LOWER BODY EXOSKELETON DESIGN AND CONTROL

By: Christian Mueth
Advisor: Dr. Alan Asbeck
Assistive Robotics Laboratory
Virginia Tech Mechanical Engineering Department

Abstract

This research focuses on the development and refinement of a motorized lower-body exoskeleton system for the recovery of astronauts after missions in space. The exoskeleton is designed for strength assistance and rehabilitation of the lower body. It was developed successfully within a budget of only $4000. The exoskeleton system consists of angle encoders, force sensors, distance sensors, spring-loaded supports, and motorized actuators. The exoskeleton supplies a force of 35lbs (155N) per leg, which provides support during locomotion. It is adjustable to be worn over clothes and has been tested with individuals of various body types.

Several control systems have been developed, with the latest utilizing distance sensors that predict the foot's strike. These control systems have been optimized to minimize sensor error, operate at a walking speed of 2.5 mph (4 kph), and successfully climb stairs. Electrical and mechanical modifications have also been made. The design of the exoskeleton and control algorithms has the potential to benefit future mechatronic exoskeleton control systems. This cost-effective and simple system demonstrates that it is possible to create a motorized exoskeleton with limited resources and paves the way for future exoskeletons to enhance recovery and rehabilitation.

System Overview

Figure 1 - Overall System

The exoskeleton system sees actuation via two motors located near the hip. The actuator itself is strapped along the length of the thigh, featuring a series elastic spring connected in series with the motor. Each actuator also has a ball screw connected to the motor, which allows the motor to move a spring plate. This spring plate can compress the spring, translating supportive force to the user. This translation of force is achieved via the actuator pushing on the frame at two points and creating a force between the hip and the ball of the foot (1).

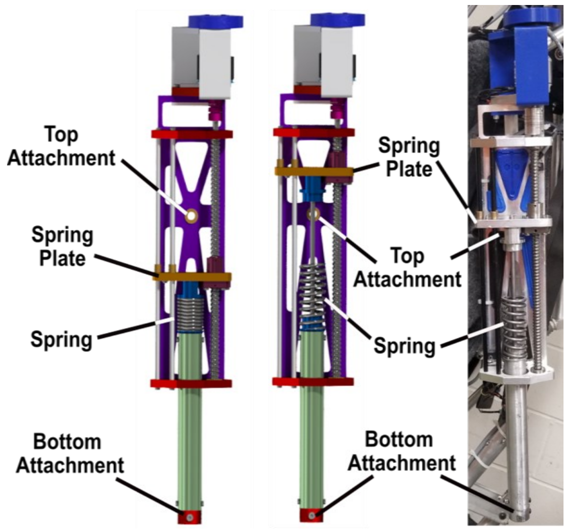

Figure 2 - Actuator Mechanics

The system provides the greatest reaction force when an angle of roughly 120 degrees is made at the knee joint. This angle is monitored by angle encoders.

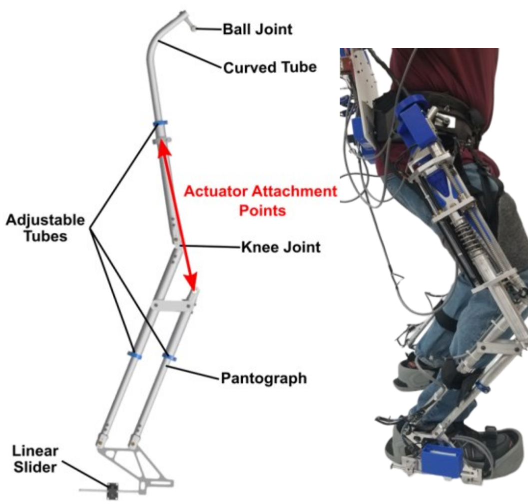

Figure 3 - Actuator Frame

The motor positions are controlled via motor encoders and connection to an ODrive board (2). They are powered by a LiPo battery. A dual core ESP32 microcontroller acts as the central brains for the circuitry, sensors, and control algorithms (3). Multiplexers at the feet collect sensor data and communicate with the microcontroller. Prior control algorithms have utilized force sensors at the feet, which could detect the force of the body. The system's most recent control algorithms use distance sensors at the feet, which can determine how far the foot is from the ground.

System Modifications

Signal reliability was increased by adding strain relief, wire sheathing and longer wiring to critical areas. Modifications were also made to mechanical design, for seamless operation with human biomechanics. A modification that involved several iterations was a new mounting for the motor encoders. The 3D printed mounting for the motor encoders was originally subject to flexure during locomotion, which would cause the motors to read errors. The mounting was changed from one 3D printed part to two 3D printed parts. This allowed the 3D printed mounting to be adjustable and to secure at multiple points on the system's frame.

Figure 4 - Motor Encoder Mounting

Following this, consideration was given toward ergonomics. Adjustable shoulder straps were connected to the back-plate of the system, to better fix the frame to the body. The thigh straps for the system were also altered. Several versions of the thigh straps were considered by the lab before a final design was selected. The shape of these new straps better secured the system to the thigh, maximizing the translation of force from the actuator.

Figure 5 - Thigh Straps

Finally, actuator covers were designed. These protected the user from accidentally interacting with the actuator mechanics, as the actuator supplied force.

Figure 6 - Actuator Cover

In terms of electronics, a 4S LiPo battery was replaced with a 6S LiPo battery with double the capacity. The higher voltage and doubled capacity allowed for longer tests. It prevented current overdrawing during actuation at high speeds. The LiPo battery housing was also redesigned and moved to the center of mass.

A low voltage battery on the system was given a fitted enclosure and a switch circuit. Multiplexer cases at the feet were made more secure with screw brackets to the frame. These brackets removed a prior need for securement via press fit. The ESP32, ODrive, and multiplexer were also all given 3D printed covers, to protect them from dust. Paddles were added to limit switches, helping them withstand higher speeds from the actuator. All of these modifications added durability during field tests of the control systems.

Distance Sensor Controls

The control system was originally designed to utilize force sensors. These sensors detected the force of a user's foot on the ground. When force was detected by the force sensors, the actuators were set to actuate. However, these force sensors were suboptimal. They were prone to wearing out and their data would fail to detect applications of force by the user. Thus, it was necessary to design a distance sensor algorithm that could reliably detect distance data.

Two distance sensors were located within each foot's sole, at the heel and ball of the foot. They could be used to detect distance from the foot to the floor. However, problems were seen by previous teams in implementing distance sensor detection (4). Distance sensor data for the system featured noise that had made the sensors unworkable.

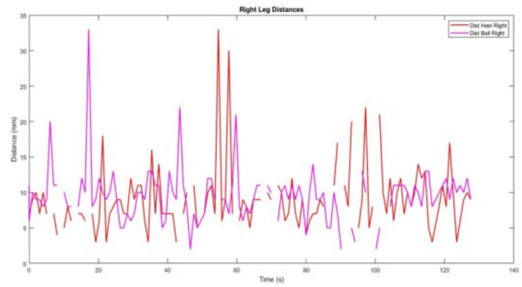

Figure 7 - Early Distance Sensor Noise

It was discovered that various parameters for the sensors were overclocked in the code. Using the data sheet for the sensors, the distance sensors were steadily tuned to provide better data (5). For example, the integration period was changed from 5 to 55. Once the distance sensors were tuned to work more in line with documentation, the distance sensors began to read clean signals. Several preliminary runs of data were collected at 0.5 mph (0.8 kph), which showed great improvement.

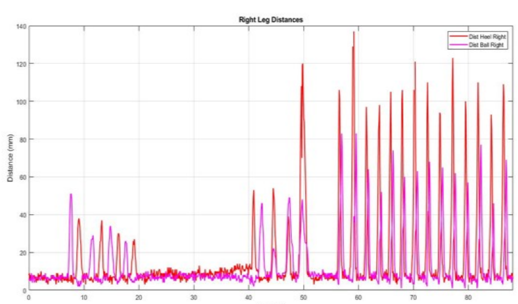

Figure 8 – Tuned Distance Sensor Data 0.5 mph (0.8 kph)

Following this, C++ code was built using the distance sensor data to actuate the system. When the sensors detected that the foot was below the threshold of 0.787 inches (20mm), the actuator was set to actuate. Specific emphasis was placed on the heel sensor. During an individual's walking cycle, the heel is key in determining whether somebody is placing their foot on the ground (stance) or has their foot in the air (swing).

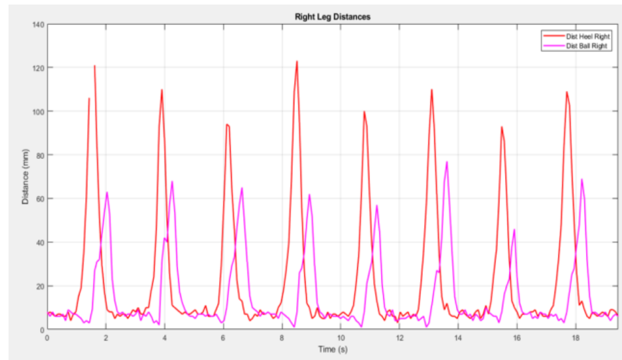

Figure 9 – Threshold Method Distance Data at 0.5 mph (0.8 kph)

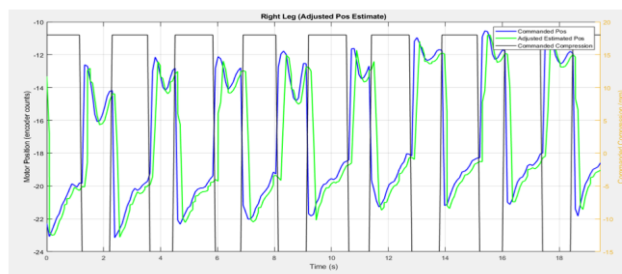

Figure 10 - Threshold Method Compression Data at 0.5 mph (0.8 kph)

This method, while rudimentary, was reliable. Below the set threshold for heel distance, the system would detect a strike state and send a signal for the motors to change position for actuation. Above the set threshold, the system would detect a swing state and send a signal for the motors to release actuation. Successful tests were performed up to speeds of 2.5 mph (4 kph).

Predicting Strike

While a simple threshold control system performed well, the actuator's reaction time featured a delay to the user at high speeds. For ideal performance, the system needed to be able to predict when a user was going to strike the ground. This would ensure rapid actuation and optimal support for the user. It was decided that a third state in the walking cycle would be defined, similar to what was used in a system called the HAL-3 (6). In addition to strike and swing, there would be a state called “descent” that the control algorithm would monitor for.

A control loop was made using reference parameters. The distance sensors would take in values. Once there were three sets of values, a difference was taken between the oldest distance value and the mid-value. Another difference was taken between the newest distance value and the mid-value. If the differences were each negative, this indicated a negative slope. A descent state would be active, and the motor would be told to home near the actuation point.

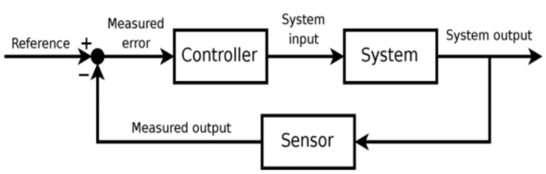

Figure 11 - Control Loop Example (7)

The 0.787-inch (20mm) threshold value for the heel sensor remained critical in determining strike condition. Swing still became active if the heel lifted from the ground. However, the swing state also became active out of the descent state if the slope was detected as rising. Several lab tests were performed with this algorithm, and it performed well.

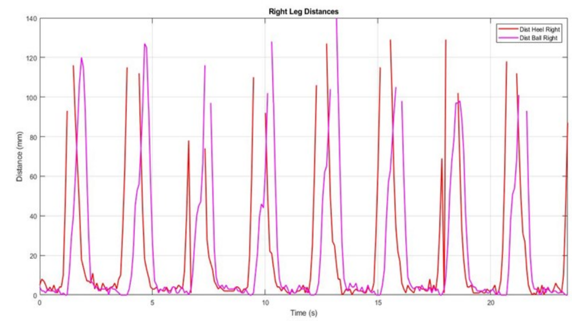

Figure 12 - Prediction Method 1 Distance Data 0.5 mph (0.8 kph)

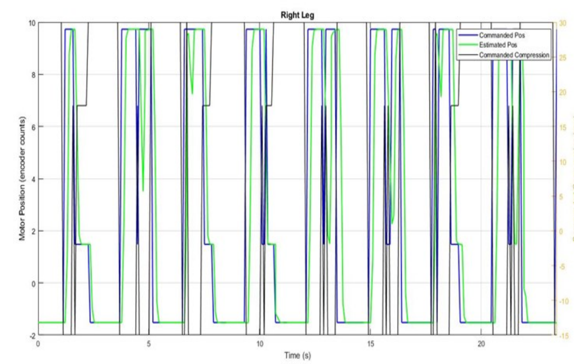

Figure 13 - Prediction Method 1 Distance Data 0.5 mph (0.8 kph)

That said, the algorithm still had its limitations where it would fail to detect descent or mistakenly detect swing. A second algorithm was built with error correction. This algorithm monitored the distance sensor data for characteristic noise errors and would correct them. It was initially a concern that the second algorithm might see lag while performing error correction, but this was not the case. This second algorithm saw improvement over the initial algorithm.

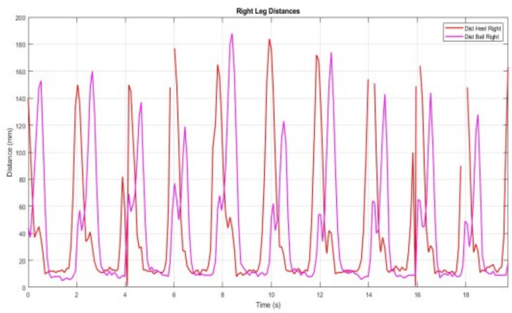

Figure 14 - Prediction Method 2 Distance Data 0.5 mph (0.8 kph)

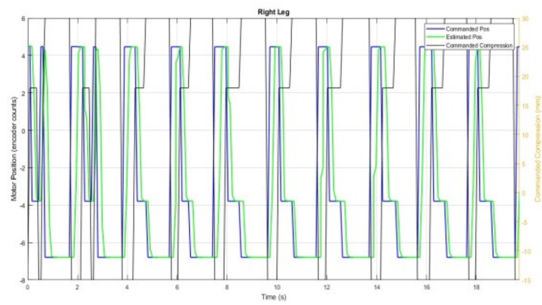

Figure 15 - Prediction Method 2 Compression Data 0.5 mph (0.8 kph)

A kinematic method was also developed, inspired by another exoskeleton designed for real-time event detection (8). Using the size and direction of the detected differences in distances per time step, a time for strike could be predicted. However, while the kinematic method has potential, it is currently too prone to unreliability. This third methodology is still being refined and likely requires higher sensor data rates. Of the three methods tested, the second algorithm performed the best. It became the default algorithm in further tests.

The system was tested across several body types. While all body types gave similar data, it was observed that taller body types provided slightly better data sets. Taller body types provided more data points between the peak of a swing state and heel strike. This was due to larger strides, which kept the foot in the air for longer time steps. There was a longer descent curve for the algorithm to predict with.

Sample Descents: Peak-to-Strike	
Height - 6’5” (2000mm)	**Height – 5’10” (1780mm)**
10in (255mm)	10in (255mm)
3.9in (99mm)	3.1in (79mm)
2in (52mm)	0.75in (19mm)
0.7in (18mm)	-

Table 1 - Sample Descents: Peak-to-Strike

Field Testing

The system was tested in several environments, to good performance. These environments included treadmills, hallways, a parking lot, and stairs. Stair behavior for the system worked better than predicted, with the sensors being able to detect the steps reliably.

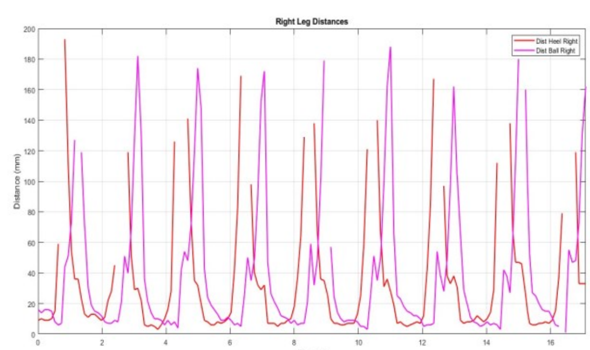

Figure 16 - Stair Distance Data 0.5 mph (0.8 kph)

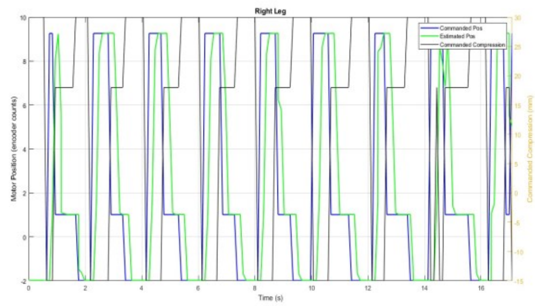

Figure 17 - Stair Compression Data 0.5 mph (0.8 kph)

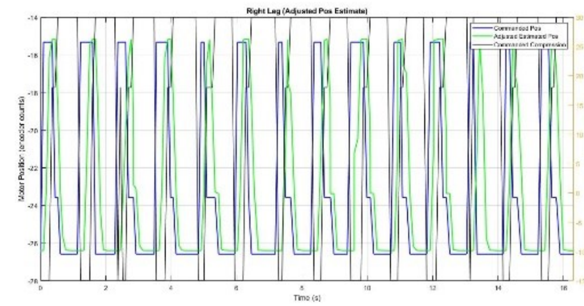

Figure 19 - Compression Data 2.5 mph (4 kph)

Following the initial stair tests, another code was written to cover stance cases on stairs where the heel distance sensor hung over the edge of a step. If the ball of the foot was detected below a set stance threshold and the heel slope was relatively static, the actuator would actuate. This control methodology has yet to be integrated into the main algorithm.

Finally, the predictive distance sensor algorithm was repeatedly verified to perform at speeds of 2.5 mph (4 kph). The system is reliably able to read data, predict strike states, and actuate at 2.5 mph (4kph) speeds for roughly a minute.

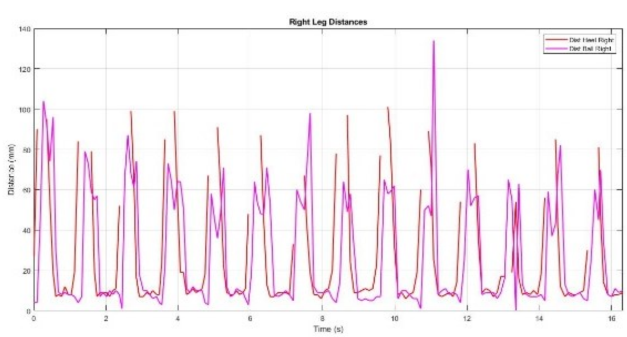

Figure 18 - Distance Sensor Data 2.5 mph (4 kph)

It is worth noting that the distance sensor data at 2.5 mph (4 kph) tends to see cutoff at the peak of swing and features noise. However, prediction and actuation still work effectively, which is important proof of practical use case. The required distance data during transitions from stance, swing, and descent states remains accurate at these speeds.

Future

Going forward, the primary goal is to increase system accuracy and precision. Bugs still occur, and finding a way to extract more data points from the distance sensors would be of great benefit. A kinematic distance method, in particular, could see drastic improvement. The stair detection for heel overhang still needs to be implemented in the main algorithm. In addition, research is being done into how the angle encoders for the system can aid in state prediction. Consideration is being made on how to detect steep inclines, so that the system can provide more support under steep incline conditions. Finally, lessons learned from this control system can be applied to future systems, which are currently in development.

Conclusion

The exoskeleton system has seen great strides, even if there is still much work to be done.

This research demonstrates that it is possible to develop an affordable motorized exoskeleton with predictive actuation, which can operate under a range of conditions. There is potential for similar systems to provide rehabilitation and strengthening for individuals in the process of lower-body recovery, such as from space-missions.

Acknowledgements

I want to thank all of my lab partners who I have worked with on this project. Zackory Biggers, Evan Claessen, Alex Gioia, Seetharam Krishnapuram, Nathan Folta, and Musharrat Mau were each instrumental in the project's progress. Also, thank you Dr. Alan Asbeck and the Virginia Space Grant Consortium for providing me with this opportunity. Finally, I'd like to thank my family for all their support.

Works Cited

[1] E. A. Claessen and A. T. Asbeck, "Design of an Underactuated Lower Body Exoskeleton Using a Pantograph," Virginia Polytechnic Institute and State University, 2022.

[2] "Control Modes — ODrive Documentation 0.0 documentation." https://docs.odriverobotics.com/v/latest/control-modes.html#trajectory-control (accessed Apr. 10, 2023).

10.1109/JSEN.2021.3111212.

[3] GitHub, "ODrive/Arduino/ODriveArduino." https://github.com/odriverobotics/ODrive/tree/master/Arduino/ODriveArduino (accessed Apr. 10, 2023).

[4] Biggers, Z. J. (2022). Design & Evaluation of an Underactuated Lower Body Exoskeleton. Unpublished master's thesis, Virginia Tech.

[5] Vishay Semiconductors. "VCNL4040X01 Datasheet." Retrieved from https://www.vishay.com/docs/84465/vcnl4040x01.pdf (Accessed Apr. 10, 2023).

[6] 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.

[7] Orzetto - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=5000019

[8] 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: