduce unnecessary movement. During Stance, the unfiltered motor position was used. This code ran simultaneously for each leg of the exoskeleton with no code dependency between legs, allowing for a more simple code structure. A flowchart of the code structure on both cores can be seen below in Figure 4.1.

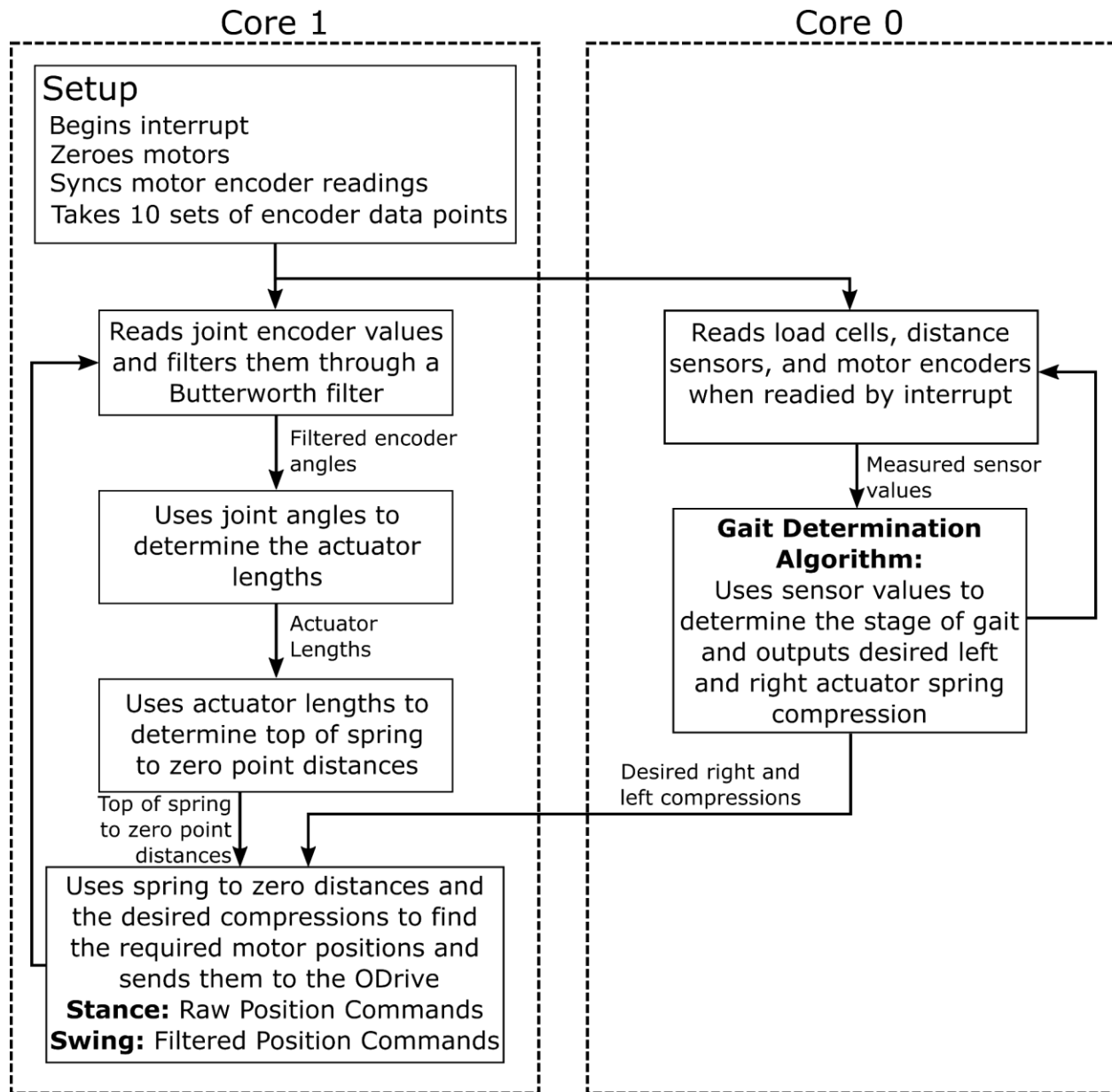


Figure 4.1: ESP32 code flowchart. Modified from [43]

Safety was an important consideration in the mechanical, electrical, and software design of the exoskeleton. In addition to performing failure calculations, multiple safety mechanisms were implemented to protect the user. Physical limit switches were installed on the top and bottom reaches of the actuators that would trigger the ESP32 to stop the code and return the spring plates to a safe position. The chosen safe position was another safety consideration to make sure that if

an error was triggered during operation, the position of the spring plate would be moved to not compress the spring so that the user could still walk. Digital limits were coded into the software to prevent the motors from being sent commands that would take them outside of the operational range. A Dead Man's switch was implemented to allow the user a quick way of turning off the exoskeleton if needed, but this method would not allow the motors to be moved to a safe position. Lastly, cases were designed to cover the actuator and protect the user and anyone nearby from pinch points or shrapnel in the case of failure (Seen in Figure 4.2).



Figure 4.2: Exoskeleton actuator safety cover

4.2 Gait Detection: Force Method

The primary gait detection algorithm used only relied on the force sensors. It read the heel and ball forces and used a threshold value for the heel force to determine if the leg was in Swing or Stance. If in Swing, the compression was set to -15 mm (off of the spring). If in Stance, the compression was set to achieve the desired support force and the signs of the heel and ball force slopes were monitored. If the heel force was decreasing and the ball force was increasing for enough readings, a foot lift was predicted, and Swing was entered. If the heel force was increasing and the ball force decreasing, then no lift was predicted, and the leg remained in Stance.

A threshold value of 35.6 N (8 lbf) was used for monitoring the heel force. This was determined experimentally by viewing walking data collected from the exoskeleton and analyzing the predicted heel strike location using different threshold values. This was initially done with tests of the exoskeleton without support force applied in order to get a preliminary value. Then using the preliminary value, tests were performed with support force and the threshold value was updated. Pseudocode of the gait determination algorithm relying on the force sensors can be seen in Figure 4.3.

Gait Determination Algorithm 1:

```
1: take in heel_force and ball_force
2: if phase == SWING
3:   compression ← -15 mm offset
4:   if heel_force > heel threshold
5:     phase ← STANCE
6:     compression ← support force compression
7: else
8:   heel_force_slope ← heel_force - prev_heel_force
9:   ball_force_slope ← ball_force - prev_ball_force
10:  switch(stance_state)
11:    case 1: No_Lift_Predicted
12:      compression ← support force compression
13:      if heel_force_slope ≤ 0 & ball_force_slope ≥ 0
14:        increment stance_counter
15:      else stance_counter ← 0
16:      if stance_counter > lift prediction threshold
17:        stance_state ← Lift_Predicted
18:    case 2: Lift_Predicted
19:      compression ← -15 mm offset
20:      if heel_force_slope > 0 & ball_force_slope < 0
21:        stance_counter ← 0
22:      compression ← -15 mm offset
23:      phase ← SWING
24:      stance_state ← No_Lift_Predicted
25:  if heel_force < 10 & ball_force < 5
26:    stance_counter ← 0
27:    compression ← -15 mm offset
28:    phase ← SWING
29:    stance_state ← No_Lift_Predicted
30:  return compression
```

Figure 4.3: Gait determination algorithm pseudocode: Force method. *heel_force* and *ball_force* are the force at the heel and ball of the foot in lbs that was read earlier and brought into this algorithm, *prev_heel_force* and *prev_ball_force* are the force values from the last reading, “heel threshold” is the value that indicates a heel strike, *phase* is the current gait cycle phase of the leg, “SWING” means the leg is in Swing Phase, “STANCE” means the leg is in Stance Phase, *compression* is what value to compress the spring to in mm, “-15 mm offset” is the 15 mm distance

away from the spring to prevent accidental force during Swing, “support force compression” is the required spring compression to achieve the desired support force, *stance_state* is the condition heel during stance (either staying on the ground or lifting off of the ground), *heel_force_slope* and *ball_force_slope* is the current force value minus the previous force value at the heel and ball of the foot, *stance_counter* is the number of times in a row a heel lift was predicted, and “lift prediction threshold” is the number of heel lifts in a row that must be predicted for the leg to enter Swing.

4.3 Gait Detection: Distance Method

The second gait detection algorithm used only relied on the heel distance sensor. It read the heel distance from the ground and used two threshold values to determine if the leg was in Swing or Stance. If the leg was in Swing, the compression was set to -15 mm (off of the spring). From Swing, if the heel distance was equal to or below the start stance threshold, Stance was entered. In Stance, the compression was set to achieve the desired support force. If the heel distance was equal to or greater than a different end stance threshold, Swing would be entered.

Like for the force method, the threshold values for this method were initially found by analyzing the walking data. While values were chosen to test, due to the amount of noise in the distance sensor data, a reliably functioning set of values was not found for this method that did not experience rapid undesired shifts between Swing and Stance or inappropriately enter Stance while the foot was low during Swing Phase. Pseudocode of the gait determination algorithm relying on the heel distance sensor can be seen in Figure 4.4.

Gait Determination Algorithm 2:

```
1: take in heel_distance
2: if phase == SWING
3:   compression  $\leftarrow$  -15 mm offset
4:   if heel_distance  $\leq$  start stance threshold
5:     phase  $\leftarrow$  STANCE
6:     compression  $\leftarrow$  support force compression
7: else
8:   if heel_distance  $\geq$  end stance threshold
9:     phase  $\leftarrow$  SWING
10:    compression  $\leftarrow$  -15 mm offset
11: return compression
```

Figure 4.4: Gait determination algorithm pseudocode: Distance method. *heel_distance* is the distance from the ground at the heel in mm that was read earlier and brought into this algorithm, “start stance threshold” is the distance value that indicates that heel strike is about to occur, “end stance threshold” is the distance value that indicates that the heel is lifting from Stance, *phase* is the current gait cycle phase of the leg, “SWING” means the leg is in Swing Phase, “STANCE” means the leg is in Stance Phase, *compression* is what value to compress the spring to in mm, “-15 mm offset” is the 15 mm distance away from the spring to prevent accidental force during Swing, “support force compression” is the required spring compression to achieve the desired support force.

Chapter 5: Calculations

5.1 Joint Moment Calculations

To examine the possible impact of the exoskeleton on the user, calculations were performed to find the moment at the hip, knee, and ankle joints resulting from the support force at the waist. The equations for the moments were derived and implemented in MATLAB using biological joint angle and moment data from Perry and Burnfield's Gait Analysis: Normal and Pathological Function [44], 50th percentile thigh, shin, and ankle length data from a U.S. Army Anthropometric Survey [45], 50th percentile foot length data from a study by Hajaghazadeh [46], and horizontal and vertical GRF data from Kirtley's Clinical Gait Analysis: Theory and Practice [47]. The limb length values for both men and women are recorded below in Table 5.1. For the calculations, the averaged value, also shown in Table 5.1, was used.

Table 5.1: 50th percentile male, female, and averaged limb length values in mm.

	Male	Female	Average
Waist Length (mm)	162	150	156
Thigh Length (mm)	408	379	393.5
Shin Length (mm)	418	402	410
Ankle Length (mm)	73	63	68
Heel-to-Ball Foot Length (mm)*	117.3	115.1	116.2

*Foot Length calculated by subtracting the average length from the heel to the lateral and medial malleolus from the length of the heel to the ball of the foot

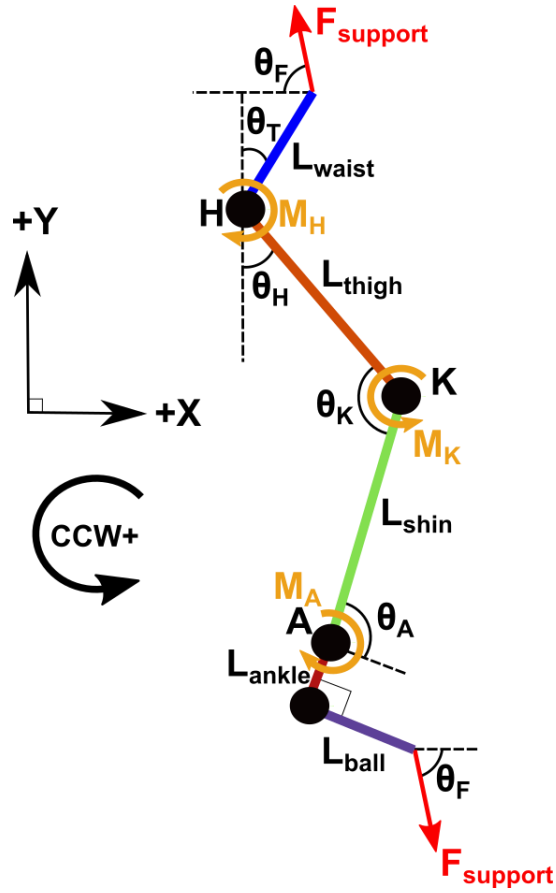


Figure 5.1: Leg free body diagram

To derive the moment equation for each joint, the free body diagram (FBD), above in Figure 5.1, was used to take the sum of moments ($\sum M$) for each individual segment about the point of interest. For the hip,

$$\sum M = -M_H + M_{FH} = 0 \quad (5.1)$$

where M_H is the moment at the hip in N*m and M_{FH} is the moment at the hip resulting from the support force in N*m.

M_{FH} was then found by summing the moment contributions of the support force in the x and y direction:

$$M_{FH} = (L_{waist} * \cos(\theta_T) * F_{support} * \cos(\theta_F)) + (L_{waist} * \sin(\theta_T) * F_{support} * \sin(\theta_F)) \quad (5.2)$$

where L_{waist} is the length between the hip and the exoskeleton upper attachment in meters, θ_T is the angle between the vertical and the waist in degrees, θ_F is the angle of the support force in degrees, and $F_{support}$ is the magnitude of the support force in Newtons.

After simplifying the equation using trigonometry identities, the moment at the hip is given by:

$$M_H = M_{FH} = (L_{waist} * F_{support}) * \cos(\theta_T - \theta_F) \quad (5.3)$$

Similar derivations were performed for the thigh and shin in order to find the moment at the knee and ankle respectively and can be seen in Appendix C. It was found that at the knee,

$$M_K = -(M_H + (L_{thigh} * F_{support}) * \cos(\theta_H + \theta_F)) \quad (5.4)$$

where M_K is the moment at the knee in N*m, L_{thigh} is the length between the hip and the knee in meters, and θ_H is the angle between the vertical and the thigh in degrees.

At the ankle,

$$M_A = -M_K - (L_{shin} * F_{support}) * \cos(\theta_K + \theta_H + \theta_F) \quad (5.5)$$

where M_A is the moment at the ankle in N*m, L_{shin} is the length between the knee and the ankle in meters, and θ_K is the angle between the thigh and the shin in degrees.

In the above equations, the support force angle, θ_F , was found geometrically by summing the x distance contributions of each leg segment, then doing the same for the y contributions. The arctangent of the y value over the x value provided θ_F .

The joint moments were found both with a constant support force of 222.4 N (50 lbf), and with a support force proportional to the GRF with the point of 100% body weight being mapped to a support force of 222.4 N (50 lbf). These can be seen in Figure 5.2.

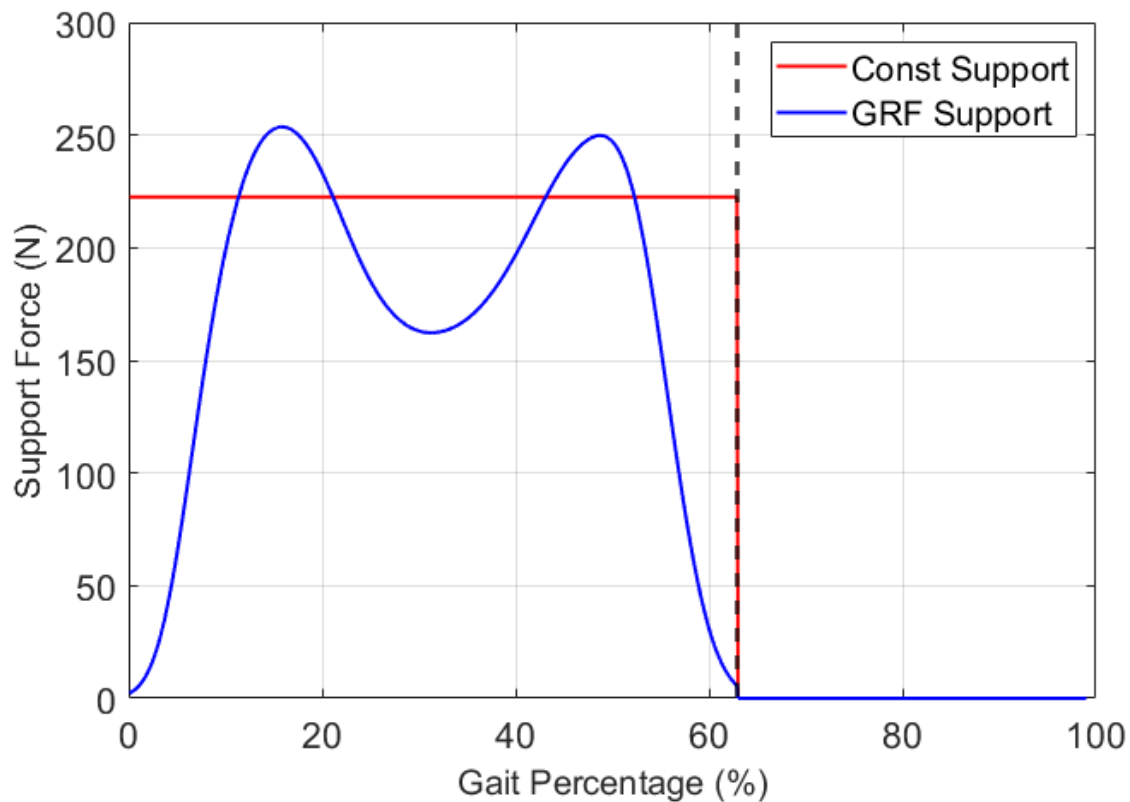


Figure 5.2: Support forces used for joint moment calculations

In order to validate the derived moment equations and the digitized biological data, a stick figure walking animation utilizing the joint angles, leg segment lengths, and key exoskeleton locations was generated and a second method of calculating the joint moments was used. Figure 5.3 shows a screenshot of the animation with the two exoskeleton attachment points labeled and the force line used to determine the support force angle displayed in red. Additionally, Figure 5.4 shows the movement of the stick figure through Stance Phase from 0% to 60% of the gait cycle.

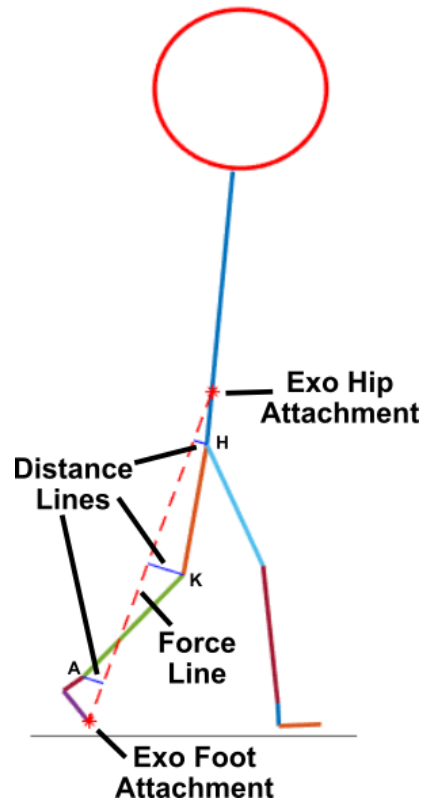


Figure 5.3: Stick figure code verification model. In the diagram, the letters H, K, and A denote the hip, knee, and ankle joints, respectively.

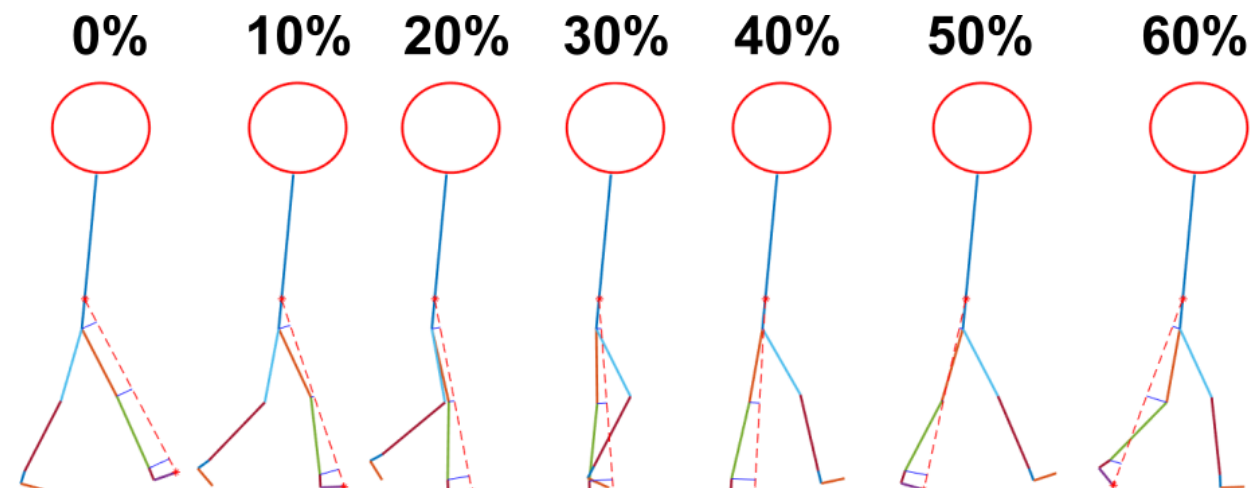


Figure 5.4: Stick figure movement through Stance Phase

The animation was able to provide a visual check that the joint angles and leg segment lengths successfully followed the expected motion of human walking.

The second method of calculating the joint moments was to find the perpendicular distance from each joint to the force line and multiplying the distance by the support force. This method was also validated by the stick figure animation by plotting the distance lines between the joints and the force line, the distances are shown in blue. A plot of the distance over the gait cycle for each joint is shown below in Figure 5.5.

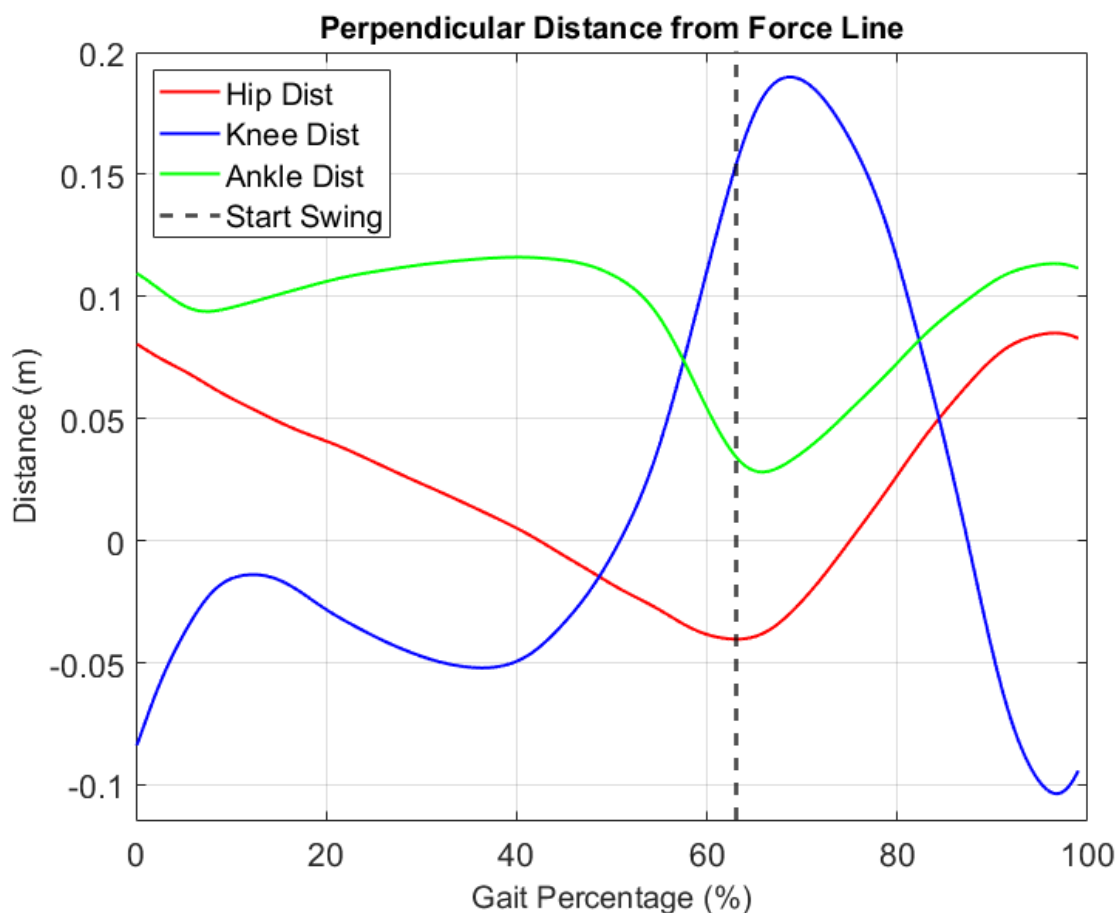


Figure 5.5: Distances from each joint to the line of force between the exoskeleton hip attachment and the connection at the foot

Figure 5.6 can be seen below and shows the moments at each joint calculated with the constant support force. While the next figure, Figure 5.7, shows the joint moments calculated with the support force proportional to the GRF.

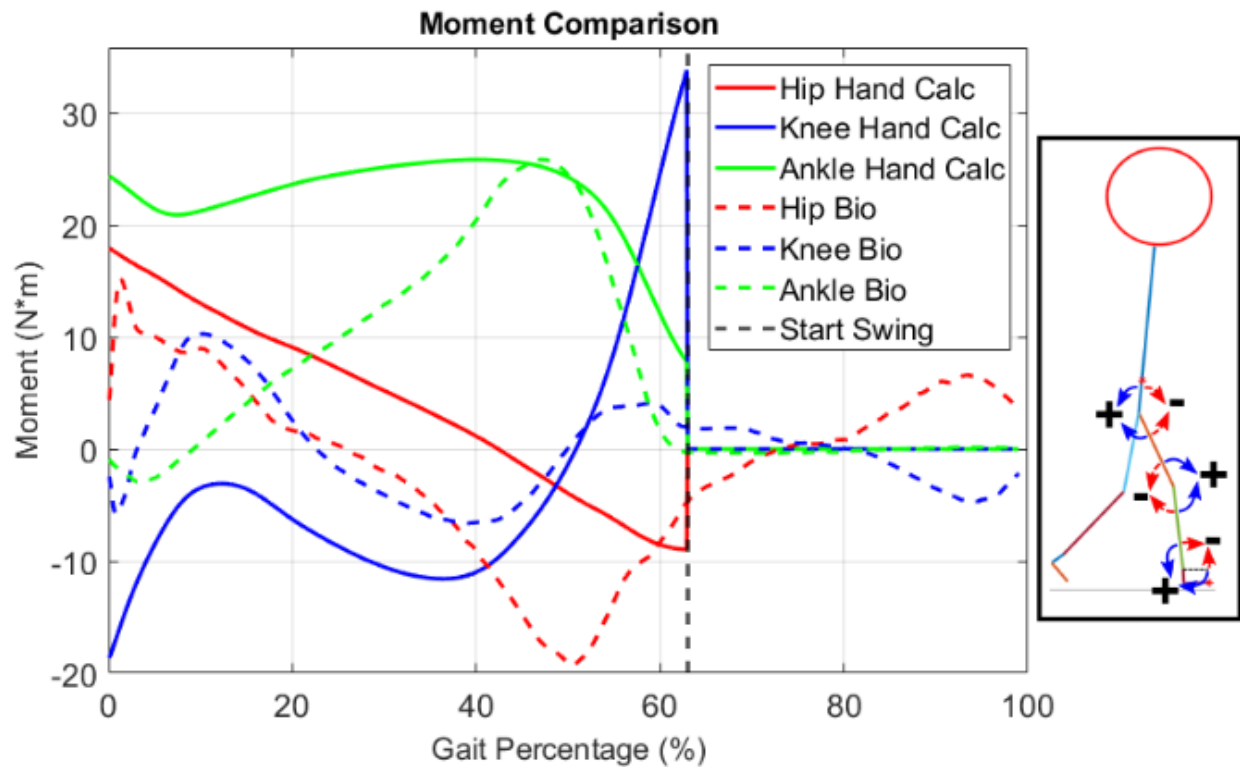


Figure 5.6: Joint moments caused by a constant support force and biological joint moments scaled so peak ankle moments match

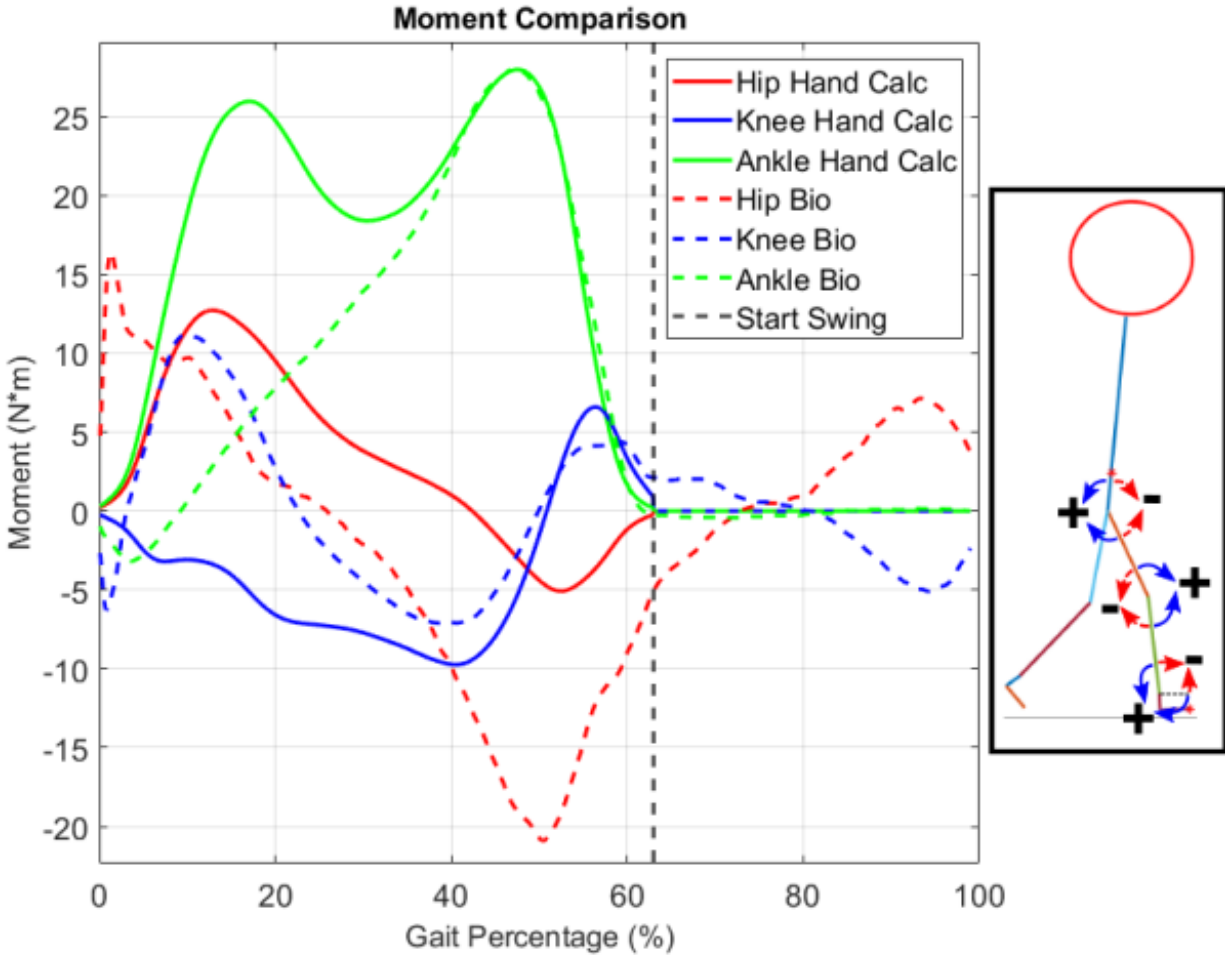


Figure 5.7: Joint moments caused by a force proportional to the GRF and biological joint moments scaled so peak ankle moments match

While the overall shape of the plots changes with the differing support force methods, the trends remain largely the same. The ankle moment consistently has the greatest magnitude until just before Stance and the hip moment is greater than the knee until around 50%. When the support force is proportional to the GRF, the ankle moment clearly follows the trend of the GRF. The calculated moments all drop to zero at 63% of the gait cycle because the leg enters Swing Phase where a support force is not being applied. The biological data has been scaled so that the maximum biological ankle moment matches the maximum calculated ankle moment.

Comparing the moment calculations with the support force proportional to the GRF to the scaled biological joint moments during Stance Phase, the overall trend of the hip moment most closely matches with an early peak and decrease and a shorter rise before Swing Phase. The knee and ankle moments both follow the biological trends during the second half of Stance, after around 35%.

The sign convention used for the calculated joint moments are oriented so that the moments are those produced by the exoskeleton to offset the biological joint moments. By subtracting the calculated moments from the biological moments, the estimated required moment at each joint is found. The assisted joint moment from the GRF support force can be seen in Figure 5.8 compared with the unassisted biological joint moments.

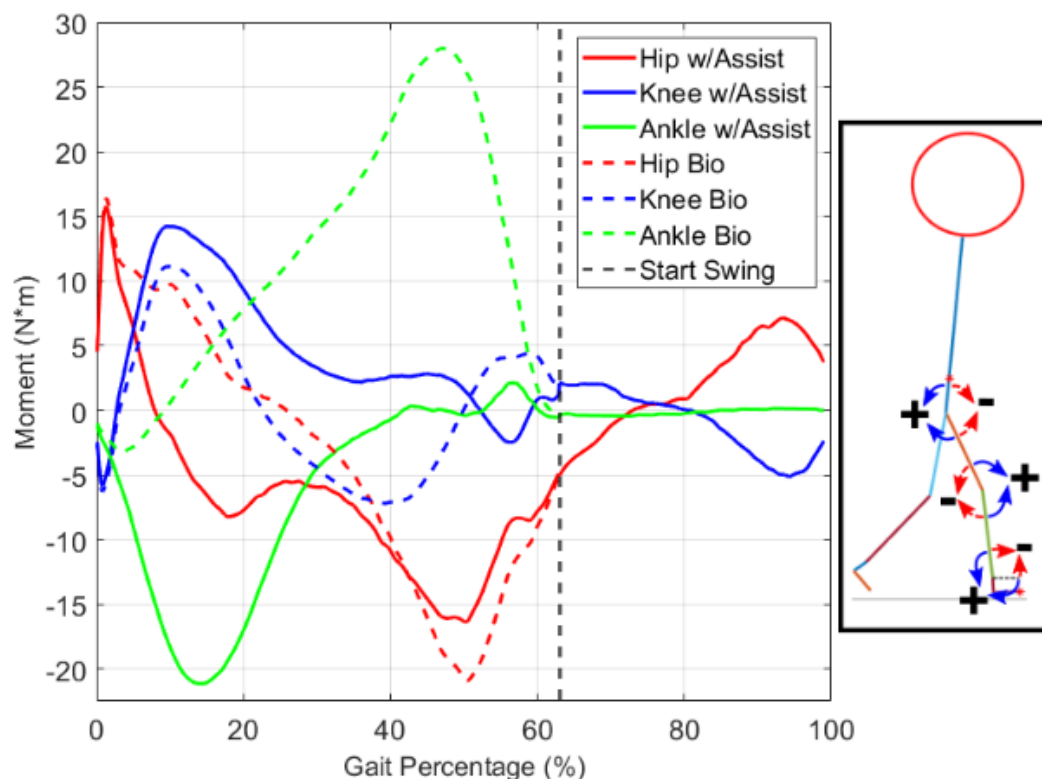


Figure 5.8: Joint moments assisted by a force proportional to the GRF and unassisted biological joint moments

5.2 Actuator Length Calculations

During the operation of the exoskeleton, it is important that the current length of the actuator is known at any given time. This allows the control program to know where the end of the spring is, and therefore where to move the spring plate to achieve the desired support force or to ensure the spring is uncompressed during Swing Phase. Figure 5.9 shows the length and angle definitions for the geometric equations used to calculate the actuator length with known lengths and angles measured by the encoders at the knee and ankle joint.

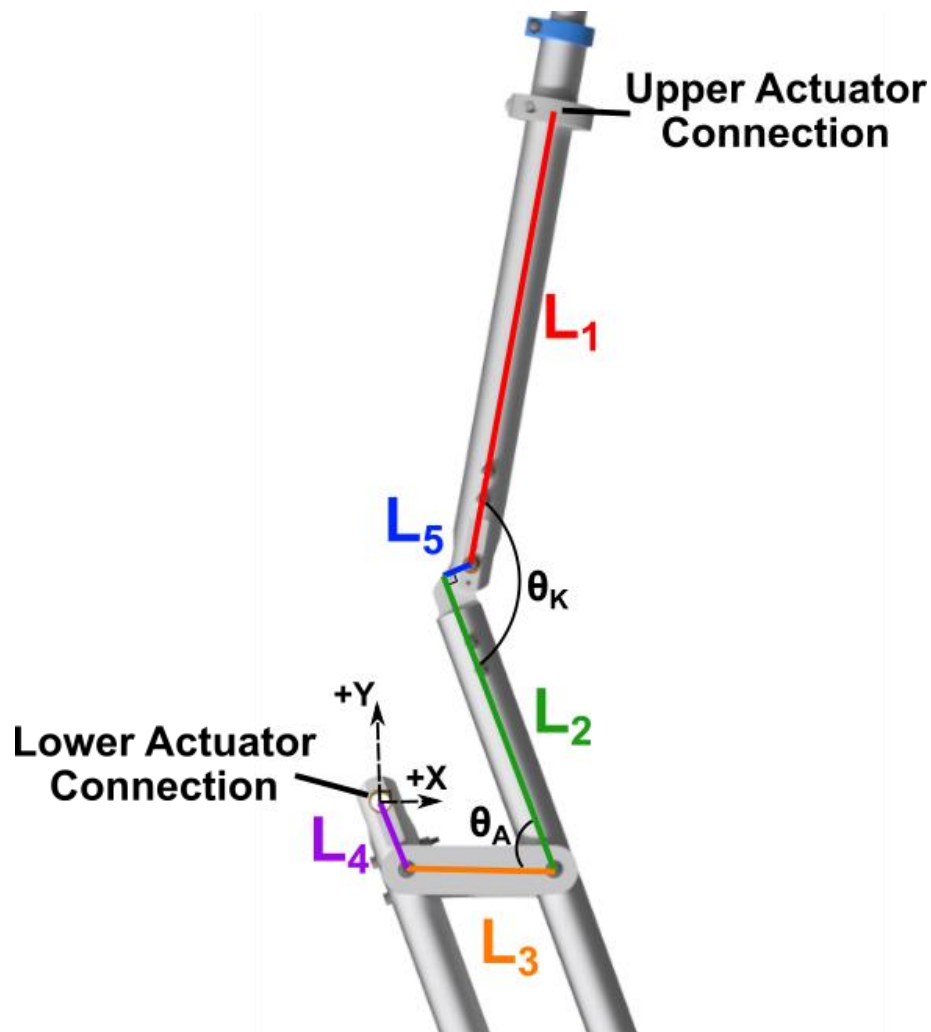


Figure 5.9: Labeled exoskeleton frame for actuator length calculations

Starting at the lower actuator connection point with the horizontal line parallel to L_3 being defined as the X axis, the sum of the x components (X) of each linkage is given by:

$$X = L_3 + L_4 \cos(\theta_A) - L_2 \cos(\theta_A) + L_1 \cos(\theta_K - \theta_A) + L_5 \sin(\theta_A) \quad (5.6)$$

where L_1 is the length from the upper actuator connection to the knee joint in meters, L_2 is the length from the knee joint to the upper pantograph connection that is parallel to the long lower leg of the exoskeleton in meters, L_3 is the length of the horizontal pantograph bar in meters, L_4 is the length from the pantograph connection on the short lower leg to the lower actuator connection in meters, L_5 is the length from the knee joint to the long lower leg perpendicular to the long lower leg in meters, θ_A is the ankle angle in degrees, and θ_K is the knee angle in degrees.

Starting from the lower actuator connection with the positive Y axis moving perpendicular to L_3 towards the upper actuator connection, the sum of the y components (Y) of each linkage is given by:

$$Y = L_2 \sin(\theta_A) - L_4 \sin(\theta_A) + L_1 \sin(\theta_K - \theta_A) + L_5 \cos(\theta_A) \quad (5.7)$$

Once X and Y are calculated, the arctangent of Y divided by X can be used to find the angle of the actuator and the resultant of X and Y can be used to find the length of the actuator.

5.3 Required Actuator/Motor Performance

An important piece of the exoskeleton design was selecting a motor capable of achieving the desired speeds and power output. To achieve this, the same biological joint angles and limb segment lengths used in Chapter 5.1 were used along with the equations for the GRF angle from that chapter and the equations for the actuator angle from Chapter 5.2. All of these angles plotted over the gait cycle can be seen below in Figure 5.10.

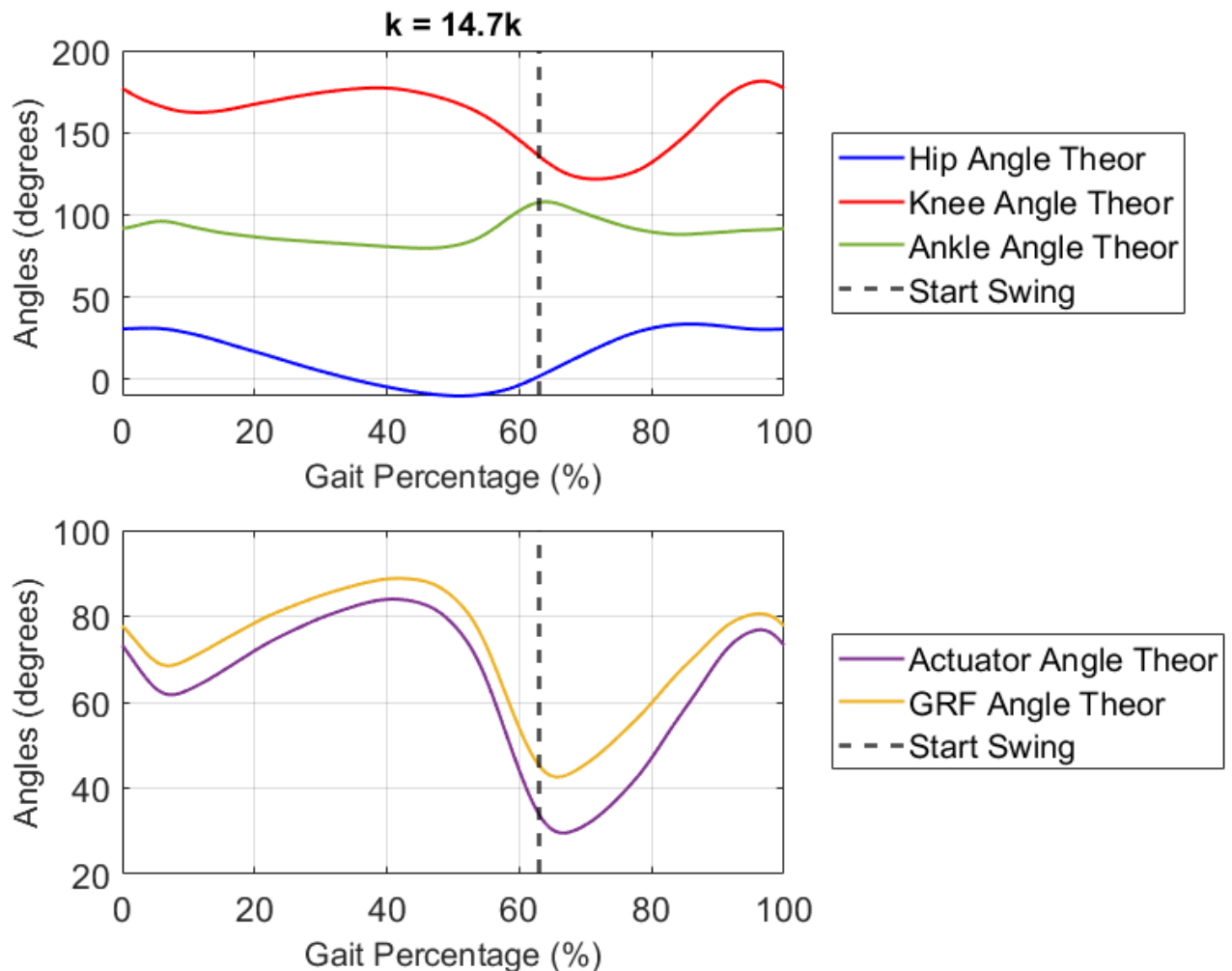


Figure 5.10: Biological joint angles and calculated actuator and GRF angles

Using these angles and the free body diagram in Figure 5.11, the support ratio relating the spring force from the actuator to the supportive force felt at the waist was derived by taking the sum of moments at the ankle joint A.

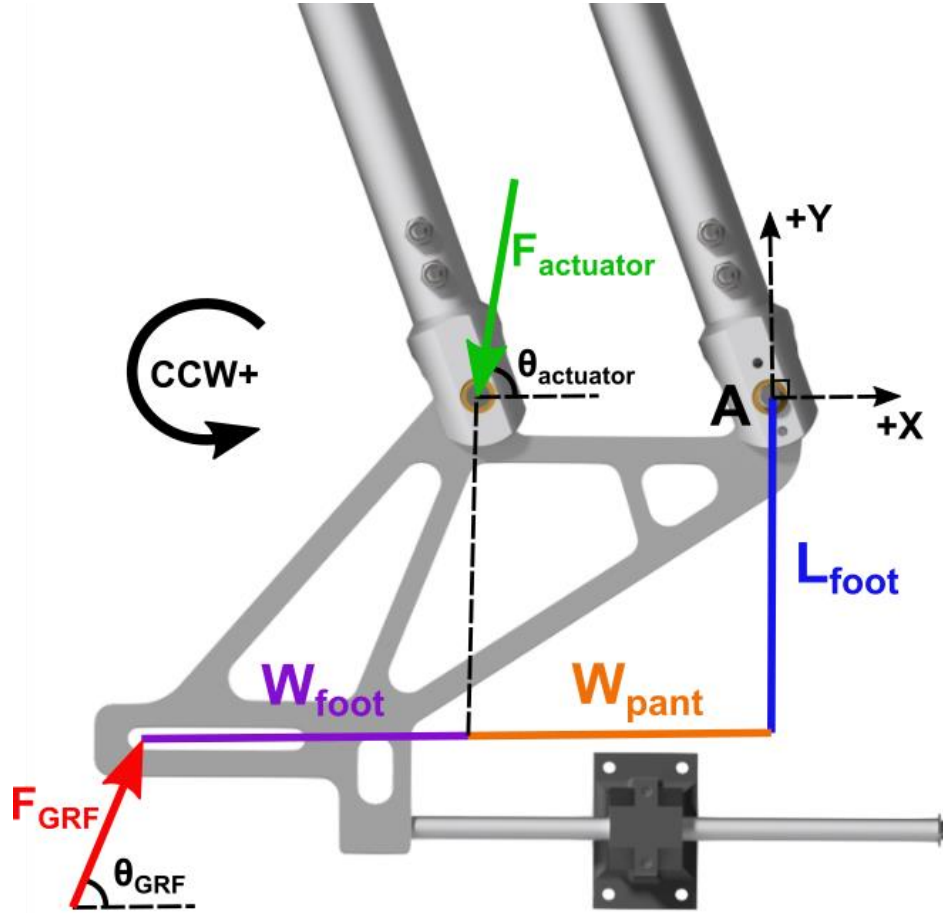


Figure 5.11: Free body diagram for support ratio derivation (A is the location of the ankle joint)

$$\begin{aligned}
 \sum M_A = 0 = & -(W_{foot} + W_{pant}) * F_{GRF} \sin(\theta_{GRF}) \\
 & + L_{foot} * F_{GRF} \cos(\theta_{GRF}) \\
 & + W_{pant} * F_{actuator} \sin(\theta_{actuator})
 \end{aligned} \tag{5.8}$$

where W_{foot} is the horizontal distance from the exoskeleton ball joint attachment at the foot to the outside pantograph leg in meters, W_{pant} is the horizontal distance between the two pantograph

legs in meters, L_{foot} is the vertical distance from the exoskeleton ball joint attachment at the foot to the pantograph connections in meters, F_{GRF} and θ_{GRF} are the magnitude and angle of the support force in Newtons and degrees respectively, and $F_{actuator}$ and $\theta_{actuator}$ are the magnitude and angle of the spring force of the actuator in Newtons and degrees respectively.

Rearranging this equation, the support ratio can be extracted and is plotted below in Figure 5.12.

$$\frac{F_{actuator}}{F_{GRF}} = \frac{(W_{foot} + W_{pant})\sin(\theta_{GRF}) - L_{foot}\cos(\theta_{GRF})}{W_{pant}\sin(\theta_{actuator})} \quad (5.9)$$

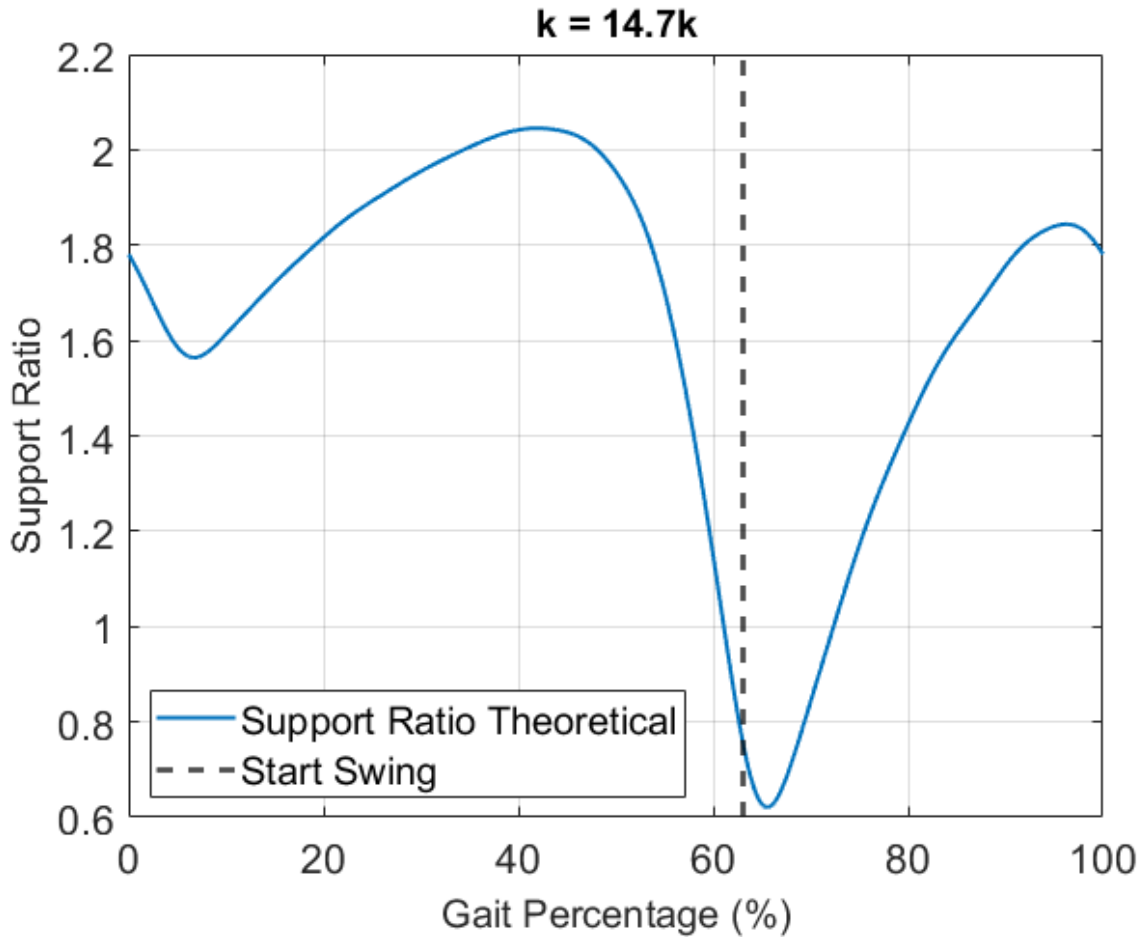


Figure 5.12: Support ratio between the force supplied by the actuator and the supportive force felt at the waist

With the support ratio, the required spring force to maintain 222.4 N (50 lbf) of support force at the waist during Stance Phase and no force during Swing Phase, and therefore the required spring compression to do so, can be calculated. Additionally, during Swing Phase, the compression is set to -15 mm so that a sudden movement of the leg during Swing does not cause a resistive force that disrupts the user's motion. To account for the amount of time it takes the motor to move the spring plate on and off of the spring, a linear decrease in compression was applied based on a chosen reaction time so that the spring plate reaches -15 mm of compression at the beginning of Swing Phase and a linear increase in compression was applied based on another chosen reaction time so that the spring plate reaches 0 mm of compression at the beginning of Stance and continues to the required compression for the desired force. The spring compression, spring force of the actuator, and force of the exoskeleton at the waist can be seen in the first plot of Figure 5.13 below. Since an immediate linear force change would cause inaccurate spikes in the data, a 3rd order lowpass filter with a 0.05 Hz half power frequency was applied to allow for a more realistic and smooth transition.

The second plot of Figure 5.13 shows the required position of the actuator plate. This position is calculated by the following equation:

$$Position = (L_{actuatorOffset} - L_{springTube}) + compression \quad (5.10)$$

where $L_{actuatorOffset}$ is the length of the actuator adjusted to include the distance from the upper actuator connection to the top limit switch in meters, $L_{springTube}$ is the length from the lower actuator connection to the top of the spring in meters, and $compression$ is the desired spring compression in meters.

The last plot of Figure 5.13 shows the velocity on the left axis and the acceleration on the right axis. The velocity in m/s was calculated by:

$$Velocity = \frac{\Delta Position}{\Delta t} \quad (5.11)$$

where $\Delta Position$ is the change in the required spring plate position in meters and Δt is the time difference between the two position values in seconds.

The acceleration in m/s^2 was similarly calculated by:

$$Acceleration = \frac{\Delta Velocity}{\Delta t} \quad (5.12)$$

where $\Delta Velocity$ is the change in the required spring plate velocity in m/s and Δt is the time difference between the two velocity values in seconds.

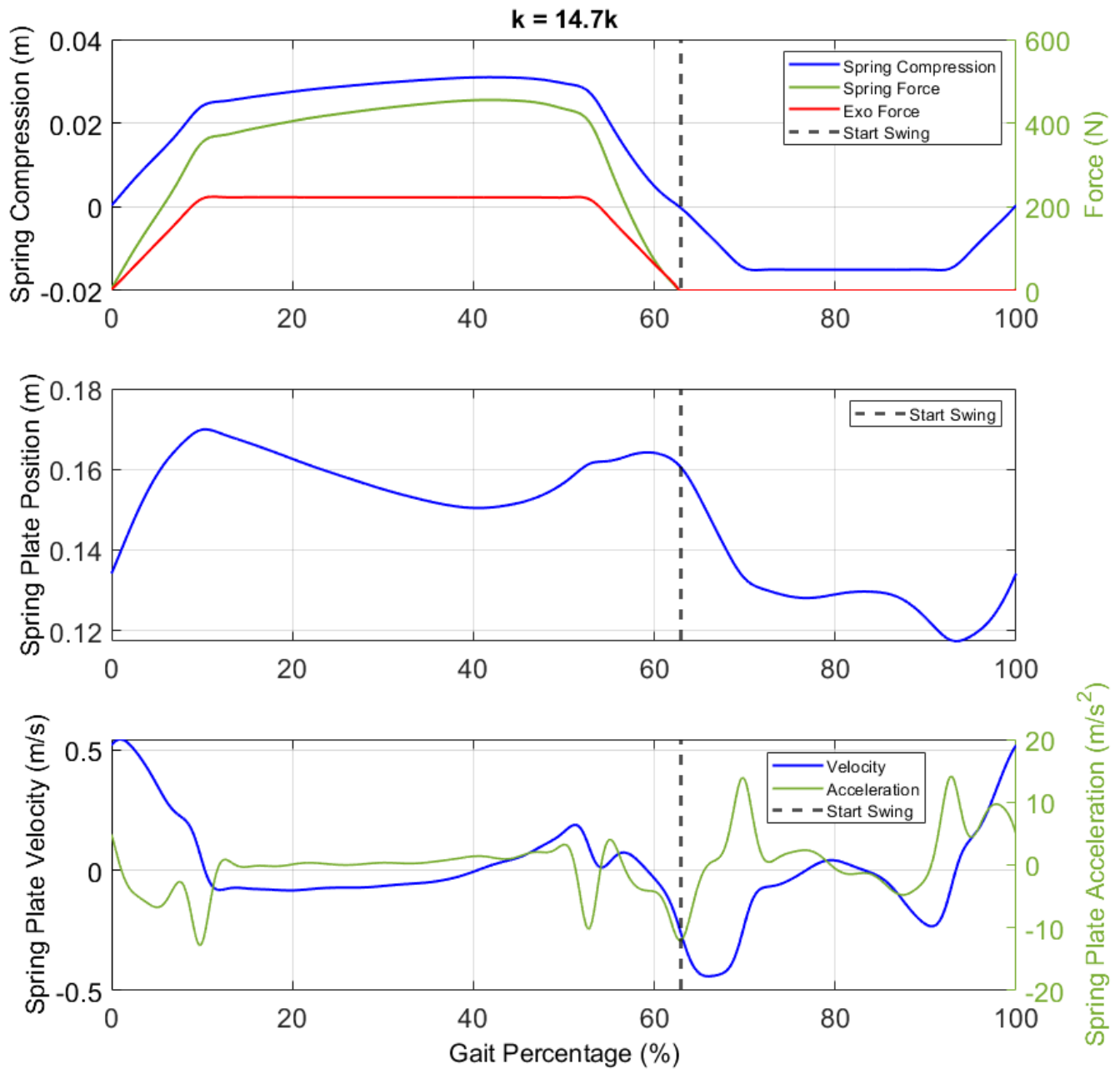


Figure 5.13: Required actuator and motor properties to maintain 222.4 N (50 lbf) at waist

Through the ODrive, velocity and acceleration limits were set experimentally for each motor to prevent them from drawing too much current and triggering an over current error. The spring plate velocity stays within the set velocity limit bounds of $\pm 0.64 \text{ m/s}$ for each leg. While the

acceleration exceeds the limit of $\pm 3 \text{ m/s}^2$ for the left leg and $\pm 4 \text{ m/s}^2$ for the right leg, this largely occurs when the spring plate makes larger movements to compress or decompress the spring. During Swing, the offset of the spring plate from the spring could be adjusted to minimize the required acceleration by turning off the distance tracking and maintaining a set value that is out of the way, or by decreasing the offset before heel strike which could also reduce the initial spike at the start of Stance.

5.4 Spring Stiffness Optimization

To optimize the spring stiffness based on the power required from the motor, it was first necessary to calculate the major inertia effects acting on the motor. These included the motor inertia (J_{motor}) in $\text{kg} * \text{m}^2$, and the ball screw inertia (J_{screw}) in $\text{kg} * \text{m}^2$.

$$J_{motor} = \frac{1}{2} * m_{motor} * r_{motor}^2 \quad (5.13)$$

where m_{motor} is the mass of the motor rotor in kg and r_{motor} is the radius of the motor in meters.

$$J_{screw} = \frac{1}{2} * \rho_{screw} * r_{screw}^4 * L_{screw} * \pi \quad (5.14)$$

where ρ_{screw} is the density of the ball screw kg/m^3 , r_{screw} is the radius of the ball screw in meters, and L_{screw} is the length of the ball screw in meters.

Next the required motor torque was calculated using the following equations where positive torque is defined as torque acting to move the spring plate towards the spring and negative torque is acting to move the spring plate away from the spring:

$$Torque_{motor} = Torque_{total} - Torque_{load} \quad (5.15)$$

$$Torque_{total} = \sum J * \frac{Acceleration}{lead} * 2\pi \quad (5.16)$$

$$Torque_{load} = \frac{F_{actuator} * lead}{2\pi * \%eff} \quad (5.17)$$

where $Torque_{motor}$ is the required torque of the motor in N*m, $Torque_{total}$ is the sum of torques at the motor in N*m, $Torque_{load}$ is the torque caused by the spring force transferred through the ball screw in N*m, $\sum J$ is the sum of the previously calculated inertias in $kg * m^2$, $Acceleration$ is the acceleration of the spring plate in m/s^2 , $F_{actuator}$ is the spring force in N, $lead$ is the lead of the ball screw in m/rev, and $\%eff$ is the ball screw efficiency taken to be approximately 80% to be conservative. [48]

Finally, the power required from the motor was calculated by the following equation and all of the calculations were performed iteratively for all spring stiffnesses of springs of the correct inner and outer diameter available from McMaster Carr. A plot of the required power through the gait cycle for different spring stiffnesses is shown in Figure 5.14.

$$Power_{motor} = Torque_{motor} * \frac{Velocity}{lead} * \frac{60}{9.5488} \quad (5.18)$$

where $Power_{motor}$ is the power required from the motor in Watts, $Torque_{motor}$ is the calculated motor torque in N*m, $Velocity$ is the required velocity of the spring plate in m/s , and $lead$ is the lead of the ball screw in m/rev. [49]

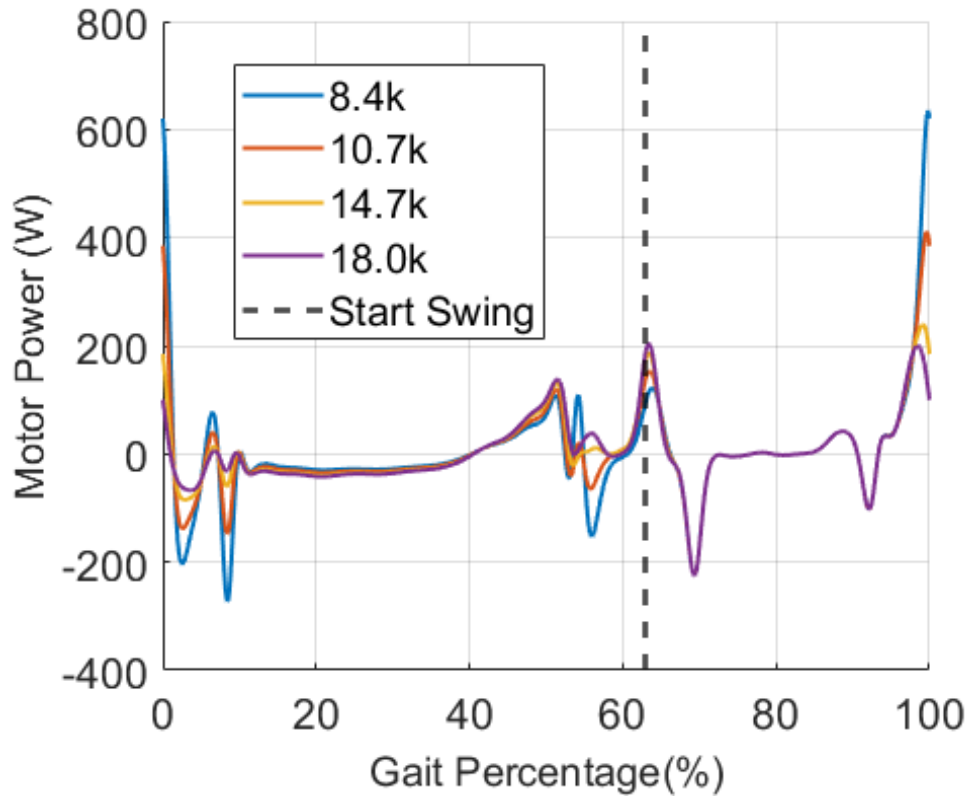


Figure 5.14: Required motor power with different spring stiffness values

While all of the required power values fall within the motor's limit of 1350 Watts, the plot shows that the spring stiffness doesn't always have the same relationship with the magnitude of the power at a given location. Greater magnitudes of power tend to occur when the spring stiffness is lower, but this can be examined more clearly in Figure 5.15 which plots the peak absolute power within the gait cycle for a large range of spring stiffnesses.

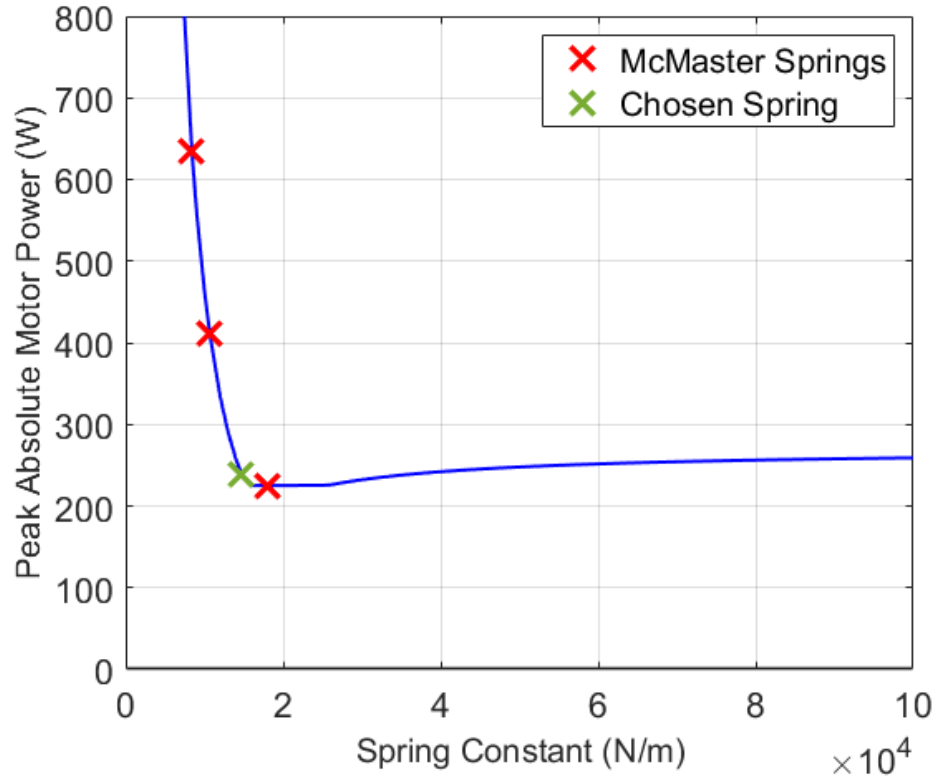


Figure 5.15: Peak absolute value of the required motor power resulting from different spring constants. Potential springs of the correct dimensions that are available from McMaster Carr are shown with Xs

From this plot it can be seen that for this exoskeleton design, there is an optimal spring constant near 16,000 N/m that will require the smallest peak power of 225.18 W. Spring constants decreasing from this value quickly require much larger amounts of power while spring constants increasing from this value also require more, but level out around 259 W. Extending beyond the plot boundaries, the required power increases to around 269 W at 1,000,000 N/m. The current spring used in the exoskeleton is the 14,700 N/m spring, requiring 239.27 W, meaning the efficiency of the exoskeleton could be improved by installing the 18,038 N/m spring, requiring 225.21 W. A similar optimization trend of peak absolute power with varying spring constants was also found by Hollander et al. [50] in their evaluation of a robotic tendon model. A flowchart detailing the code and calculations from Chapter 5.3 and 5.4 can be seen in Appendix D.

Chapter 6: Testing/Results

6.1 Range of Motion

To examine the effect of the exoskeleton on the range of motion of the user, measurements were taken for the user's right leg with and without the exoskeleton. These measurements were also compared with previously recorded range of motion data of the exoskeleton with no user. All range of motion measurements can be seen below in Table 6.1. Two primary methods were used for measuring the range of motion of the user. Method one, used for hip extension/flexion and hip adduction/abduction, involved standing next to a wall and marking the point on the wall in line with the hip joint, as well as the vertical line adjacent to the leg. Depending on the desired measurement, the user would then extend their leg forward, back, out, or in (across their other leg), multiple times to the full extent they were able to. A second person would measure the furthest reach of the user's foot and mark that point on the wall. After accounting for the offset between the edge of the user's foot and the point in line with the center of the leg, a line was marked from that point to the marked hip location and a goniometer was used to measure the angle between this line and the vertical leg line. A diagram of this method can be seen below in Figure 6.1 for hip adduction.

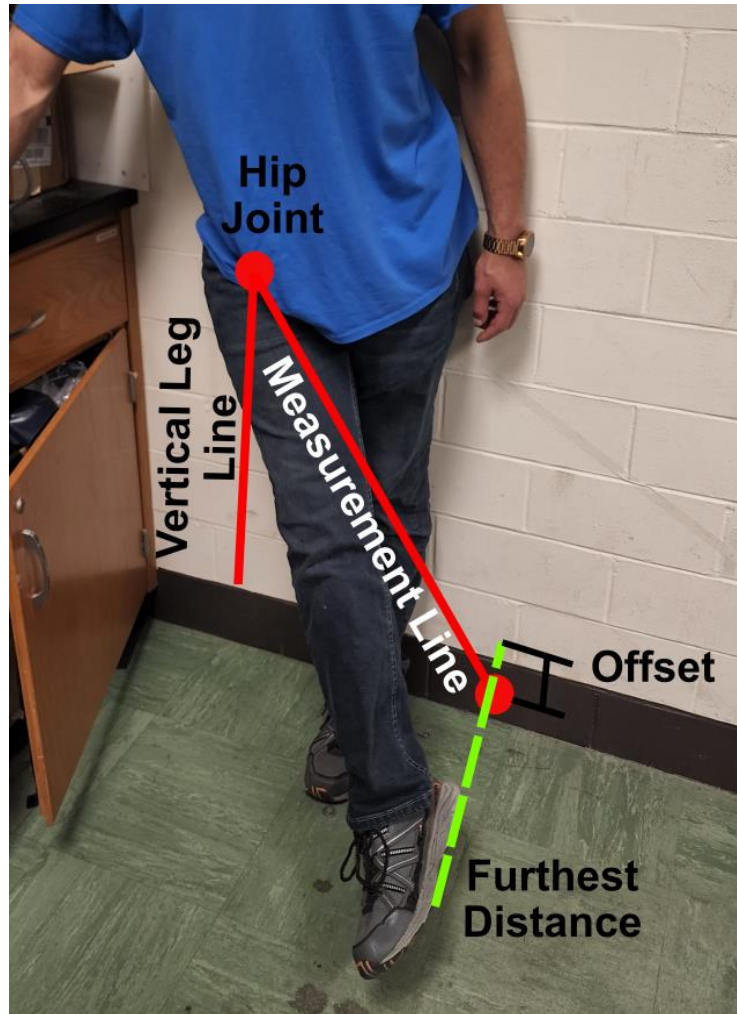


Figure 6.1: Range of Motion measurements, method one

The second method of range of motion measurements, used for all other measurements, was to support the desired joint for measurement when possible. This was done by bracing against a surface or holding the leg. A second person would then take a direct measurement with the goniometer. Supporting the joint is necessary to minimize the effect of the flexibility of the user from skewing the range of motion data. Examples of this method can be seen in Figure 6.2, Figure 6.3, and Figure 6.4 below.



Figure 6.2: Range of Motion measurements, method two (Knee Flexion)



Figure 6.3: Range of Motion measurements, method two (Ankle Dorsiflexion)



Figure 6.4: Range of Motion measurements, method two (Ankle Inversion)

Table 6.1: Range of Motion Results. The Exo Range of Motion is the angle achievable by the exoskeleton linkages, which are not exactly aligned with the user's joints.

Motion	Exo Range of Motion (Degrees)	User Range of Motion (Degrees)	Exo & User Range of Motion (Degrees)
Hip Extension/Flexion	(-180) - 180	(-47) - 91.5	(-45) - 70
Hip Adduction/Abduction	(-17.5) - 14	(-29.5) - 59.5	(-29) - 25.5
Hip Internal/External	(-17.5) - 14	(-84.5) - 59.5	(-14) - 35
Knee Extension/Flexion	44 - 237.5	40 - 189.5	40 - 189.5
Ankle Dorsiflexion/Plantarflexion	60 - 148	73 - 126	73 - 126
Ankle Eversion/Inversion	(-31) - 37.5	(-31) - 33	(-31) - 31

While the ultimate goal was to avoid the exoskeleton from limiting any of the range of motion of the user, that goal was not able to be fulfilled while minimizing weight, cost, and complexity of the frame design. At the hip, every direction of motion experienced some limitation from the exoskeleton. Hip extension and adduction both had minor limitations, less than 2 degrees, while the other ranges of motion at the hip experienced limitations greater than 21.5 degrees. Hip internal rotation experienced the greatest limitation of 70.5 degrees due to the placement of the rigid exoskeleton frame on the outer side of the leg. At the knee and ankle joint, only ankle inversion experienced any limitation. The limitation experienced was only 2 degrees. Every other range of motion was unaffected by the exoskeleton. While it may appear strange that the limitation of the exoskeleton range of motion is sometimes greater than the limitation found with the user in the exoskeleton, this can be accounted for by the fact that the exoskeleton specific measurements were taken just of the joint and the measurements of the user in the exoskeleton could include additional give due to softgoods and interfaces between the exoskeleton and the user.

6.2 Treadmill Walking Tests

In order to test the viability of the exoskeleton, walking trials were performed on a treadmill at speeds of 0.45 m/s, 0.67 m/s, 0.89 m/s, and 1.12 m/s (1.0 mph, 1.5 mph, 2.0 mph, and 2.5 mph) using a supplied spring force of 323.6 N (72.75 lbf) which equates to around 161.8 N (36.38 lbf) of force at the waist. During each trial, multiple gait cycles worth of data were recorded. The recorded data included time of measurements, distance from the ground at the heel and ball for each foot, force at the heel and ball for each foot, joint angles of the knee and ankle for each leg, the requested spring compression of each actuator, the requested motor position of each actuator,

and the actual motor position of each actuator. To perform the test, the user would stand on the treadmill while a second person (tester) would switch on the exoskeleton and monitor the data collection to ensure all sensors were working properly. Once the exoskeleton successfully completed the calibration sequence and all sensors were working, the tester would start the treadmill and slowly increase the speed to the desired point. Once at the desired speed with the user walking steadily, the tester would clear the data collected to that point so the data set would start at the desired walking speed. The tester remained prepared to slow the speed or press the emergency stop button if the user fell or the exoskeleton failed. Once multiple gait cycles worth of data had been recorded, the tester slowed the treadmill to a stop, turned off the exoskeleton, and saved the data. An image of the treadmill used can be seen in Figure 6.5.



Figure 6.5: Human subject walking trials on treadmill

Along with plotting the data for visual inspection, the average sample period and the heel strike detection delay were calculated at each speed. The sample period was found by taking the difference between each time during the chosen time window and finding the average. The heel strike detection delay required determining the approximate point of heel strike. This was done by using a slope threshold to find the points in the heel force data where the force just starts to spike. Since it is known that the point that compression of the spring is requested is the point that heel

strike was detected by the code, finding the average time difference between the approximate heel strike and the detected heel strike will provide the detection delay. These values were always found for the right leg data and the approximate heel strike for the right leg is plotted on all graphs to show the separation of different gait cycles. Figure 6.6, below, shows the approximate heel strike and the predicted heel strike marked on the right leg force data for the 1.12 m/s (2.5 mph) trial.

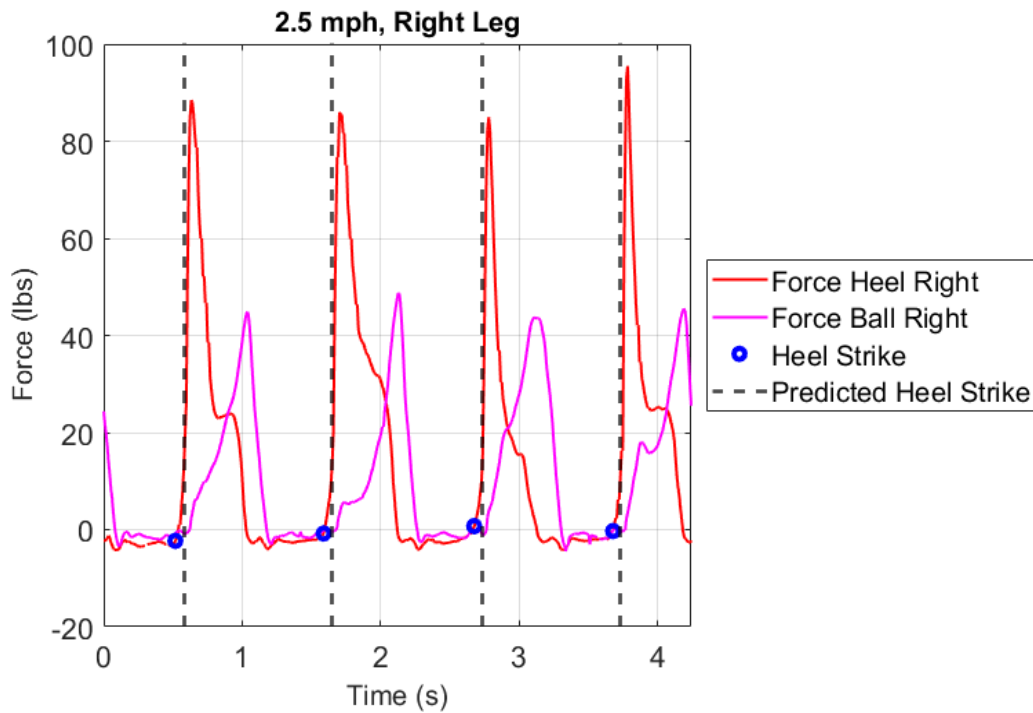


Figure 6.6: 1.12 m/s (2.5 mph) experimental force data with estimated and predicted heel strike

Observing the joint angle data for the 1.12 m/s (2.5 mph) trial, seen in Figure 6.7, the periodic pattern between gait cycles can be seen. Looking at the right leg knee data, the pattern of peak, small trough, peak, larger trough can be seen as the leg moves from being close to straight at heel strike, bends during stance, becomes almost straight right before Swing, and bends again during Swing. The recorded angles never near 180 degrees because there is a pre-bend in the exoskeleton knee joint caused by the give in the softgoods. This pre-bend reduces the risk of the exoskeleton

knee inverting due to exceeding 180 degrees. When looking at the right ankle data, the angle drops soon after heel strike when the user's center of mass is transferring over the leg before Swing Phase. After Swing Phase is entered, the ankle angle increases again. The ankle data varies much less than the knee angle and while some of this can be attributed to the different joint locations and functions, it is also believed that the additional weight and resistance provided by the exoskeleton could subconsciously cause the user to reduce their ankle motion and maintain a more neutral position.

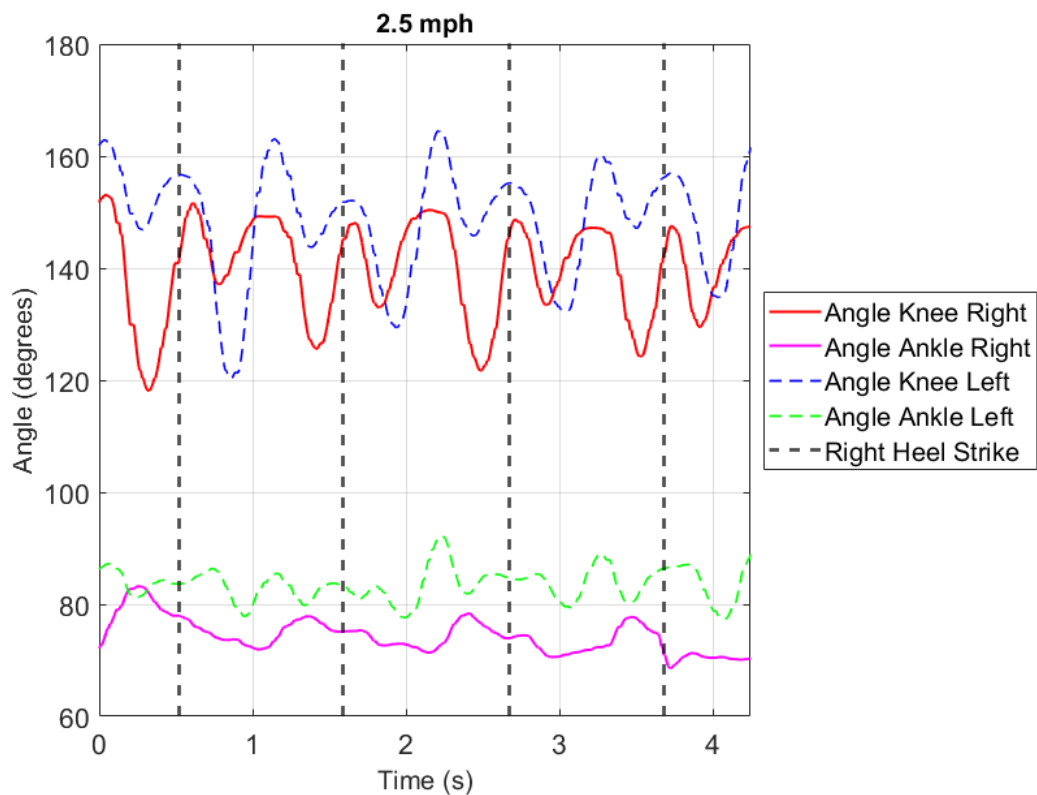


Figure 6.7: 1.12 m/s (2.5 mph) experimental joint angle data

Figure 6.8 shows a plot of the actual and commanded motor position as well as a plot of the force data for the right leg. In the motor position plot, it can be seen that there is a delay between when the commanded position changes and when the actual position catches up to it. This is limited by

the motor acceleration and deceleration set on the ODrive to prevent the motors from exceeding the current limit. This delay plus the detection delay for detecting heel strike is the amount of time between when the approximate heel strike occurs and the appropriate assistive force is applied.

Looking at the force data, there is a very consistent trend of heel force spiking then transferring to the toe during stance phase. When summed together, the total force profile created by the heel and toe forces matches the trend of the GRF during human gait.

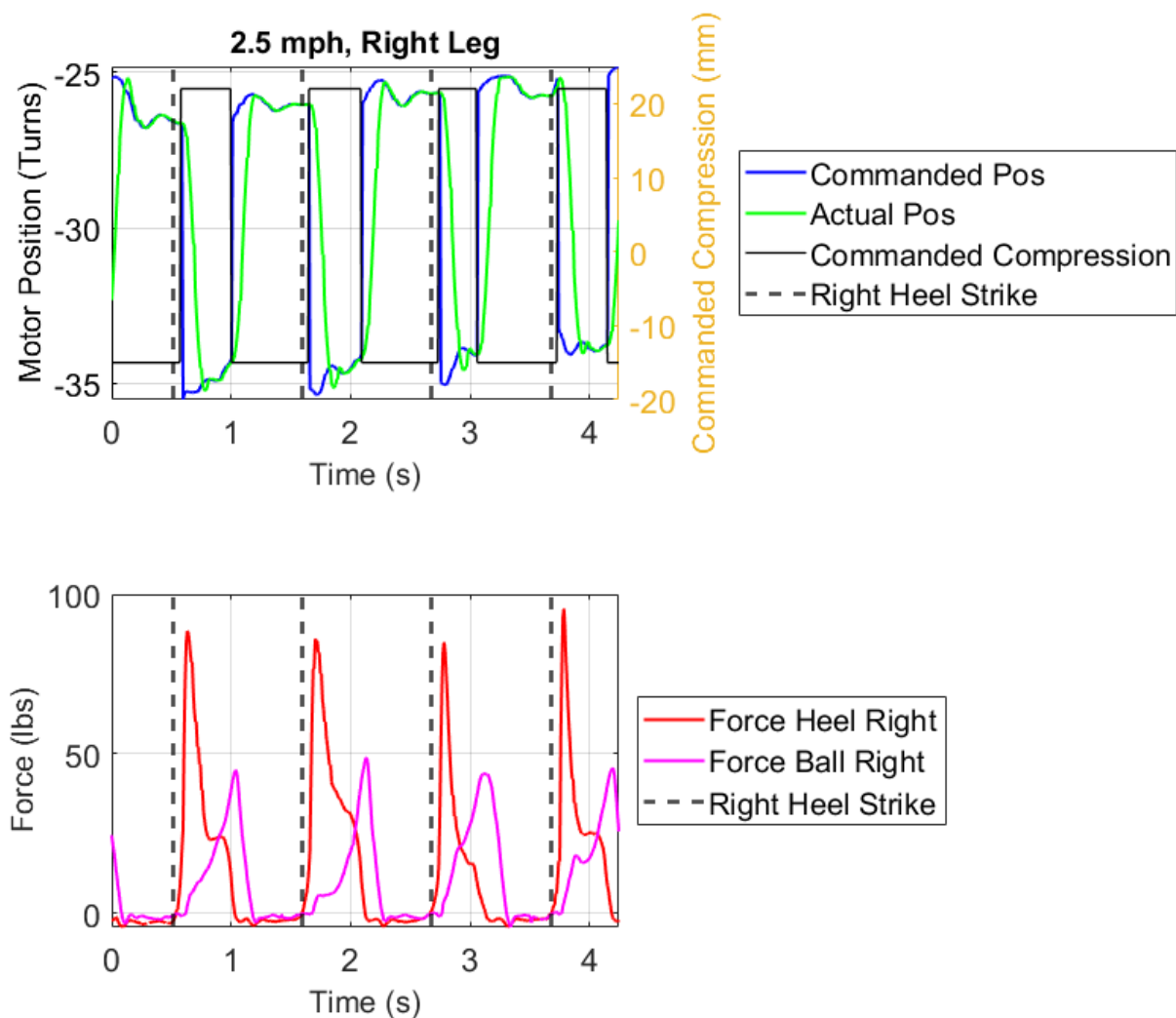


Figure 6.8: 1.12 m/s (2.5 mph) experimental actuator data and force data for right leg

Figure 6.9 shows the same data as Figure 6.8, but for the left leg. The same trends can be seen in this leg, but based on the marked right heel strike, the consistent offset can be seen between the data sets. This constant offset can be seen even better in Figure 6.10 which just shows the actual motor position of each actuator and the approximate right heel strike. With some overlap on either side, the back-and-forth transfer of which leg is providing assistance (troughs of motor position plots) can be seen.

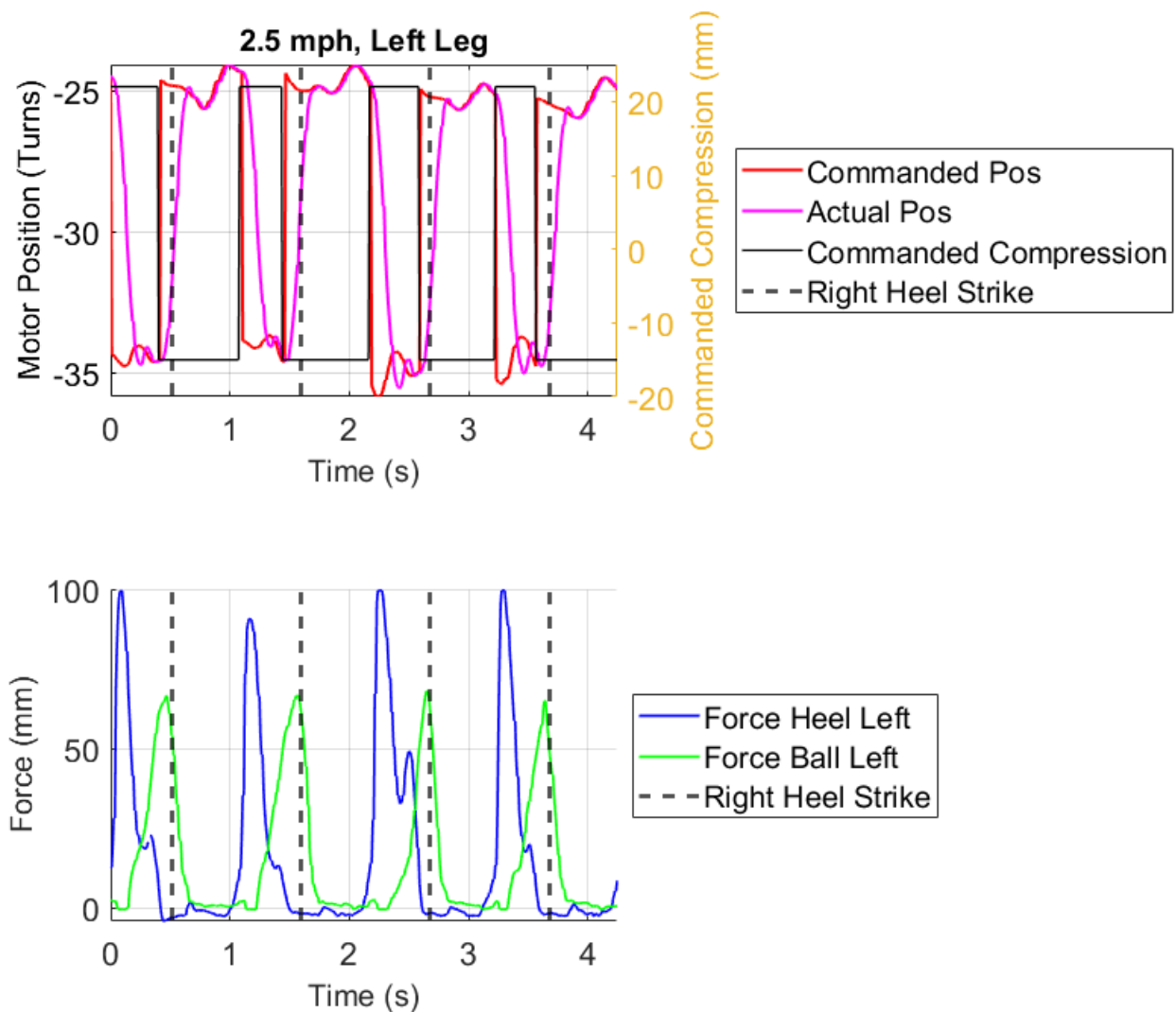


Figure 6.9: 1.12 m/s (2.5 mph) experimental actuator data and force data for left leg

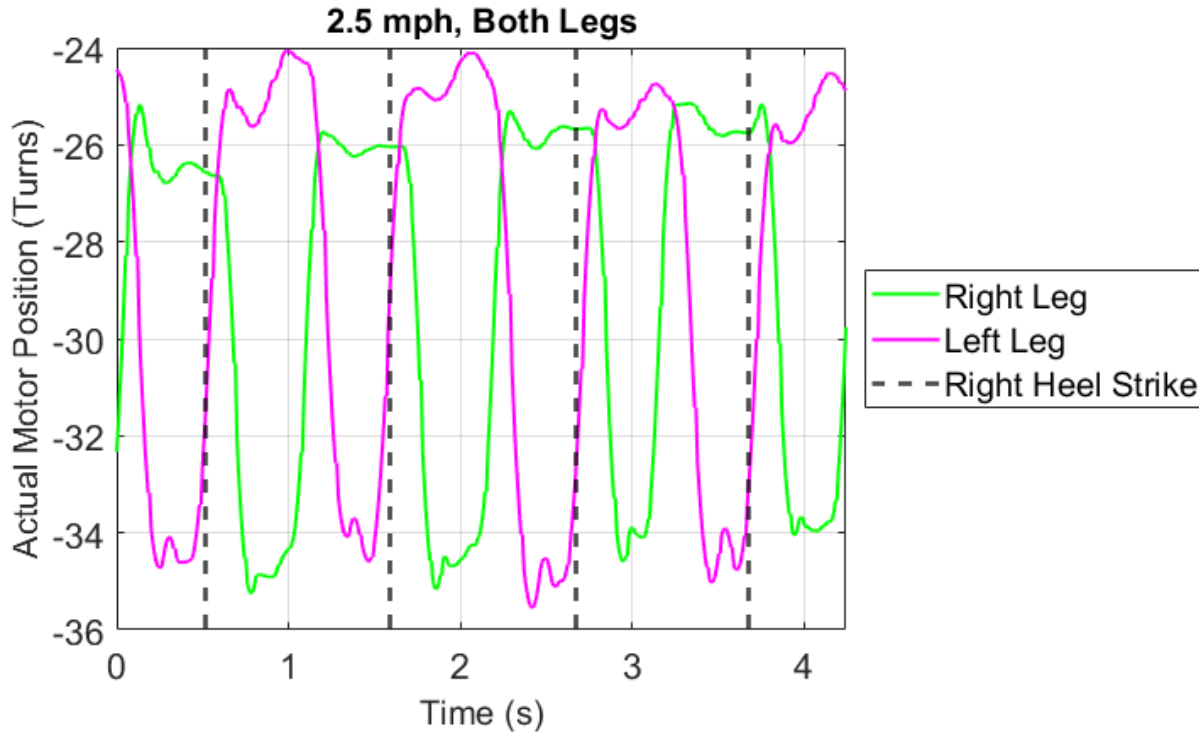


Figure 6.10: 1.12 m/s (2.5 mph) experimental motor position data for both legs

It was found that at 1.12 m/s (2.5 mph) walking, the distance sensors occasionally stopped working. For that reason, Figure 6.11 and Figure 6.12 show distance data for the right and left leg respectively from the 0.89 m/s (2.0 mph) walking trial. While the different trial speed does not allow for direct comparison with the 1.12 m/s (2.5 mph) data, it does show the general distance trends and give a view of the quality of data available from the distance sensors. The breaks seen in these plots are due to the distance sensors reading out of range. Once above approximately 100 mm, the distance sensors will return an error value; these points were removed from the data set. Looking at the right leg data, the heel distance can be seen to drop low at heel strike, quickly followed by the toe at foot flat. The heel then increases out of range while the toe remains lower, but occasionally exceeds the range as well during Swing Phase. Once again, the left leg data follows the same trends, but shifted in time.

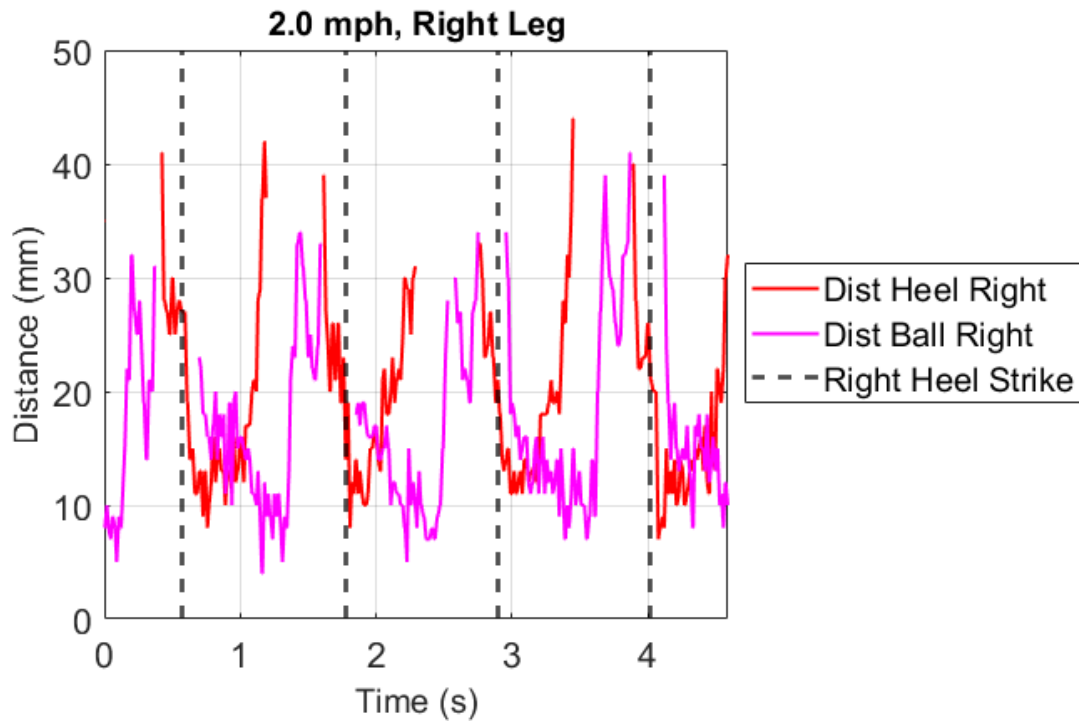


Figure 6.11: 0.89 m/s (2.0 mph) experimental distance data for right leg

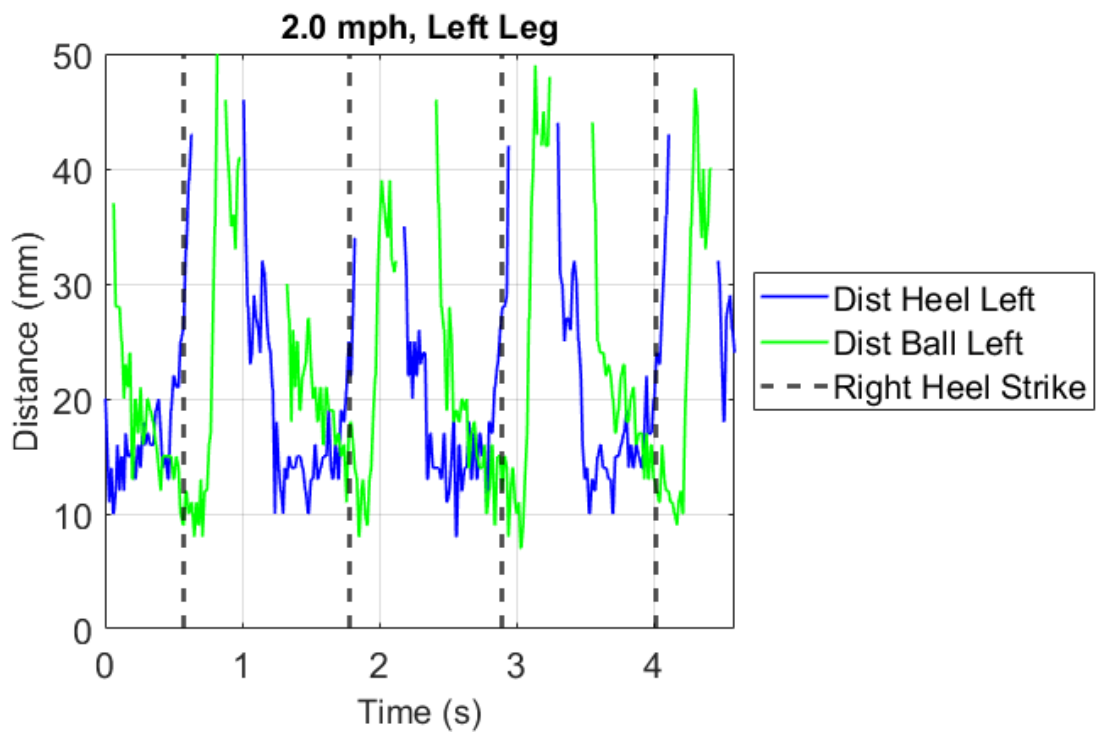


Figure 6.12: 0.89 m/s (2.0 mph) experimental distance data for left leg

Unlike the force sensor data, the data provided by the distance sensors contained a lot of noise. This largely prevented the successful use of simple codes relying only on the distance sensor data. An attempt to apply a filter to the data did show a great decrease in the noise, but not enough to completely remedy the issue. Additionally, it was found that reading the distance sensors significantly lowered the sampling rate achievable by our microcontroller. In Figure 6.13 below, right leg force data at 0.45 m/s (1.0 mph) was taken while reading the force and distance sensors, and while only reading the force sensors. The decreased discretization and enhanced ability to pick out significant small details visually shows the difference in the sampling rates achieved which was 52.15 Hz with the distance sensors and 116.25 without the distance sensors. As a result of these findings, all data sets not specifically looking at distance data were taken without reading the distance sensors.

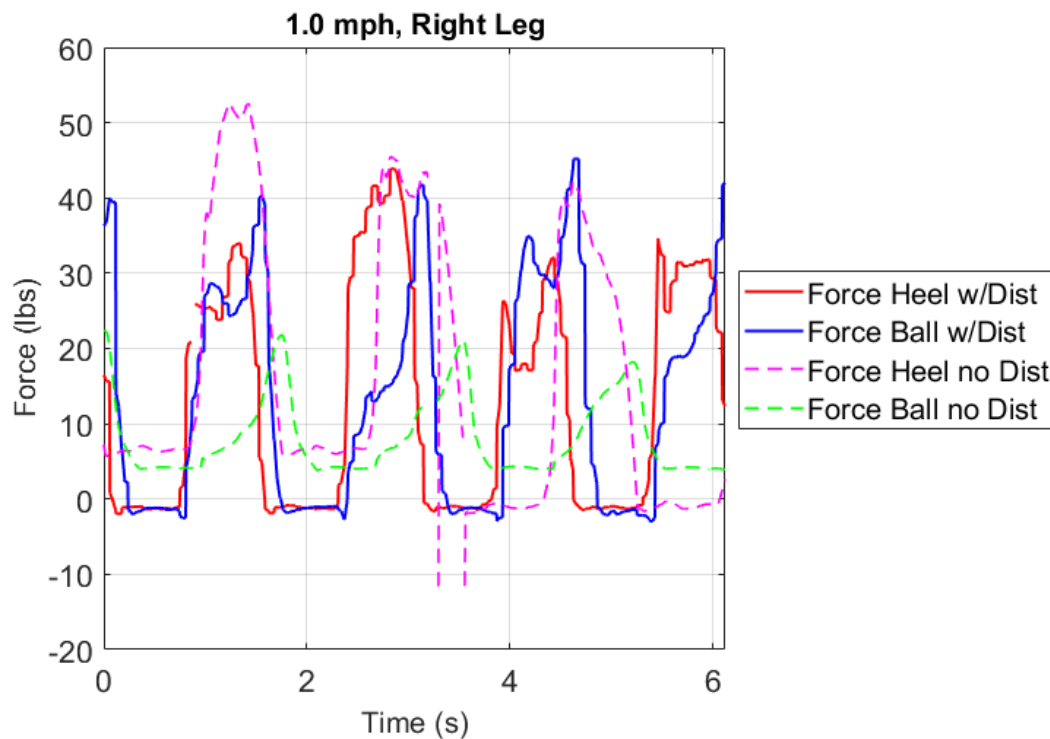


Figure 6.13: 0.45 m/s (1.0 mph) experimental force data for right leg comparison when distance sensors are and are not being read

When comparing the effect of walking speed on the quality of the data and the ability of the code, Figure 6.14 shows that the sampling period (when distance sensors are not read) remains constant around 8.7 ms. The detection delay does not have a definitive trend that can be seen based on the tested walking speed, but the detection delay remains between 59 ms and 69 ms through the recorded trials. From the qualitative experience of the user, greater discomfort and feelings of decreased stability and control of motion while walking with the exoskeleton at 1.12 m/s (2.5 mph) were reported.

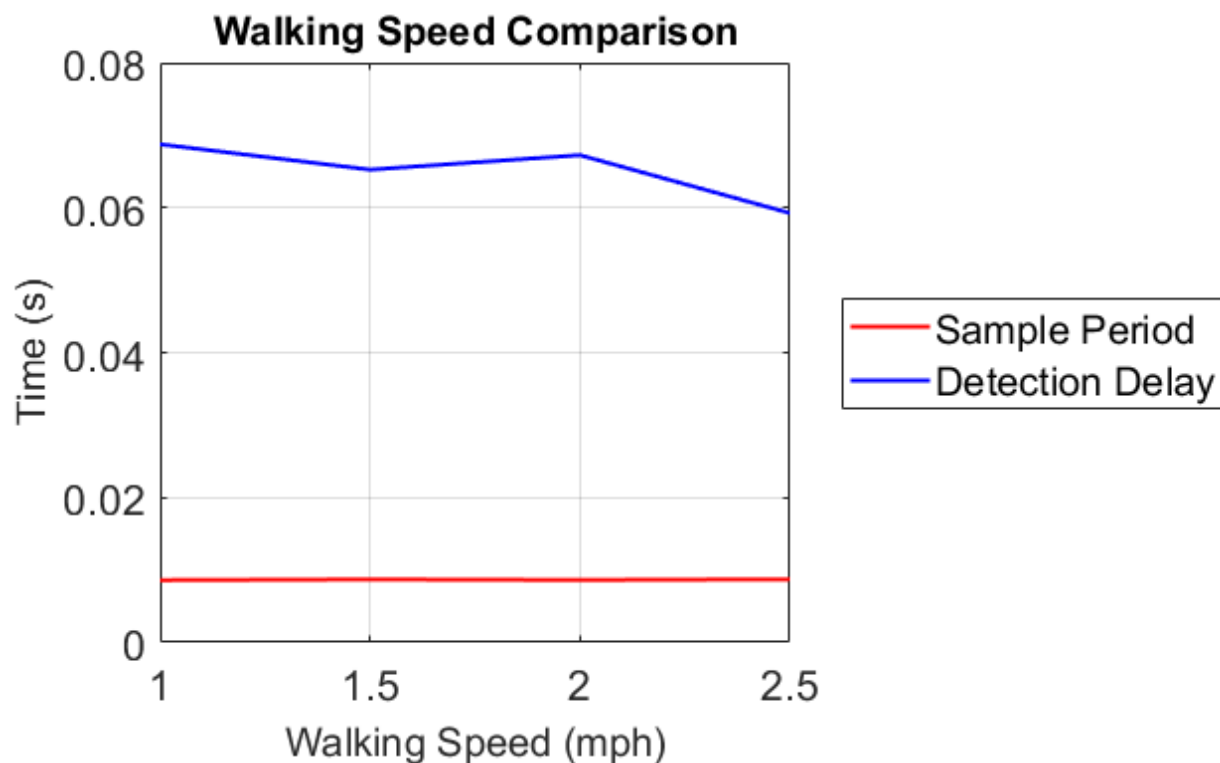


Figure 6.14: Sample period and detection delay time comparison for multiple walking speeds

6.3 Model Comparison

In Chapter 5.3, biological joint angles and limb segment lengths from another study were used to simulate the response of the exoskeleton if it were to exactly follow the same joint angles. In reality, the exoskeleton does not perfectly follow the user's body due to non-rigid softgood attachment points to the user. Figure 6.15 shows the theoretical joint angles and calculated actuator and GRF angles used in the initial calculations alongside the joint angle data taken from the exoskeleton during the 1.12 m/s (2.5 mph) trial and used to calculate the actuator and GRF angles.

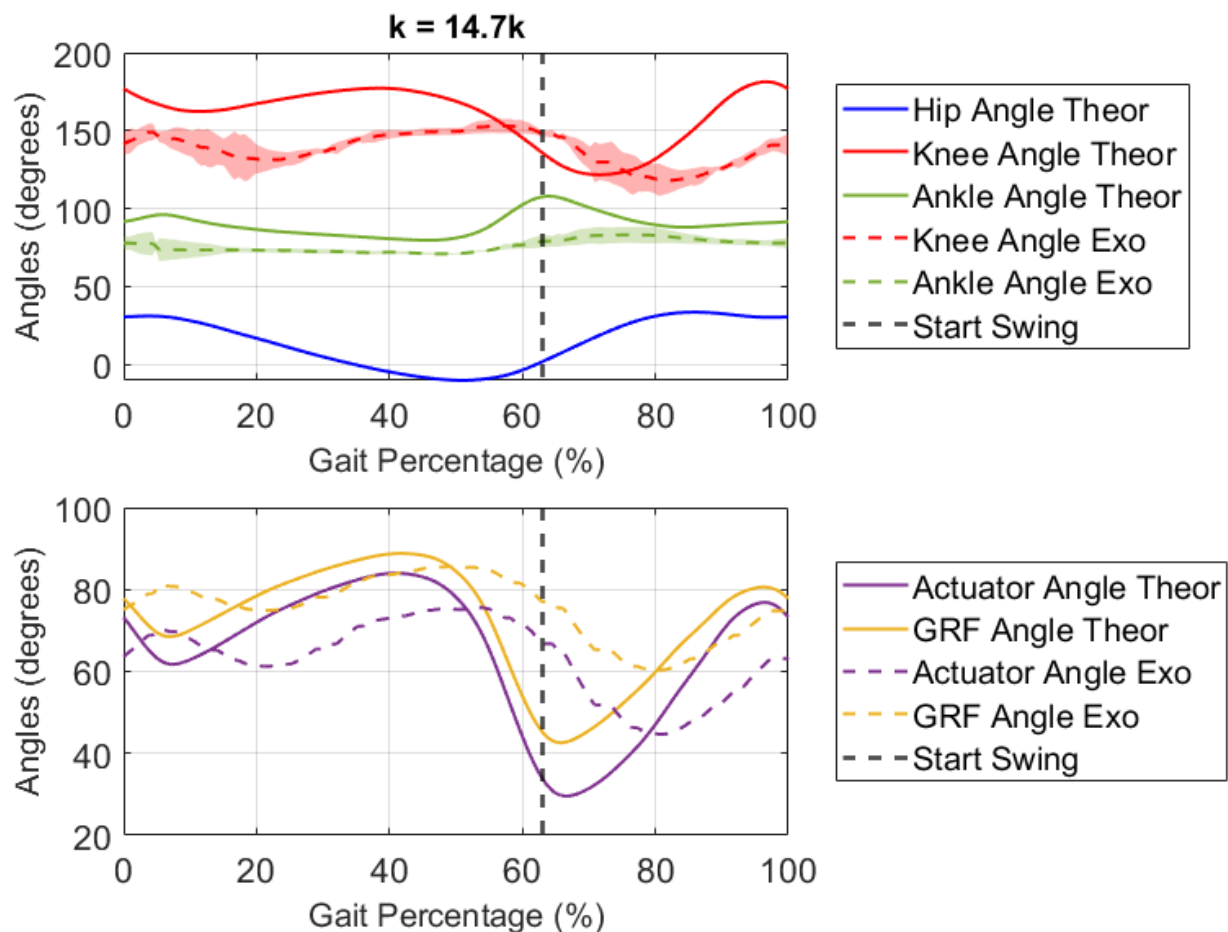


Figure 6.15: Biological joint angles and calculated actuator and GRF angles compared to joint angles collected from exoskeleton and resulting actuator and GRF angles. The shaded fields around the knee and ankle exoskeleton angles show the variance of the data over 7 gait cycles.

To successfully match the experimental data with the theoretical data, 7 gait cycles of data starting from one detected heel strike until the next, were resampled to ensure matching data length, and examined. Out of the 7 cycles, one was chosen that had consistent data and fell around the median of the cycles visually. Next, to match the experimental data with the model, the detection delay of 0.0593 seconds that was calculated for the 1.12 m/s (2.5 mph) trial from Chapter 6.2 was used to appropriately shift the experimental data. It can generally be seen that the trend in the experimental and theoretical angles largely match while being shifted in the gait cycle, but the experimental data has a smaller variation from its mean. This is likely a result of the weight of the exoskeleton and the increased resistance of the joints influencing less joint movement. This is also reflected in the length of the actuator and the leg shown in Figure 6.16.

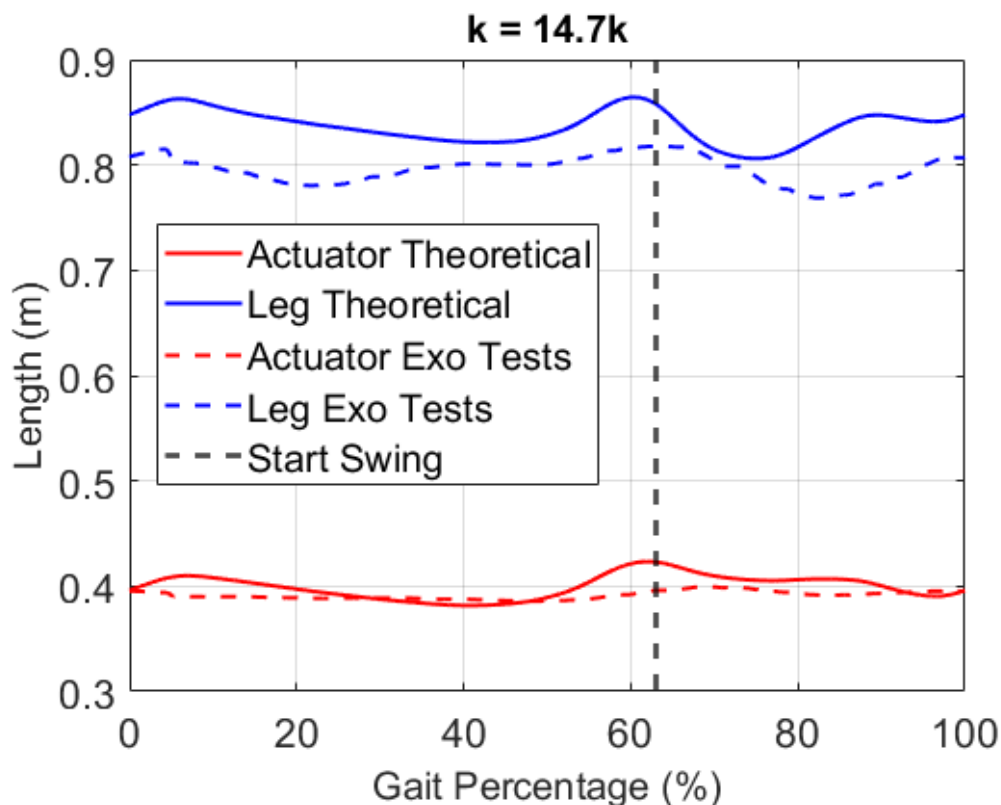


Figure 6.16: Actuator and leg length based on theoretical joint angles and joint angles collected from exoskeleton

Due to its relation to the amount of force that needs to be applied from the actuator in order to feel the desired force at the waist, the support ratio calculation based on the actual exoskeleton joint angles was of particular interest. The result can be seen in Figure 6.17.

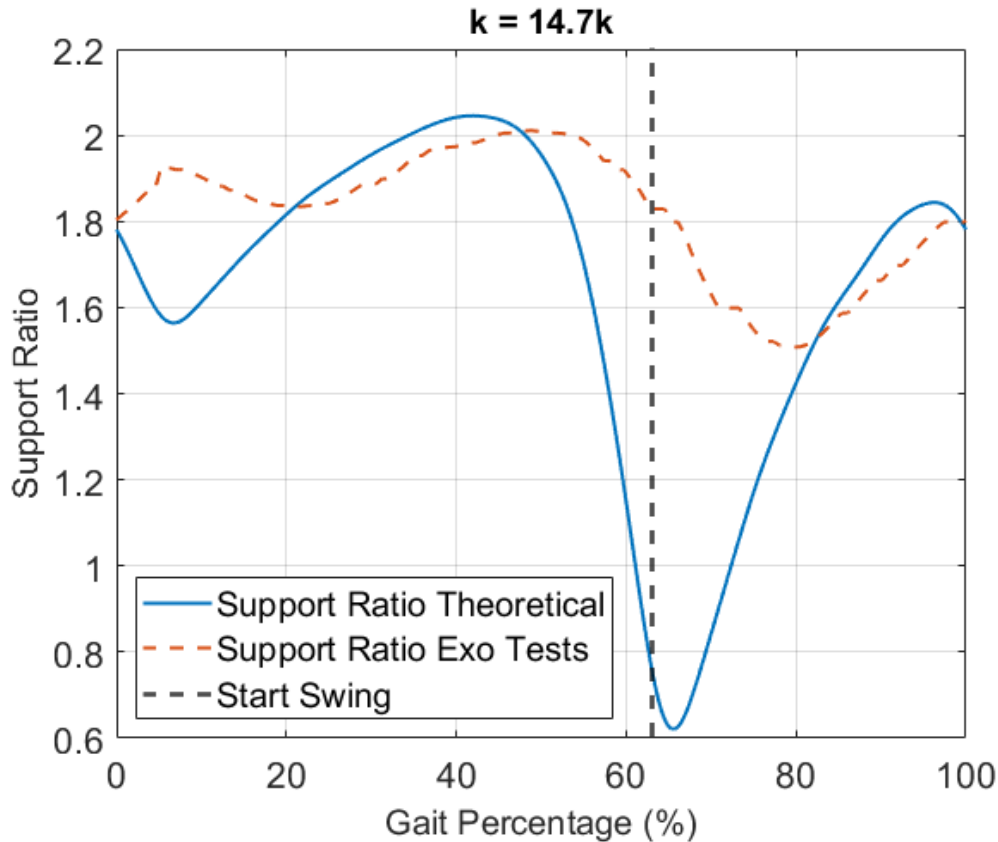


Figure 6.17: Required support ratio to maintain 222.4 N (50 lbf) at waist based on theoretical joint values and joint values collected from exoskeleton

Again, much less variation is seen in the experimental support ratio than the theoretical support ratio. The support ratio of the exoskeleton stays bounded between 1.5 and 2.1, meaning the current method of applying a constant support force assuming a support ratio of 2 is sufficient for the current testing. Figure 6.18 below shows the spring compression, actuator spring force, and exoskeleton force felt by the user for the theoretical model from Chapter 5.3, the values commanded by the software, and the actual values achieved by the exoskeleton.

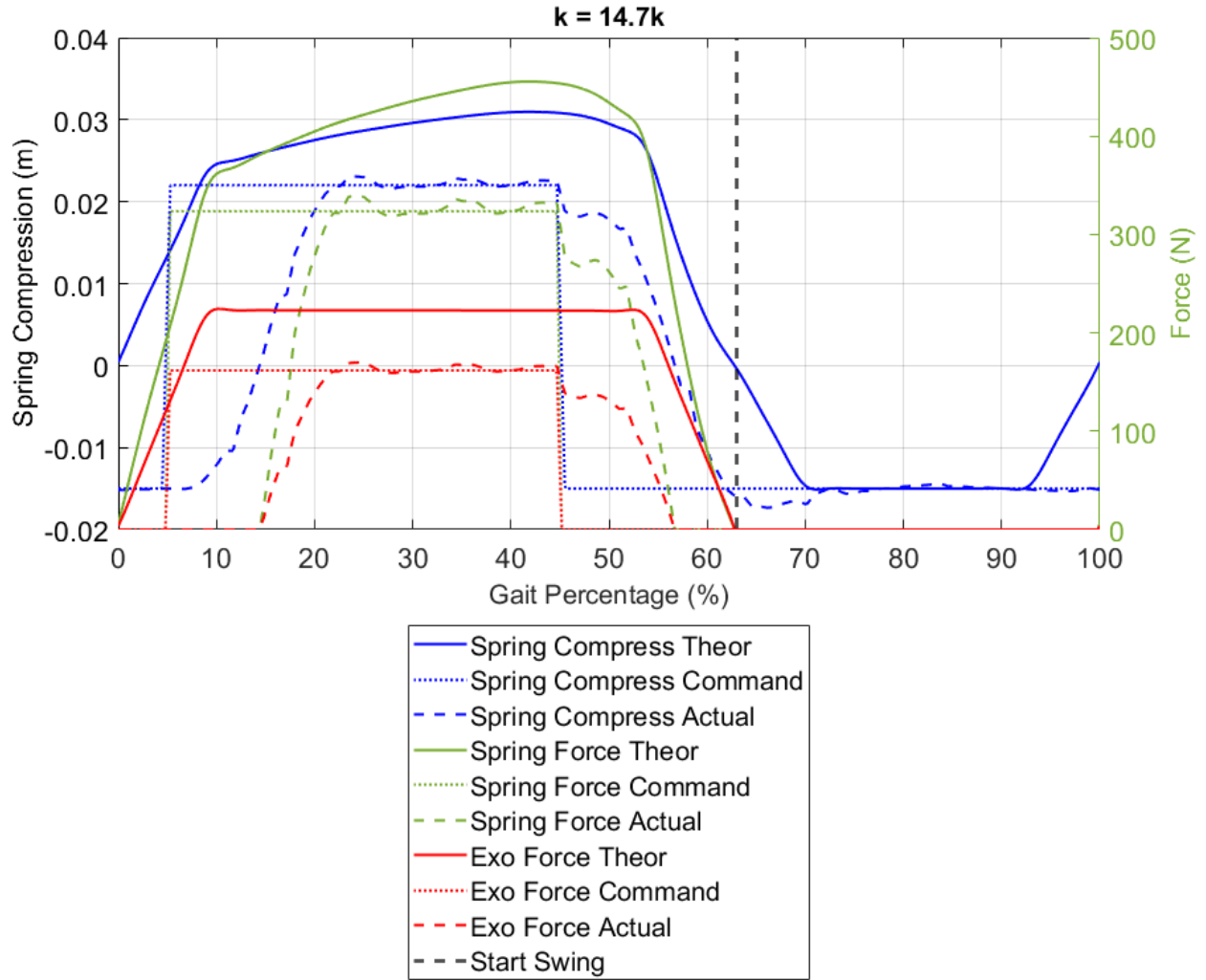


Figure 6.18: Exoskeleton force comparison showing spring compression, spring force from the actuator, and force felt by the user from the exoskeleton for the theoretical model, commanded exoskeleton values, and actual exoskeleton values

The plot shows that the exoskeleton can maintain the commanded compression but requires approximately 15% of the gait cycle to move to or from the uncompressed 15 mm away from the spring position and the 22 mm desired compression. The delay in detection of stance and the early detection of swing phase can also be seen in the sudden rise and fall of the commanded values and greatly affect the application of the force supplied by the exoskeleton.

Chapter 7: Conclusion

Throughout this thesis, a novel exoskeleton design intended for light rehabilitation has been presented and its performance evaluated. This work contributes to the advancement of exoskeleton research by examining a new underactuated design that uses one motor per leg and makes use of a pantograph to reduce the overall profile and protrusion of the exoskeleton from the user's body. The exoskeleton operates by compressing a spring in parallel with the user's leg during Stance Phase to provide an assistive force between the ball of the user's foot and their waist. The exoskeleton was designed for a support force up to 222.4 N (50 lbf), but was tested up to approximately 161.8 N (36.38 lbf) in human subject experiments using a simple gait detection method based on GRF at walking speeds from 0.45 m/s to 1.12 m/s (1.0 mph to 2.5 mph). Range of motion testing was also performed to evaluate the limitations the exoskeleton places on the user's typical movement.

Calculations were presented examining the effect of exoskeleton assistance on the biological joint moments and optimizing the actuator design to reduce power consumption. Finally, the actual performance of the exoskeleton was compared with predictions based on the joint angles during a typical walking cycle.

7.1 Future Work

Moving forward, there are multiple routes of improvement that could improve the function of the exoskeleton, the largest of which being to implement a more complex gait detection method. By

making use of the GRF data and distance data available from each leg, it should be possible to create an algorithm that can successfully predict heel strike in advance and eliminate or greatly reduce the delay of support force during Stance Phase. The greatest obstacle to this advancement is the reliability of the sensors in the foot attachment. It is believed that the inconsistent operation of the sensors is a result of the varying loads on the sensors and wires in the foot attachment and the pliable material that make up the foot attachment. A re-design of the foot attachment would likely be necessary to fully take advantage of a more complex gait detection method.

An additional software advancement would be the inclusion of algorithms that allow for walking on stairs and inclines and allow for sitting. The greatest challenge here would be determining how to detect when to switch between the methods without requiring user input.

There were two reasons the exoskeleton was only tested at 161.8 N (36.38 lbf) instead of the designed 222.4 N (50 lbf): the greater potential for exceeding the current limit, and the sizing of the actuator causing the compression point for full support to occasionally exceed the physical limit of the spring plate. Both issues could potentially be addressed by some re-designs to the actuator. Increasing the tolerancing on the actuator parts or changing the design to better direct the forces applied on the linear guide rails could reduce the current required by the motor to move the spring plate, allowing for larger and quicker movements. Lastly, increasing the length of the spring tube to move the spring further up into the actuator would allow for the additional space needed to achieve full support.

Appendix A: ODrive Initial Setup Values

ODrive Motor Controller Settings v0.5.1

odrv0.axis1.motor.config.current_lim = 55 [A]

odrv0.axis1.motor.config.current_lim_margin = 5 [A]

Note: Margin for current overshoot

odrv0.axis1.motor.config.requested_current_range = 120 [A]

odrv0.axis1.controller.config.vel_limit = 125 [turns/s]

Note: max possible is $(K_v * \text{Voltage}) / 60$

odrv0.axis1.motor.config.calibration_current = 16 [A]

odrv0.config.brake_resistance = 2 [Ohm]

Note: No brake resistor needed, used to account for resistance in board

odrv0.axis1.encoder.config.bandwidth = 1000

odrv0.config.enable_brake_resistor = False

odrv0.config.dc_max_negative_current = -55 [A]

Note: battery max -131 A

odrv0.axis1.motor.config.pole_pairs = 7

odrv0.axis1.motor.config.torque_constant = 8.27/500

Note: torque_constant = $8.27 / (\text{motor KV})$

odrv0.axis1.motor.config.motor_type = 0 = MOTOR_TYPE_HIGH_CURRENT

odrv0.axis1.encoder.config.cpr = 4000 [CPR]

odrv0.axis1.encoder.config.calib_scan_distance = 52

Note: raised ~ 2 over default. Can raise up to 4 above

odrv0.axis1.encoder.config.mode = 0 = ENCODER_MODE_INCREMENTAL

odrv0.axis1.encoder.config.calib_range = 0.0199999

odrv0.axis1.motor.config.resistance_calib_max_voltage = 2

odrv0.axis1.encoder.config.use_index = True

odrv0.axis1.requested_state = AXIS_STATE_FULL_CALIBRATION_SEQUENCE

```
odrv0.axis1.requested_state = AXIS_STATE_ENCODER_INDEX_SEARCH
odrv0.axis1.requested_state = AXIS_STATE_ENCODER_OFFSET_CALIBRATION
```

VERIFY:

```
odrv0.axis1.error = 0
odrv0.axis1.encoder.config.phase_offset = #
    Note: # should be value such as -326 or 1364
odrv0.axis1.encoder.config.direction = 1 or -1
```

```
odrv0.axis1.encoder.config.pre_calibrated = True
    Note: Check that this stays True after setting motor.config.pre-calibrated = True
odrv0.axis1.config.startup_encoder_index_search = True
odrv0.axis1.motor.config.pre_calibrated = True

odrv0.axis1.config.startup_motor_calibration = False
odrv0.axis1.config.startup_encoder_offset_calibration = False
odrv0.axis1.config.startup_closed_loop_control = True
```

TUNING Motor 1

```
odrv0.axis1.controller.config.pos_gain = 19 [(turns/s)/turns]
odrv0.axis1.controller.config.vel_gain = 0.135 [Nm/(turns/s)]
odrv0.axis1.controller.config.vel_integrator_gain = 8.4499998 [Nm/((turns/s)*s)]
    Note: vel_integrator_gain  $\approx 0.5 * 200 \text{ Hz} * \text{vel\_gain}$ 
```

TUNING Motor 0

```
odrv0.axis0.controller.config.pos_gain = 19 [(turns/s)/turns]
odrv0.axis0.controller.config.vel_gain = 0.0845 [Nm/(turns/s)]
odrv0.axis0.controller.config.vel_integrator_gain = 13.5000009 [Nm/((turns/s)*s)]
    Note: vel_integrator_gain  $\approx 0.5 * 200 \text{ Hz} * \text{vel\_gain}$ 
```

OTHER COMMANDS

`dump_errors(odrv0)`

`odrv0.clear_errors()`

`odrv0.save_configuration()`

`odrv0.reboot()`

`odrv0.axis1.controller.input_pos = # Turns (absolute position)`

Trapezoidal Trajectory Position Control

`odrv0.axis0.controller.config.control_mode = 3`

`odrv0.axis0.controller.config.input_mode = 5`

`odrv0.axis0.trap_traj.config.vel_limit = 160`

`odrv0.axis0.trap_traj.config.accel_limit = 1000`

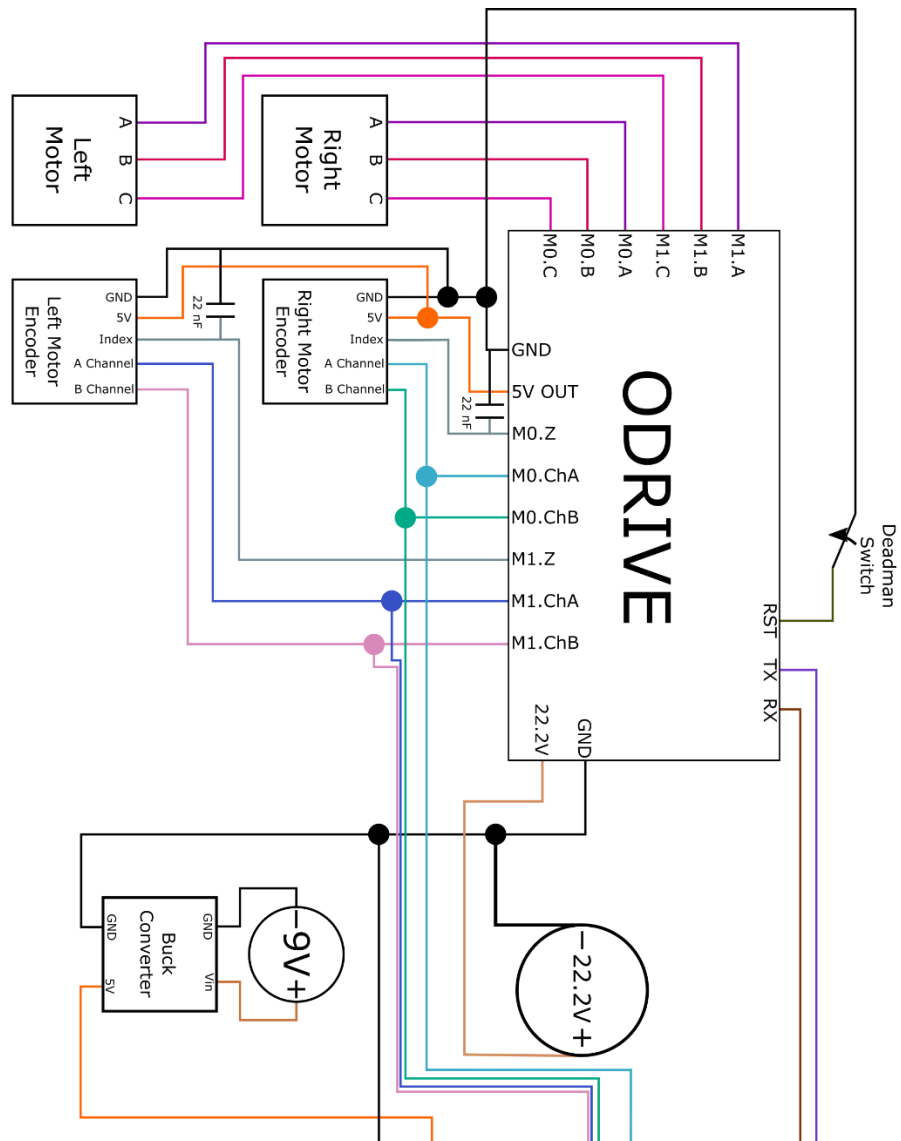
Note: (750 for left leg)

`odrv0.axis0.trap_traj.config.decel_limit = 1000`

Note: (750 for left leg)

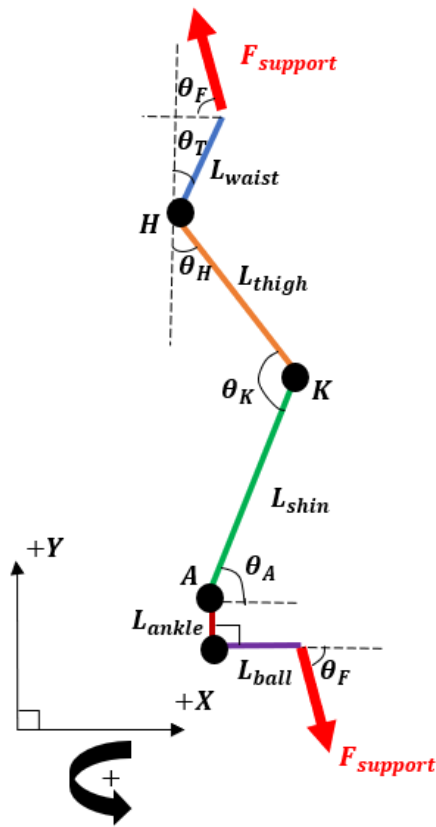
Appendix B: Full Electrical Schematic*

*Modified from [43]





Appendix C: Moment Derivation

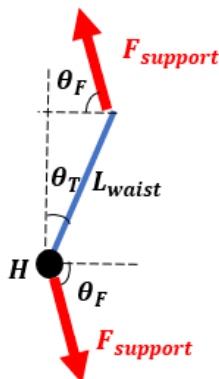


Full Model

5.1 Joint torques on the body due to a ground reaction force at the waist

- Given the nominal biological GRF, compute the joint torques at each joint in the sagittal plane and compare to the nominal biological joint torques

Hip



$$\sum M = -M_H + M_{FH} = 0$$

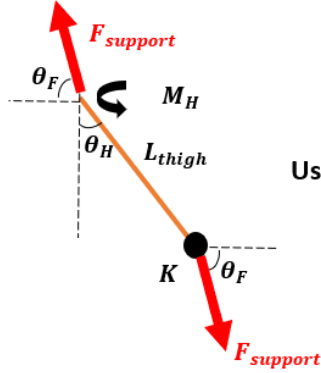
$$M_H = M_{FH}$$

$$M_{FH} = (L_{waist} * \cos(\theta_T) * F_{support} * \cos(\theta_F)) + (L_{waist} * \sin(\theta_T) * F_{support} * \sin(\theta_F))$$

Use Trig Ident:

$$M_H = M_{FH} = (L_{waist} * F_{support}) * \cos(\theta_T - \theta_F)$$

Knee



$$\sum M = M_H + M_K + M_{FK} = 0$$

$$-M_K = M_H + (M_{FK})$$

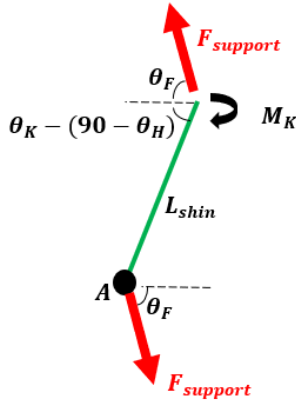
$$M_{FK} = (L_{thigh} * \cos(\theta_H) * F_{support} * \cos(\theta_F)) - (L_{thigh} * \sin(\theta_H) * F_{support} * \sin(\theta_F))$$

Use Trig Ident:

$$M_{FK} = (L_{thigh} * F_{support}) * \cos(\theta_H + \theta_F)$$

$$M_K = -(M_H + (L_{thigh} * F_{support}) * \cos(\theta_H + \theta_F))$$

Ankle



$$\sum M = -M_K - M_A + M_{FA} = 0$$

$$M_A = -M_K + (M_{FA})$$

$$M_{FA} = (L_{shin} * \sin(\theta_K - (90 - \theta_H)) * F_{support} * \cos(\theta_F)) + (L_{shin} * \cos(\theta_K - (90 - \theta_H)) * F_{support} * \sin(\theta_F))$$

Use Trig Ident:

$$M_{FA} = (L_{shin} * F_{support} * (\sin(\theta_K + \theta_H - 90) * \cos(\theta_F)) + (\cos(\theta_K + \theta_H - 90) * \sin(\theta_F)))$$

$$M_{FA} = (L_{shin} * F_{support} * (-\sin(-\theta_K - \theta_H + 90) * \cos(\theta_F)) + (\cos(-\theta_K - \theta_H + 90) * \sin(\theta_F)))$$

$$M_{FA} = (L_{shin} * F_{support} * (-\sin(90 - (\theta_K + \theta_H)) * \cos(\theta_F)) + (\cos(90 - (\theta_K + \theta_H)) * \sin(\theta_F)))$$

$$M_{FA} = (L_{shin} * F_{support} * (-\cos(\theta_K + \theta_H) * \cos(\theta_F)) + (\sin(\theta_K + \theta_H) * \sin(\theta_F)))$$

$$M_{FA} = (L_{shin} * F_{support} * -(\cos(\theta_K + \theta_H) * \cos(\theta_F)) - (\sin(\theta_K + \theta_H) * \sin(\theta_F)))$$

$$M_{FA} = -(L_{shin} * F_{support} * (\cos(\theta_K + \theta_H) * \cos(\theta_F)) + (\sin(\theta_K + \theta_H) * \sin(\theta_F)))$$

$$M_A = M_K - (L_{shin} * F_{support} * \cos(\theta_K + \theta_H + \theta_F))$$

Appendix D: Motor Calculation Code Flowchart

1. Define Constants
2. Calculate Inertia
 - a. $J_{motor} = \frac{1}{2} * m_{motor} * r_{motor}^2$
 - b. $J_{screw} = \frac{1}{2} * \rho_{screw} * r_{screw}^4 * l_{screw} * \pi$
3. Calculate Actuator Length through gait cycle (independent of applied force)
4. Calculate needed motor movement to maintain 0 compression (actuatorDifference)
 - a. Tracks distance from Top Limit Switch to Top of Spring @ 0 compression
5. Calculate Support Ratio through gait cycle to maintain desired force at hip (based on exo geometry)
6. Calculate Desired Spring Force & Needed Spring Displacement to track Support Ratio
7. Adjust desired Compression and Spring Force to account for reaction time before swing phase
8. Position
 - a. $Position_{actuator} = actuatorDifference + displacement$
 - b. $Position_{motor} = Position_{actuator} / ballPitch$
9. Velocity
 - a. $Velocity_{actuator} = \frac{\Delta Position_{actuator}}{dt}$
 - b. $Velocity_{motor} = \frac{\Delta Position_{motor}}{dt}$
 - c. $Velocity_{angular} = 2\pi * Velocity_{motor}$
10. Acceleration
 - a. $Acceleration_{actuator} = \frac{\Delta Velocity_{actuator}}{dt}$
 - b. $Acceleration_{motor} = \frac{\Delta Velocity_{motor}}{dt}$
 - c. $Acceleration_{angular} = 2\pi * Acceleration_{motor}$
11. Torque (Positive torque = rotation to move actuator towards spring, Negative torque = rotation to move actuator off of spring)
 - a. $Torque_{total} = \sum J * Acceleration_{angular}$
 - b. $Torque_{load} = \frac{Force_{spring} * lead}{2 * \pi * \%eff}$
 - c. $Torque_{total} = Torque_{motor} + Torque_{load}$
 - d. $Torque_{motor} = Torque_{total} - Torque_{load}$
12. Power
 - a. $Power_{motor} = Torque_{motor} * Velocity_{motor} * \frac{60}{9.5488}$ (divide by 1000 for kW)

References

- [1] “EXO-H3 - Exoskeleton Report.” <https://exoskeletonreport.com/product/exo-h3/> (accessed Apr. 20, 2022).
- [2] S. Yang, “UC Berkeley Researchers Developing Robotic Exoskeleton that can Enhance Human Strength and Endurance,” *UC Berkeley News*, Mar. 03, 2004. https://www.berkeley.edu/news/media/releases/2004/03/03_exo.shtml (accessed Apr. 20, 2022).
- [3] “REX - Exoskeleton Report.” <https://exoskeletonreport.com/product/rex/> (accessed Apr. 20, 2022).
- [4] Y. Sankai, “Leading edge of cybernics: Robot suit HAL,” *2006 SICE-ICASE Int. Jt. Conf.*, 2006, doi: 10.1109/SICE.2006.314982.
- [5] Z. Yan, N. Li, X. Long, H. Ren, and X. Wu, “Bionic Mechanical Design and Stair Ascending/Descending Gait Planning of A Lower-limb Exoskeleton Robot,” *2018 IEEE Int. Conf. Cyborg Bionic Syst. CBS 2018*, pp. 155–160, Jan. 2019, doi: 10.1109/CBS.2018.8612274.
- [6] “ReWalk™ Personal 6.0 Exoskeleton For Spinal Cord Injury.” <https://rewalk.com/rewalk-personal-3/> (accessed Apr. 20, 2022).
- [7] TWIICE, “TWIICE.” <https://twiice.ch/exoskeletons/> (accessed Apr. 20, 2022).
- [8] Parker Hannifin Corporation, “Indego .” <https://www.indego.com/indego/us/en/home> (accessed Apr. 20, 2022).
- [9] H. Aguilar-Sierra, W. Yu, S. Salazar, and R. Lopez, “Design and control of hybrid actuation lower limb exoskeleton:,” <http://dx.doi.org/10.1177/1687814015590988>, vol. 7, no. 6, pp. 1–13, Jun. 2015, doi: 10.1177/1687814015590988.
- [10] F. Giovacchini *et al.*, “A light-weight active orthosis for hip movement assistance,” *Rob. Auton. Syst.*, vol. 73, pp. 123–134, Nov. 2015, doi: 10.1016/J.ROBOT.2014.08.015.
- [11] A. M. Dollar and H. Herr, “Design of a quasi-passive knee exoskeleton to assist running,” *2008 IEEE/RSJ Int. Conf. Intell. Robot. Syst. IROS*, pp. 747–754, 2008, doi: 10.1109/IROS.2008.4651202.
- [12] B-TEMIA INC, “Keeego.” <https://keeego.com/?ref=steemhunt> (accessed Apr. 20, 2022).
- [13] Lockheed Martin, “ONYX.” <https://www.lockheedmartin.com/en-us/products/exoskeleton-technologies/first-responder.html> (accessed Apr. 20, 2022).
- [14] J. E. Pratt, B. T. Krupp, C. J. Morse, and S. H. Collins, “The RoboKnee: An exoskeleton for enhancing strength and endurance during walking,” *Proc. - IEEE Int. Conf. Robot.*

- Autom.*, vol. 2004, no. 3, pp. 2430–2435, 2004, doi: 10.1109/ROBOT.2004.1307425.
- [15] M. B. Wiggin, G. S. Sawicki, and S. H. Collins, “An exoskeleton using controlled energy storage and release to aid ankle propulsion,” *IEEE Int. Conf. Rehabil. Robot.*, vol. 2011, 2011, doi: 10.1109/ICORR.2011.5975342.
 - [16] J. A. Blaya and H. Herr, “Adaptive control of a variable-impedance ankle-foot orthosis to assist drop-foot gait,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 12, no. 1, pp. 24–31, Mar. 2004, doi: 10.1109/TNSRE.2003.823266.
 - [17] V. Firouzi, A. Davoodi, F. Bahrami, and M. A. Sharbafi, “From a biological template model to gait assistance with an exosuit,” *Bioinspir. Biomim.*, vol. 16, no. 6, p. 066024, Nov. 2021, doi: 10.1088/1748-3190/AC2E0D.
 - [18] T. Wang, Y. Zhu, T. Zheng, D. Sui, S. Zhao, and J. Zhao, “PALExo: A Parallel Actuated Lower Limb Exoskeleton for High-Load Carrying,” *IEEE Access*, vol. 8, pp. 67250–67262, 2020, doi: 10.1109/ACCESS.2020.2986357.
 - [19] A. M. Dollar and H. Herr, “Lower extremity exoskeletons and active orthoses: Challenges and state-of-the-art,” *IEEE Trans. Robot.*, vol. 24, no. 1, pp. 144–158, Feb. 2008, doi: 10.1109/TRO.2008.915453.
 - [20] K. H. Ha, H. A. Varol, and M. Goldfarb, “Volitional control of a prosthetic knee using surface electromyography,” *IEEE Trans. Biomed. Eng.*, vol. 58, no. 1, pp. 144–151, Jan. 2011, doi: 10.1109/TBME.2010.2070840.
 - [21] B. Chen and Q. Wang, “Combining human volitional control with intrinsic controller on robotic prosthesis: A case study on adaptive slope walking,” *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*, vol. 2015–November, pp. 4777–4780, Nov. 2015, doi: 10.1109/EMBC.2015.7319462.
 - [22] C. Li, Y. He, T. Chen, X. Chen, and S. Tian, “Real-Time Gait Event Detection for a Lower Extremity Exoskeleton Robot by Infrared Distance Sensors,” *IEEE Sens. J.*, vol. 21, no. 23, pp. 27116–27123, Dec. 2021, doi: 10.1109/JSEN.2021.3111212.
 - [23] H. Kawamoto, S. Kanbe, and Y. Sankai, “Power assist method for HAL-3 estimating operator’s intention based on motion information,” *Proc. - IEEE Int. Work. Robot Hum. Interact. Commun.*, pp. 67–72, 2003, doi: 10.1109/ROMAN.2003.1251800.
 - [24] Tidy Trax, “Tidy Trax Shoe Cover Product Info.” <http://www.tidy-trax.com/store/pg/7-Product-Info.aspx> (accessed Apr. 21, 2022).
 - [25] ST Microelectronics, “VL6180X Datasheet,” Aug. 2014. Accessed: Apr. 21, 2022. [Online]. Available: www.st.com.
 - [26] TE Connectivity Sensors, “FX29 Compact Compression Load Cell Datasheet,” Oct. 2020. Accessed: Apr. 21, 2022. [Online]. Available: te.com/sensors.
 - [27] Texas Instruments, “TCA9548A Low-Voltage 8-Channel I2C Switch With Reset

- Datasheet,” Jan. 2015, Accessed: Apr. 21, 2022. [Online]. Available: <https://cdn-shop.adafruit.com/datasheets/tca9548a.pdf>.
- [28] Linear Technology, “LTC4311 Low Voltage I2C/SMBus Accelerator,” 2008. Accessed: Apr. 21, 2022. [Online]. Available: <https://cdn-learn.adafruit.com/assets/assets/000/095/254/original/4311fa.pdf?1601481203>.
- [29] US Digital, “MAE3 Absolute Magnetic Kit Encoder Datasheet,” Dec. 2021. Accessed: Apr. 21, 2022. [Online]. Available: <https://www.usdigital.com/support/resources/reference/technical-docs/technical->.
- [30] C&K, “MS Series Small Size Snap-acting Switches Datasheet,” Jul. 2020. Accessed: Apr. 21, 2022. [Online]. Available: www.ckswitches.com.
- [31] OMRON Corporation, “Miniature Basic Switch that Offers High Reliability and Security Datasheet.” Accessed: Apr. 21, 2022. [Online]. Available: www.omron.com/ecb.
- [32] “Turnigy Graphene 4000mAh 6S 45C Lipo Pack w/XT90.” https://hobbyking.com/en_us/graphene-4000mah-6s-45c-w-xt90.html?queryID=fba7e32955225c8c6cb9416a01e4526&objectID=45458&indexName=hbk_live_products_analytics (accessed Apr. 21, 2022).
- [33] Texas Instruments, “LM2596 SIMPLE SWITCHER® Power Converter 150-kHz 3-A Step-Down Voltage Regulator Datasheet,” Apr. 2021. Accessed: Apr. 21, 2022. [Online]. Available: www.ti.com.
- [34] “Turnigy DLUX LIPO Battery Cell Display and Balancer (2S~6S).” https://hobbyking.com/en_us/turnigy-dlux-lipo-battery-cell-display-and-balancer-2s-6s.html?queryID=0b11b17418dd48e28cabb71dee5b359a&objectID=45202&indexName=hbk_live_products_analytics (accessed Apr. 21, 2022).
- [35] “Turnigy Aerodrive SK3 - 5045-500kv Brushless Outrunner Motor.” https://hobbyking.com/en_us/turnigy-aerodrive-sk3-5045-500kv-brushless-outrunner-motor.html (accessed Apr. 21, 2022).
- [36] US Digital, “E3 Optical Kit Encoder Datasheet,” Jan. 2022. Accessed: Apr. 21, 2022. [Online]. Available: <https://www.usdigital.com/pc4/>.
- [37] Espressif Systems, “ESP32 Series Datasheet,” 2022. Accessed: Apr. 21, 2022. [Online]. Available: <https://www.espressif.com/en/support/download/documents>.
- [38] Espressif Systems, “ESP32 Technical Reference Manual,” 2021. Accessed: Apr. 21, 2022. [Online]. Available: <https://www.espressif.com/en/support/download/documents>.
- [39] “Getting Started — ODrive Documentation 0.0 documentation.” <https://docs.odriverobotics.com/v/latest/getting-started.html> (accessed Apr. 21, 2022).
- [40] “Control Structure and Tuning — ODrive Documentation 0.0 documentation.” <https://docs.odriverobotics.com/v/latest/control.html> (accessed Apr. 21, 2022).

- [41] GitHub, “ODrive/Arduino/ODriveArduino.”
<https://github.com/odriverobotics/ODrive/tree/master/Arduino/ODriveArduino> (accessed Apr. 21, 2022).
- [42] “Control Modes — ODrive Documentation 0.0 documentation.”
<https://docs.odriverobotics.com/v/latest/control-modes.html#trajectory-control> (accessed Apr. 21, 2022).
- [43] E. A. Claessen and A. T. Asbeck, “Design of an Underactuated Lower Body Exoskeleton Using a Pantograph,” Virginia Polytechnic Institute and State University, 2022.
- [44] J. Perry, J. M. Burnfield, and L. M. Cabico, *Gait Analysis: Normal and Pathological Function*. SLACK, 2010.
- [45] C. C. Gordon *et al.*, “2012 ANTHROPOMETRIC SURVEY OF U.S. ARMY PERSONNEL: METHODS AND SUMMARY STATISTICS,” Dec. 2014.
- [46] M. Hajaghazadeh, R. Emamgholizadeh Minaei, T. Allahyari, and H. Khalkhali, “Anthropometric Dimensions of Foot in Northwestern Iran and Comparison with Other Populations,” *Heal. Scope*, vol. 7, no. 3, Aug. 2018, doi: 10.5812/JHEALTHSCOPE.14063.
- [47] C. Kirtley, *Clinical Gait Analysis: Theory and Practice*. Elsevier Ltd, 2006.
- [48] “Converting Between Linear Force and Torque of a Ballscrew | Kollmorgen.”
<https://www.kollmorgen.com/en-us/developer-network/calculating-linear-force-and-torque-ballscrew/> (accessed Apr. 21, 2022).
- [49] WEN Technology, “Power-Torque Calculator,” 2002.
http://wentec.com/unipower/calculators/power_torque.asp (accessed Apr. 21, 2022).
- [50] K. W. Hollander, R. Ilg, T. G. Sugar, and D. Herring, “An efficient robotic tendon for gait assistance,” *J. Biomech. Eng.*, vol. 128, no. 5, pp. 788–791, Oct. 2006, doi: 10.1115/1.2264391.