Worcester Polytechnic Institute

Major Qualifying Project

QRP
Quadrupedal Robotics Platform

SUBMITTED BY:
Zachary Armsby, Robotics Engineering and Computer Science
Richard Cole, Robotics Engineering and Mechanical Engineering
Brandon Coll, Robotics Engineering and Computer Science
Joseph Martin, Robotics Engineering and Computer Science
Andrew Nagal, Robotics Engineering

ADVISED BY:
Gregory Fischer, Associate Professor
Mechanical Engineering, Biomedical Engineering, Robotics Engineering
William Michalson, Professor
Robotics Engineering, Electrical and Computer Engineering
Craig Putnam, Associate Director
Robotics Engineering

Submitted in partial fulfillment of the requirements for the degree of Bachelor of Science

April 25, 2018

Abstract

The purpose of this project is to take lessons learned from past MQPs, current industry products, and current research to create a quadrupedal platform capable of attaining unsupported walking. The team designed the platform to utilize series elastic actuation, force-sensing feet, and custom hardware to create a modular and easily expandable platform for future project use. CNC milling and water-jetting were used to manufacture the complete platform which was then vigorously tested under its own weight to determine its capabilities.

Acknowledgements

Thanks to our advisors, who kept pushing us to keep the project on track over the course of this year. Without you, we would not have gotten anywhere near as far as we did.

Thank you to Prof. Loris Fichera, Prof. Nicholas Bertozzi, Prof. Bradley Miller, and Kevin Harrington for their help, advice, and problem solving skills through this project.

Thank you to Howard Products, Harmonic Drive LLC, and Maxon Precision Motors Inc. for working with us to get the products we needed at the lowest cost we could.

Thank you to WPI Washburn Shops for letting us use your equipment to create our project.

Finally, we would like to thank Keion Bisland, Keshuai Xu, Rachel Rynazewski, Xavier Little, Timothy Bell, Christian Cooper, Nicoli Liedtke, and anyone else who helped us along the way. Without the help of people like you, this would not have happened.

Contents

List of Figures

List of Tables

1 Introduction

To satisfy Worcester Polytechnic Institute's Requirements for the Major Qualifying Project (MQP) this team decided to design and construct a quadrupedal robotics platform. The main motivation for this project is the potential benefit a quadrupedal robotics platform can bring to the WPI Robotics program. The intent is for this platform to be utilized by future MQPs or even WPI research projects as a quadruped can be used for a variety of applications since they have a high potential for transversability and mobility. While working on this MQP throughout the 2017-2018 academic year the team will utilize their research and personal experience to analyze the problem that walking on four legs provides and create a platform that will hopefully benefit many future WPI students and faculty.

The quadruped platform itself poses interesting mechanical challenges (as can be seen in the previous HydroDog and Sabertooth MQPs) that the team must overcome in order to make sure the platform can balance and maneuver well. It is especially important to make sure the platform can stabilize well so it can provide a steady base for equipment mounted on the platform. The motion of the platform also might cause difficulty in navigation, something that the team wishes to analyze and find solutions for.

Initial analysis of previous legged-robot MQPs indicates that most of the problems those projects faced were attributed to the scale of the system they created. As the system increases in size, it must also increase in complexity and power due to its increased weight. For example, when the mass increases on the chassis of a quadruped each joint must sustain more torque. This creates issues with hysteresis on lower joints and the need for more complicated legs. This also means that the motors and physical structure required cost more money because they have to handle more force. On the other side of the spectrum, a small system, while benefiting from the reduced weight, requires much more precise control due to a much more volatile center of mass, thereby increasing the difficulty to implement and cost of the control systems and motors (as the ARMS MQP team found to be the case). Based upon this brief analysis, the team concluded that a medium sized platform in the middle of the spectrum is ideal. In this case the system would not be heavy enough to require more expensive motors based upon weight and not small enough to cause the control systems and motors to need to be extremely precise. The intended size of the system is approximate to a medium size dog (much like one of the quadruped MQP teams attempted to do last year) which the team believes is the best size for a project of this scope.

One of the largest tasks for this MQP is the design and in order for this project to be successful the team will need to complete a full model of the robot that is suitable for manufacturing and fabrication. This model will be created in Solidworks and will simplify other necessary tasks such as prototyping, manufacturing, and physical fabrication. Designing the overall robot will also include many considerations such as battery selection, motor selection, sensor selection, miscellaneous electronics selection, and the placement of electrical components on the robot. Of course many of these design considerations will require calculations including torque requirements and stress calculations in order to verify the feasibility of the design before it is constructed.

A complete quadrupedal robot platform will need to have the ability to walk, so the controller is another vital aspect of this MQP. Depending on the design approach, the team will have to design and utilize a GAIT controller for leg movement so the platform can walk

properly. This control system design will be determined through their research and what sensors are utilized in the final design.

2 Background Research

In order to get a better understanding of how to approach this project, several key topics associated with quadrupedal robot designs were researched. The team's research on these topics can be found in the following sections.

2.1 Research on Quadrupedal Robots

Research on previous and existing legged robots was deemed necessary so the team could gain proper insight on the design considerations and requirements for a quadrupedal robot.

2.1.1 WPI Sabertooth

Figure 2.1: Sabertooth MQP

The WPI Sabertooth Robot was a Major Qualifying Project for the 2010-2011 academic year. This robot had gained much inspiration from the anatomy of animals such as lions and quadrupeds such as BigDog by Boston Dynamics. The entire platform weighed approximately 300lb and unfortunately never actually walked as the robot was designed to support approximately 200lb. Despite this shortcoming, it was able to demonstrate leg motion that correlated with dynamic walking gaits while it was supported by a stand. Each leg of Sabertooth had three degrees of freedom with cable driven shoulder and knee joints to make the design of the leg more compact. To move the mass of the leg motors further into the body and reduce their contribution to the leg's inertia, Sabertooth attempted to implement a two degree of freedom body joint. This also would have allowed the robot to have greater flexibility when going around tight corners and ascending stairs.

According to the Sabertooth Team's MQP report the construction of the cable driven system proved to be the most difficult aspect of the robot, a problem that we took into consideration when the team designed our own quadruped. We also learned from their

mistakes of underestimating the final weight of the system and will over-design our robot's strength to hopefully avoid the same problems.

2.1.2 MIT Cheetah

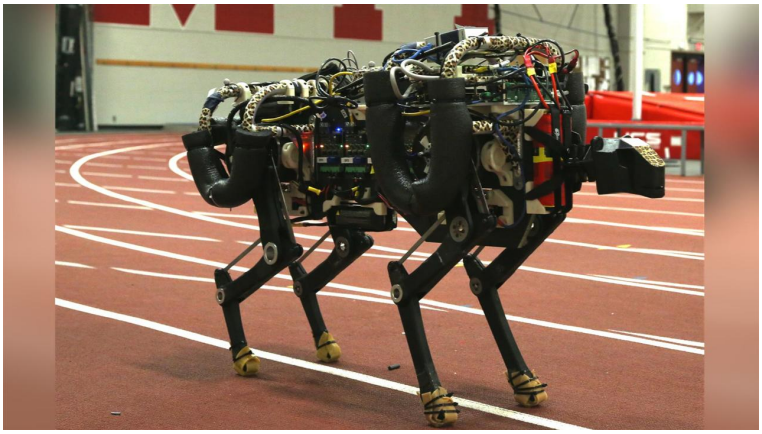


Figure 2.2: MIT Cheetah

Cheetah is a quadruped robot developed at the Massachusetts Institute of Technology. Its purpose is to provide a platform for students to study dynamic locomotion capabilities. This robot is known to currently be one of the fastest quadruped robot platforms as it can reach speeds of up to 6.4 m/s. Additionally, Cheetah can jump over obstacles 40cm in height while running 2.5 m/s. It accomplishes these feats by utilizing combinations of the trot and gallop gaits. In the group of quadrupeds we researched it is certainly one of the most mobile and dynamic.

Cheetah is designed with the gallop gait in mind. It's legs are oriented so that the shock of landings and takeoffs are delivered directly into the legs rather than into the joints. Additionally, it uses hydraulics to produce the required power to push the robot off of the ground. Due to us focusing on the crawl and walk gaits, Cheetah's design decisions are of interest to us more as proof of concept and indications of how to handle the intense dynamic loading a quadruped will undergo rather than contributing directly to our design.

2.1.3 Ghost Robotics Minitaur

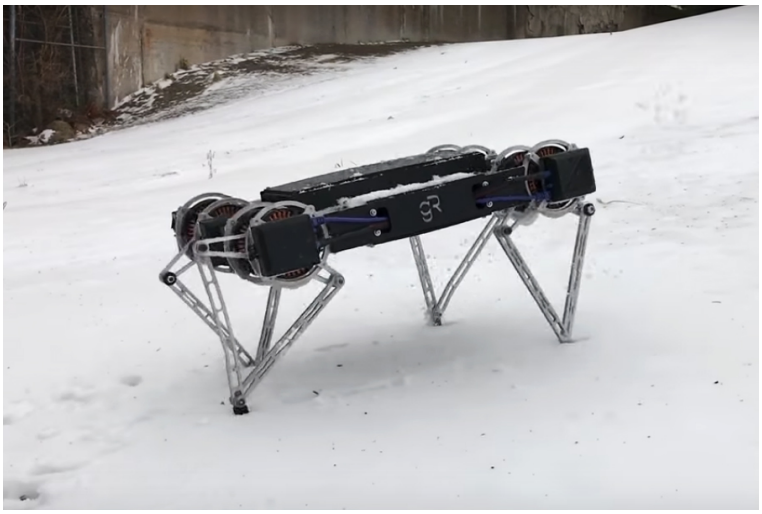


Figure 2.3: Ghost Robotics Minitaur

The Minitaur from Ghost Robotics is a relatively unique and lightweight quadruped (only 6kg) in that it utilizes direct drive motors. These direct drive motors can be programmed to behave like a damped spring system that can even be tuned with software. This is useful when trying to walk over rough, unknown terrain. The lack of a gear train also makes the robot more robust as there are no gears to break. Minitaur has demonstrated the ability to walk over very difficult terrain such as ice and is capable of walking gaits of up to 2.0 m/s.

This design is of interest to us and worth mentioning due to its modeling of motors as springs. This concept could allow us to apply compliance to a completely motor driven system, something that we are interested in. Of the robots mentioned so far, this is the only robot with any amount of compliance, something that is hard to incorporate into quadrupeds due to the required high precision movements and the presence of both high and low torque operation during a periodic gait. The team would like to add compliance to the system to decrease the jerk that the gearboxes and motors would undergo during motion. However, building a compliance system into a quadruped is a complicated task, so simulating one with motors might be an easier solution, something we looked into during our design.

2.1.4 Boston Dynamics SpotMini

Figure 2.4: SpotMini

SpotMini is a quadrupedal robot platform that was developed by Boston Dynamics in 2016 as a smaller version of their famous Spot Robot. SpotMini was a major source of inspiration for this project as its size and weight made it an ideal platform for many different uses. The quadruped weighs 30kg and is completely electrically actuated, unlike its hydraulically actuated predecessor. The platform can fit within a 1x1x1 meter cube and can support a payload of up to 14kg. Each leg on SpotMini has three degrees of freedom: a hip joint, a shoulder joint, and a knee joint. To assist with general control of the robot, SpotMini utilizes position sensors in each joint and force sensors within its limbs. This combination grants SpotMini a high degree of mobility that makes it capable of crawling and walking gaits. To compliment this mobility, SpotMini has an extensive set of sensors (including stereoscopic and depth cameras) that allow it to navigate in a complex environment and locate objects of interest.

The overall design of SpotMini is extremely efficient for its size, making it a highly relevant reference for the project team both in terms of inspiration and proof of concept. We also used SpotMini as an prime example of general leg structure and gait motion due to its having the same leg orientation as our design.

2.1.5 ANYbotics ANYmal

Figure 2.5: ANYmal

ANYmal is a multi-use quadrupedal platform developed initially for the ARGOS competition which challenged competitors to construct robots that were capable of navigating and inspecting oil & gas platforms. In the pursuit of this competition, researchers at ETH Zurich developed a quadrupedal robot platform capable of dynamic walking and crawl gaits of up to 1.0 m/s. Each leg has the same degrees of freedom as SpotMini, but because of the construction of the legs each joint is capable of rotating a full 360 degrees. The entire robot weighs 30kg and is robust enough for many outdoor conditions, making it an ideal research platform. Each joint of the ANYmal robot is direct-driven by precise actuators the ANYmal team developed themselves for the robot. For navigation and path planning the ANYmal utilizes cameras and LIDARs on the back and front of the robot to map out unknown environments.

This quadruped was used for inspiration when designing our leg structure due to its unique 360 degree rotation of each leg. However, due to the motors being custom-designed for the purpose, the design choices in ANYmal are not entirely useful to our team and the robot was used more as a proof of concept for navigation and stability control.

2.1.6 ETHZurich StarlETH

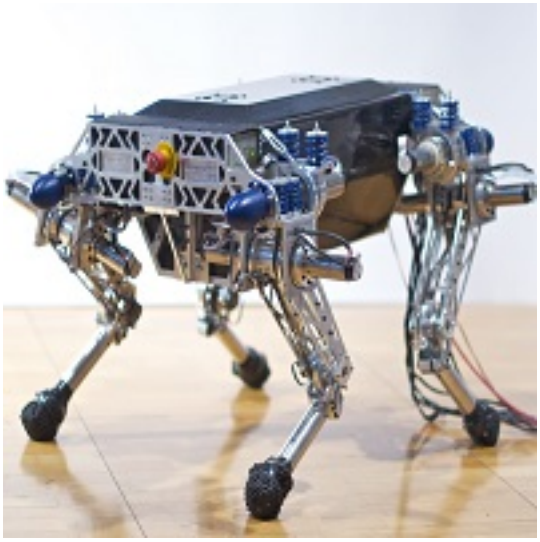


Figure 2.6: StarlETH

StarlETH is a thesis project to develop a quadrupedal platform that is based on nature where elastic tendons are used to store and recover energy, therefore creating a robot that is speedy, versatile, efficient, and robust. The main aspect of this robot that made it different than the state of the art is its use of series elastic actuation to include compliance in each joint of the leg. This allowed them to store energy during motion, however, it added complexity to the position control of the legs due to the oscillation that springs add to a system. The StarlETH paper also goes in depth into the dynamic modeling, control equations, and design of the entire system which makes it a incredibly helpful paper for the team.

The team really liked the addition of series elastic actuators to add compliance to the legs. The in-depth description of the complications that series elastic actuation provides to position control as well as how to deal with these complications were very useful. Additionally, the requirements that the paper set out to meet are almost exactly the ones that we set out for our project. This made the paper very relevant to our design, and the team relied heavily on this paper and the design decisions that went into StarlETH.

2.2 Movement Gaits and Control

One of the main things that makes quadrupeds (and legged robots in general) so different from other forms of robotics is how they move. Quadrupeds are not always statically stable in their motion which requires having precise calculation and control of features such as the center of mass, position and motion of legs, and body orientation (Hwang, 2008; Moro et. al, 2013; Raibert, 1990; Rebula et. al, 2007; RunBin, 2013; Shkolnik, 2007). Fortunately, unlike bipedal robots, a quadruped has many different strategies by which it can move its limbs to achieve motion of the body in a defined direction. These strategies are known as gaits, the manner by which one walks. Most of the research on optimal gaits and motion of gaits has been done by studying the motion of animals such as horses, dogs, and cheetahs (Moro

et. al., 2013; Raibert, 1990). While the motion of spined animals are slightly different than those of most quadrupeds, these studies have found that most motion follows one of four gaits: crawl, walk (otherwise known as trot), pace, and bound (otherwise known as gallop). These gaits can be defined through the use of certain gait parameters (Lee & Song, 1991):

- Stroke, R, is the distance a foot travels in relation to the body in the support phase of the cycle.
- Angular stroke, Φ, is the angle that a leg rotates about its turning center during the support phase of the cycle.
- Stroke pitch, p, is the distance between the centers of two adjacent legs (on the same side).
- Duty factor, β, is the time fraction that the leg is in contact with the ground.
- Stride, γ, is the distance that the body travels in one cycle. ($\gamma = \frac{R}{\beta}$ for periodic, straight line gaits)
- Phase, ψ, is the time fraction that the placement of a leg lags behind that of the first leg of the cycle.
- Crab angle, α, is the angle between the direction of motion and the positive direction of the body coordinates.
- Stability margin, S_m, is the shortest distance between the projection of the COM onto the support polygon and the edge of the support polygon.
- Longitudinal stability margin, S_l, is the distance along the instantaneous direction of motion between the projection of the COM onto the support polygon and the edge of the support polygon.

2.2.1 Crawl Gait

Out of these aforementioned gaits, the crawl gait is the only statically stable gait due to the robot maintaining at least three points of contact with the ground at all times (Hwang, 2008; Lee et al., 1991; Pongas, 2007; Raibert, 1990; Rebula et al, 2007). It does this by only lifting one leg at a time to a new location while swinging the other legs to create forward motion. Assuming that the front left leg is leg 1, the front right leg is leg 2, the back left leg is leg 3, and the back right leg is leg 4, the motion of this gait is described in Figure 2.7.

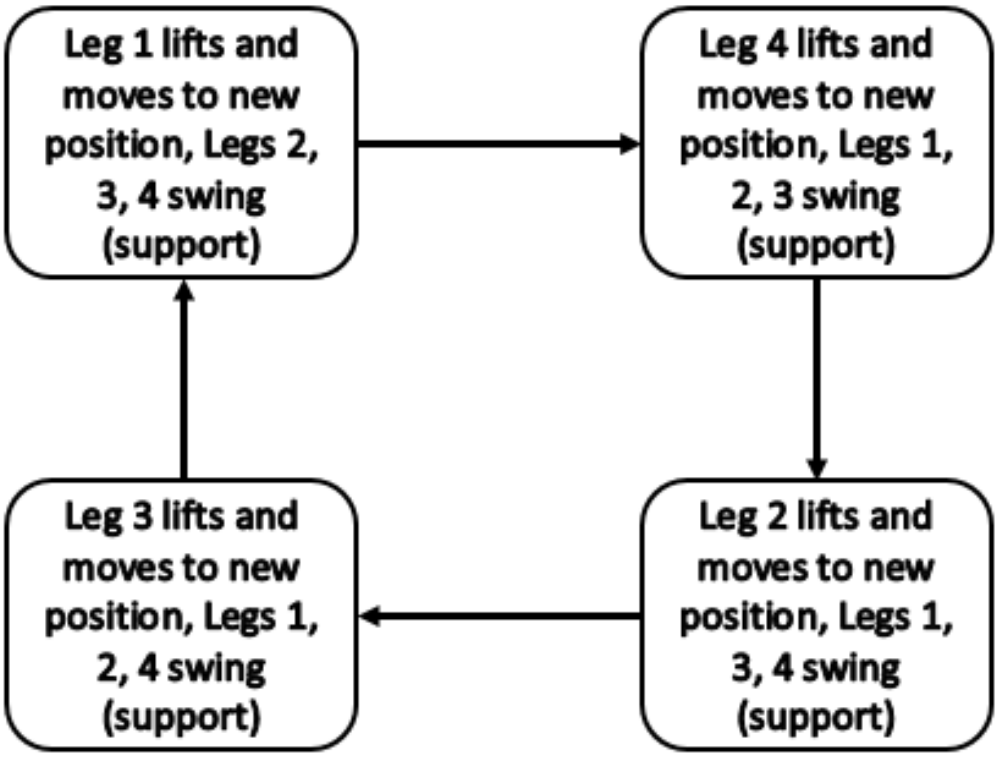


Figure 2.7: Crawl Gait Block Diagram

The crawl gait is slower than the rest of the gaits, but is ideal for obstacle strewn environments because the placement of each leg can be calculated to best maintain the balance of the robot. Additionally, the stability of the gait allows for more complex leg placement calculations to run without having to worry about the robot maintaining its balance, another reason why this gait is preferred in uneven terrain. Due to the stable nature of this gait and its usefulness on uneven terrain, a significant amount of research has gone into studying and creating ideal controllers for such a system.

Control of the robot during the crawl gait can be done in many ways (Chen, 2000; Dekker, 2009; Hwang, 2008; Moro et. al, 2013; Pongas, 2007; Raibert, 1990; Rebula et. al, 2007; Rudy, 2014; RunBin, 2013; Shkolnik, 2007). The two most popular methods are through calculating the Center of Mass (COM) or calculating the Zero-Moment Point (ZMP), both of which are explained more in Section 2.2.5. If both of these are within the support polygon made by the three legs on the ground, the robot is in a stable position, although usually only one is computed for computational speed reasons. However, once either of these points moves outside of the support polygon (the ZMP will move outside the support polygon before the COM if the robot is moving), the system becomes unstable.

2.2.2 Walk Gait

The walk gait moves the two diagonal legs at once, as seen in Figure 2.8, and therefore is not always statically stable, however, it is dynamically stable under certain conditions (Moro et al, 2013; Raibert, 1990).

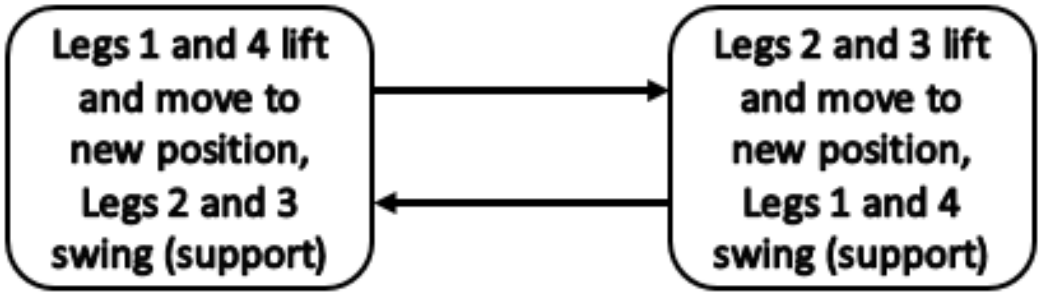


Figure 2.8: Walk Gait Block Diagram

In order for the robot to be dynamically stable, the ZMP and COM must be on (or above) the line created by the two supporting legs (Pongas, 2007). This requires a slight side to side motion of the body while moving forward, necessitating a more complex control system than required for a crawl gait. Due to the walk gait's stability, this method of motion is often used for speeds that are realistically unattainable for a crawl gait, however this gait become much more complex in an environment with an uneven surface.

2.2.3 Pace Gait

The pace gait is similar to the walk gait with the exception that the lateral pair of legs are moved in tandem creating a motion like the one described in Figure 2.9 (Moro et al, 2013; Raibert, 1990).

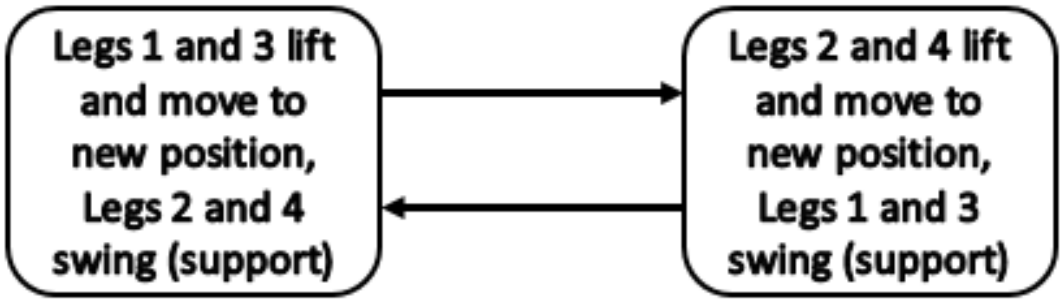


Figure 2.9: Pace Gait Block Diagram

The control of the pace gait is similar to that of the walk gait (Pongas, 2007; Raibert, 1990). However, unlike the walk gait, this gait is not dynamically stable without significant

sideways motion of the body or having high crab angles in the legs. This causes the pace gait to be able to utilize less of the leg's full stroke for forward motion and therefore is slower. However, the pace gait allows for the quadruped to avoid possible interference between the front and back legs at lower speeds, making it potentially more ideal at slower speeds. Additionally, some research has gone into pushing off of the ground with enough force that the robot body moves up as the legs leave the ground and comes back down as the legs complete their placement, making the gait act more like a sideways bound gait than the walk gait.

2.2.4 Bound Gait

The bound gait is similar to the pace gait with the exception that it moves the forward and back pairs of legs at the same time by bounding off of the back legs and landing on the front legs, as described in Figure 2.10 (Moro et al, 2013; Raibert, 1990).

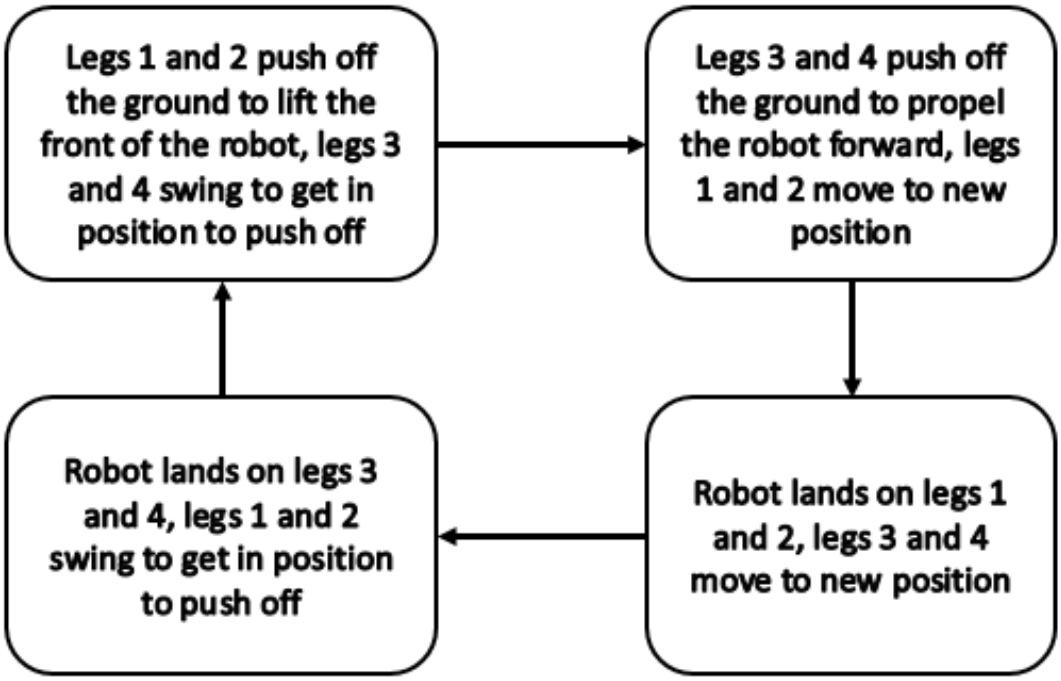


Figure 2.10: Bound Gait Block Diagram

This gait requires more force to be generated in the legs of the robot since they must be able to lift the robot completely off of the ground for part of the cycle (Moro et al, 2013; Raibert, 1990). This gait also inherently creates a lot of dynamic forces and vibrations on the legs and any equipment mounted on the body. This often requires the use of pneumatic systems for rapid force generation and damping. Control of this type of gait relies more on generating the required moments and forces to lift the robot off of the ground and predict its flight rather than specific leg placements seen in the other gaits. As this gait is often

used for faster motion, the inertia of the robot becomes more important when calculating how long the step lengths should be and how fast the cycle time is.

2.2.5 Gait Control

There is much disagreement among academics about what is the best way to control the motion of quadrupeds. Many researchers indicate that their controller works the best in certain circumstances (Hwang, 2008; Rebula et. al, 2007; RunBin, 2013; Shkolnik, 2007). However, much of the discussion centers around the priority of controlling the body or the individual legs and the best process for controlling them. According to J.R. Rebula et. al. (2007) and Shkolnik and Tedrake (2007), the control of individual legs is almost always done using inverse kinematics. However, this can be complicated by quadrupeds usually having redundant degrees of freedom in their actuation leading to infinite solutions to attain a desired ending position. In these cases, the Gradient Projection, Weighted Least-Norm, or Extended Jacobian matrix methods are usually used due to their abilities to provide closed joint paths, with a significant amount of research pointing to the superiority and usefulness of the Extended Jacobian method for real-time control (Hwang, 2008; RunBin, 2013; Shkolnik, 2007). Controlling the body orientation of a quadruped is even harder, specifically due to the fact that most quadrupeds are point-foot robots where the exact angle that the foot meets the ground is inherently undefined, unless the ground is assumed to be perfectly flat, due to it not having an "ankle" joint. This inherently means that the center of mass and orientation of the body is hard to define through regular inverse kinematics. For this reason, most of the research into quadruped motion looks at stability through other methods such as the Static Stability Margin, Zero-Moment Point Margin, Resolved Momentum Control, or a hybrid of inverse kinematics and whole-body Jacobians. Computationally, all of these methods are complex and require varying amounts of simplification to run in real-time on a quadruped, but all of them have been shown to be feasible on developed research platforms and through simulation.

The simplest control methods calculate the COM or ZMP and define stable body orientations as well as the necessary leg positions for the entirety of a gait (Chen, 2000; Dekker, 2009; Hwang, 2008; Moro et. al, 2013; Pongas, 2007; Raibert, 1990; Rebula et. al, 2007; Rudy, 2014; RunBin, 2013; Shkolnik, 2007). The COM of a robot can be determined using the equation

$$c = \frac{1}{\sum_{j=1}^{N} m_j} \sum_{j=1}^{N} m_j c_j \tag{1}$$

where m_j is the mass of the jth part of the robot and c_j is the position of the center of mass of the jth part. This type of control takes into account the static forces that the quadruped is under but does not account for the motion of any of the parts, potentially leading to undesired operational situations. One likely occurrence of such a situation is during the crawl gait when moving either of the front feet. At this moment, the support triangle includes both back legs and the opposite front foot, potentially placing the COM close to the edge of the triangle, as seen in Figure 2.11.

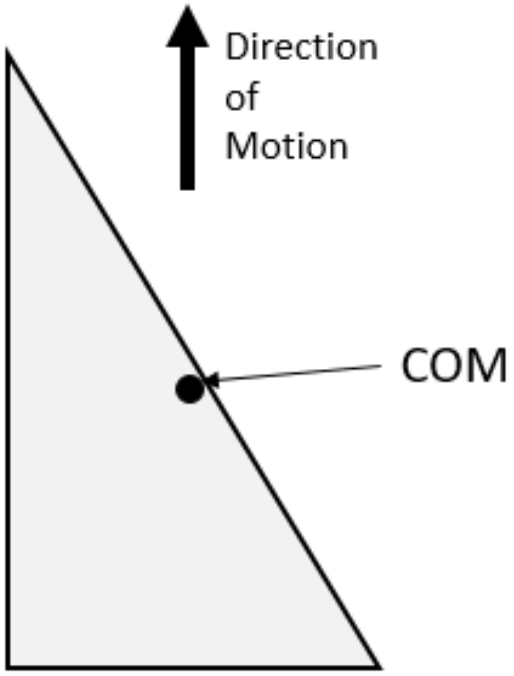


Figure 2.11: Crawl Gait COM Approaching Instability

While statically stable, the inertia of the motion might bring the COM outside of the support polygon and therefore cause the system to become unstable (Chen, 2000). Some researchers have created controllers that deal with this flaw be padding the support polygon based upon the speed the robot is traveling, resulting in the controller throwing out potentially unstable orientations and speeds to ensure stability. This works, but can often lead to a robot being unable to attain its top potential speed. Additionally, constantly calculating the COM can be computationally expensive and often needs to be simplified to run in real-time.

Another method of calculating the stability of a certain body trajectory and orientation is through calculating the ZMP (Dekker, 2007; Rudy, 2014). The ZMP is defined as the point at which the sum of torques upon the robot is equal to zero. Using this definition, the exact point can be calculated using the equations

$$p_x = \frac{Mgc_x + p_z\dot{P}_x - \dot{L}_y}{Mg + \dot{P}_z} \tag{2}$$

$$p_y = \frac{Mgc_y + p_z\dot{P}_y - \dot{L}_x}{Mg + \dot{P}_z} \tag{3}$$

where p is the location of the ZMP, M is the total mass of the system, g is the acceleration due to gravity, $\dot{P}$ is the total linear momentum of the system, and $\dot{L}$ is the total angular momentum of the system. As long as the ZMP is within the support polygon, the robot will be stable. This calculation takes into account the motion of the quadruped as a whole to produce an actual ZMP, but is very complex and requires simplification to be run in real

time. It is also worthy of note that while the robot is static, the ZMP and the projection of the COM onto the support polygon are the same. ZMP calculations can also be done to find a dynamically stable gait for the walk and pace methods by determining how to place the ZMP on the support line created by the two legs on the ground, whereas the padded COM method is not accurate enough without significant experimentation and tuning (Pongas, 2007).

2.3 Actuator Research

Actuation of the quadruped platform's legs determines many of the characteristics of the final design. Many standard actuation methods including series elastic actuators, direct drive actuators, and parallel actuators using both electric and hydraulic components have been previously developed for high performance quadrupeds. Research into various actuation methods has shown many design trade-offs are present with each type of actuation and leg design.

2.3.1 Linear Actuators

Linear actuators provide motion in a straight line. When using linear actuators the driving elements of the system are able to be located closer to the joints of actuator, reducing power requirements, but have issues with the usable sections of the workspace and maneuverability. The linear actuators that this project could use would potentially be electric or hydraulic in nature. Pneumatic actuators were not considered as they require compressed gas and would take a considerable amount of weight and space. Electric linear actuators typically make use of a motor and convert rotary motion into linear motion. This is done with a variety of mechanisms including cams, wheel and axle combinations, and screw configurations. With both hydraulic and electric types there is a concern with the cost of an actuator that can account for the potential required torques of the system. Hydraulic actuators do not always need tanks like pneumatics do, but they can still require components of considerable size and weight such as a hydraulic pump.

2.3.2 Electric Motors

Electric motors are a convenient and common type of actuator used in robotics. However, electric motors are typically high speed and low torque, which is not very suitable for legged robots, which need significant torque at relatively low speeds (compared to what most motors produce). To achieve the required torque, a gear train can be added. This can allow even relatively small motors to produce large torques at ample speeds, but is not without drawbacks. However, the addition of a gear train greatly increases the reflected inertia and hampers the ability to backdrive the system. This can be undesirable for legged robots where it is good to be able to tolerate unexpected disturbances, and the gear train is a component that has the potential to break if exposed to sudden, high torques.

2.3.3 Series Elastic Actuators

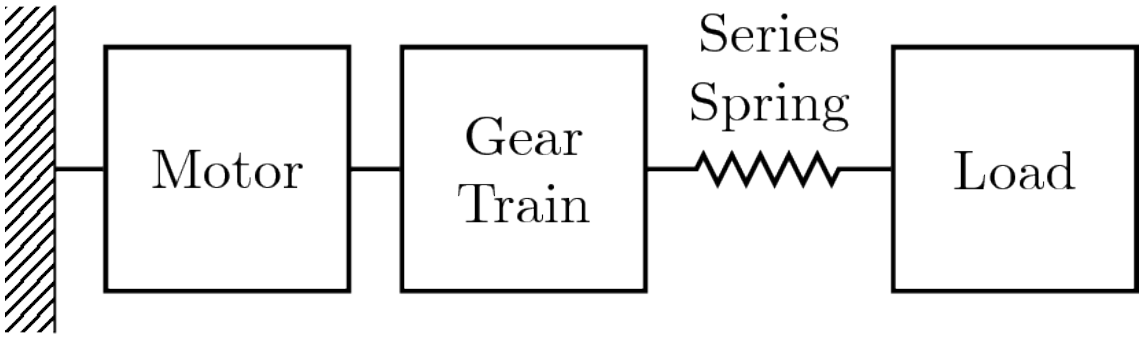


Figure 2.12: General topology of a series elastic actuator

In many actuated systems, it is seen as desirable that the actuator is as rigidly connected as possible to the load it is driving. The idea behind series elastic actuators, however, is that stiffer is not always better. Figure 2.12 depicts a typical setup for a series elastic actuator. The actuator, in this case an electric motor (which has low torque but high speed), drives a gear train, increasing the torque output and decreasing speed. The gear train, rather than being directly connected to the load, is instead connected to the load through a series elastic component such as a spring. Because the load is no longer rigidly connected to the actuator, the achievable bandwidth is reduced. However, in exchange, series elastic actuators introduce a number of advantages.

The addition of an elastic element reduces output impedance and reduces the effect of the inertia and friction of both the motor and gear train. The elastic element also greatly increases the impact tolerance of the system. Another big advantage is that the fidelity and stability of force control greatly improved. This is in part because the force control problem has essentially been reduced to a position control problem: the displacement of the elastic element is proportional to the force exerted by it. Because gear trains typically transmit position with greater fidelity than force, this results in improved force control. Additionally, the elastic element can temporarily store energy; for cyclic actions, energy can be stored during one part of the cycle and released during another, increasing the efficiency of the motion (J. E. Pratt and B. T. Krupp).

Series elastic actuators are not limited to just electric motors. Many of the advantages of adding an elastic element in series also apply to e.g. hydraulic actuators, as hydraulic systems also exhibit high output impedance, and may have poor force fidelity for small forces due to seal friction.

2.4 Sensor Research

The use of sensors are vital to any robot. For the scope of this project two types of sensors were of concern, feedback sensors for the robot's position and orientation, and navigation sensors to sense the robot's environment. Feedback sensors that provide the robot's position and orientation would assist in precise control of the robot's motion as it would allow the robot to know the error between its current state and the desired state. The following

sections describes the sensors that the team expects to use and how they can specifically benefit this project.

2.4.1 Force Sensors

It is important for a quadruped to be able to sense the world around it and to not only rely on proprioceptive sensors to control the robot dynamics. For a more complete image of the quadruped's situation in the execution of a path, exteroceptive sensors are needed on the feet. The feet are the only point where physical contact occurs with the environment regularly and the amount of force experienced at these points can be extremely useful to any control scheme. Usually successful quadrupeds have used F/T (force/torque) sensors to measure the normal force. Others, such as MIT Cheetah, used various sensor topologies including magnets and hall effect sensors or barometric pressure sensors both encased in polymeric material to determine a three dimensional force vector in a lighter and cheaper form factor. A force sensing solution will need to be devised for effective control of the quadruped legs and successful gait execution.

There are many challenges associated with sensing the force on the foot of a quadruped. Located on the end of the manipulator, the mass of the sensing solution has a large impact on the dynamics of the leg. It is necessary to minimize the mass of the sensing solution for the best manipulator performance. Another important factor is the ability of the sensor to survive large and sudden impacts with large momentary forces. As a quadruped walks the amount of force on the legs spikes when the foot contacts the ground and when the leg pushes off of the ground to propel itself forward.

Three main methods of force sensing for robot end effectors exist currently, T/F sensors, load cells, and force sensitive resistors. T/F (torque/force) sensors are complete sensing solutions that measure the forces and torque on the end of robotic manipulators. These sensors are very accurate and are often used in high accuracy and precision applications. While they are very accurate and precise they have a few drawbacks; a major limiting factor is that the T/F sensors are very large and heavy, which is problematic for an end effector where mass should be minimized. Also they have a limited sensing range and are not able to absorb large forces outside of the operating range well. Another drawback is that the sensors are very expensive, making them very difficult to afford for collegiate projects uch as this (M. Y. Chuah and S. Kim).

Another well known force sensing method is using load cells to sense the force on the end of a manipulator. As the cell makes contact with a surface the metal housing deforms slightly and strain gauges embedded in the cell deform with the metal. As the strain gauges deform, their resistance changes, in turn altering the voltage read by a wheatstone bridge circuit, which is usually later amplified to be read by an ADC. This sensing method is usually very accurate for measuring forces and relatively low cost. However load cells have two main problems that make them less ideal. To actually sense a force it needs to be transmitted through the load cell, limiting the possible mounting configurations and not allowing for a distribution of force around the sensing element. Additionally they do not handle sudden impulses very well and are not designed to be exposed to repeated high force impacts that exceed their maximum force threshold. This is problematic for applications such as robotic legs which are designed to purposefully have small periods of extreme force exerted on the

legs when executing different gaits (M. Y. Chuah, M. Estrada and S. Kim).

The other popular force sensing method is to use force sensitive resistors (FSRs). These function very similarly to strain gauges mentioned earlier but on a larger scale and without a metal housing. As pressure is exerted on the sensor, the resistance changes, in turn changing the voltage measured across the resistor when arranged in a sensing circuit (e.g. voltage divider or wheatstone bridge configuration). While these sensors are very cheap, shock tolerant, and lightweight they are not very accurate or precise. They are mostly used to sense qualitatively if a surface is being pushed, not quantitatively how hard it is being pressed. This property limits its usefulness as a foot sensor when quantitative force information is extremely important in leg control applications.

In the field of soft robotics a novel technique for force sensing is used where barometric pressure sensors are embedded in polyurethane (a flexible polymer) to sense the force applied to the sensors. This sensor is able to sense a force when the flexible polymer deforms as a force is applied its outer surface. When the polymer deforms, it acts as a spring which transmits the force to the inner surface. At the interface of the flexible polymer surface with the pressure sensor's membrane, a force is allowed to slightly deform the sensor's membrane. Normally the membrane of the pressure sensor is deformed with air pressure, but in this case the flexible polymer fills this space. The pressure sensor then measures the membrane deflection using a piezoelectric element in a Wheatstone bridge sensing configuration which is then amplified to be read by an ADC. This technique for force sensing is resistant to forces that exceed its sensing range since the barometric pressure sensors are able to withstand very large pressures that far exceed their sensing range. The sensors themselves are not subjected to large forces as the force on the foot's surface is distributed over the entire inner surface while the actual sensor elements have a very low surface area. This sensor is able to provide quantitative data at a reasonable resolution compared to other sensing solutions. This system is also very lightweight as it is primarily made of a thin polymer layer and a PCB. However there are two main drawbacks to this method. First of all, the mechanics underlying the deformation of the flexible polymer are extremely complex and computationally difficult. This requires experimental data collection and design iterations to make a sensor with a desired force sensing range. The other main disadvantage to this system is that extra engineering effort is required to cast the pressure sensors in a flexible polymer and to determine the final sensor configuration (Tenzer, Yaroslav, Leif P. Jentoft, and Robert D. Howe.).

2.4.2 Encoders

To track the angular position, velocity, and acceleration of an axle, shaft, or joint, encoders are often used. There are two main types of encoders. For the scope of this project the benefits of both were researched. The use of encoders is beneficial for this project as they can potentially be used to track the angular position, velocity, and acceleration of each joint on the quadruped with only one sensor per joint. By keeping track of these values at each joint the control of the quadruped would be much more efficient.

2.4.2.1 Incremental Encoders

Incremental encoders are typically optical and make use of a black and white segmented disk. A light detection sensor within the encoder has an output that changes based on when

it detects a black or white segment. The angular position is measured in ticks, or the number of times the light detection sensor measures a change. Single disc encoders are only capable of measuring displacement while quadrature encoders are capable of measuring displacement and direction of motion. This is because in quadrature encoders there are two tracks offset by some angle that are both read to produce outputs. The direction of rotation can be determined in quadrature encoders by measuring the phase difference of the two tracks. Because incremental encoders only measures changes in position, some form of indexing or calibration is required to achieve absolute position measurements.

2.4.2.2 Absolute Encoders

Unlike incremental encoders, absolute encoders can have a unique output for each distinct shaft angle. The design of an absolute encoder can vary (i.e optical or magnetic), but the principle is always the same. Typically there are two discs, both with concentric rings and offset markers. One disc is typically fixed to the central shaft while the other moves freely. When the disc turns, the markers along the track of absolute encoders change position on the fixed disc. Each configuration represents a unique binary code and interpreting that configuration determines the absolute position of the encoder. For optical absolute encoders, the marker is an opening which lets through light. In magnetic absolute encoders, the markers are a magnetic sensor array that passes over a magnet and detects the position of the magnetic poles. Because of their construction, absolute encoders generally have a higher resolution and orientation than incremental encoders.

2.4.3 Potentiometers

Potentiometers are variable resistance devices that typically have a shaft going through the center of the. The resistance of the potentiometer is determined by the rotational position of that shaft. If a potentiometer is attached to a joint of a robot it can determine the angle between the two mechanical links that joint is connected. This is because the change in angle at that joint will correlate to the change in resistance of the potentiometer. While potentiometers are widely used they have problems. This includes being susceptible to mechanical wear, sensitive to voltage changes, and having a limited rotation range.

2.4.4 LIDAR

A LIDAR (Light Detection and Ranging) is an optical sensor that can be used to measure distances between itself and objects within its range. It does so by sending out a pulsed laser and measuring the amount of time it takes for a specific pulse to return to the LIDAR. It does this by using a laser scanner to provide two dimensional distance data. A photo detector is also necessary to measure when a light pulse returns to the LIDAR. For exploration and navigation tasks, there are LIDARs that continually rotate. This is advantageous as an entire set of distance values from a single plane can be generated. This set of distance values can be interpreted to produce maps of obstacles and the general environment around a LIDAR. For this MQP, a LIDAR could be used for path planning by mapping areas unknown to the quadruped.

2.4.5 Camera

Cameras are widely used in the robotics field for tasks involving object detection and recognition. There is a large amount of software support available for the use of computer vision systems. For this project computer vision systems can be used to detect and avoid obstacles to assist with the navigation of unknown environments and path planning.

2.4.6 IMU

An Inertial Measurement Unit (IMU) typically makes use of an accelerometer, gyroscope, and a magnetometer to interpret the forces acting upon the sensor and the orientation of the sensor with respect to the Earth's magnetic field and center. The number of axes for the accelerometer may vary, but measurement of velocity and orientation remains the same for all IMUs. Using an IMU would be beneficial for this project as the orientation of the robot in terms of pitch, roll, yaw and acceleration can be interpreted. It is important to note that it is common for IMUs to accumulate error over time which can cause problems over long run times. The errors accumulate due to the way IMUs continually add detected changes to previously calculated values. This makes it so errors are blindly added to the values every time step. There are methods to overcome this problem such as the use of additional sensors or a Kalman filter so it is still possible to use IMUs.

2.5 Simulation Research

An important aspect of this project is simulation of the robotics system as it will facilitate faster iteration on the designs and allow experimentation before the team produces physical prototypes. There are many existing projects currently used for robotics simulations; this includes software such as Gazebo, V-Rep, and Webots. Some teams choose instead to create their own simulation tools, often based on existing physics engines such as ODE or Bullet (an example of this is described by Belter et al.). This approach affords more control over the simulation, but is also much more work than using an off-the-shelf simulation platform such as Gazebo (which, for instance provides features such as the simulation of sensors that would not be present in a physics engine).

When choosing which software to use for these simulations it is important to consider the advantages and disadvantages of each. Because of the limited time for this project, it does not make much sense for the team to develop their own simulation software and tooling on top of a physics engine. Therefore, we considered only complete system simulators. A survey by Ivaldi et al. studied which simulation tools were being used, and what the users thought of them. While V-Rep received the best user ratings in the survey, Gazebo was used by the largest percentage of participants. Ivaldi et al. admit that different projects may have different requirements in terms of simulation (accuracy vs. performance, for example) and suggest that it may be the more modular nature of Gazebo and V-Rep that make them more flexible and suitable for a wider range of projects that accounts for the popularity.

3 Methodology

3.1 Task Specifications

In order to more fully and concretely define a problem statement the team first came up with a set of design requirements for the system. These requirements were very focused and measurable to allow us to gauge the success of the project. The following are the defined goals of the project:

- Entire system weighs under 30kg
- The platform can support up to a 10kg payload
  - System is capable of moving a payload from one location to another
- Entire system fits within a 1x1x1 meter cube
- Robust mechanical base for general use
- Each leg will have three degrees of freedom:
  - Hip Joint- Theta 1
    - This joint adds an extra degree of freedom to the 2-link leg for an additional motion capability and control.
  - Shoulder Joint-Theta 2
    - The top joint of each 2-link leg
  - Knee Joint- Theta 3
    - The joint between both links of the leg
- System is mechanically designed to be physically capable of standing and different gait motions
  - If system is mechanically complete it should minimally achieve crawl gait
  - System can achieve a minimum speed of 0.5 m/s in a walk gait
- System is untethered when in use and has a on-board power supply
- System has modular functionality that would allow for:
  - Attachment of additional mechanisms and sensors to compliment the robot
  - The testing and experimentation of different leg designs
- Complete 3-D models of all mechanical aspects of the robot
- Develop software for the following purposes:
  - Motion control of robot

- General navigation and path planning
- Users can wirelessly connect to the robot

The team decided to separate these goals into realistic and stretch goals. The realistic goals are prioritized above the reach goals and attempts at the reach goals will not be made until the realistic goals have been completed. In most cases this is because the realistic goals are required for a reach goal.

Our team decided to determine the Torque requirements of each joint before tackling other goals. Once these values are determined, actuators that meet requirements can be found. With the selected actuators a CAD model of the quadrupedal robotics platform will be developed. This CAD model will be an assembly of all of the required parts. Within this assembly there will be sub-assemblies of specific aspects of the robot. This will allow us to do simulations and analysis on the model that will allow us to determine design improvements and motion control.

The CAD model will be put into custom simulations that the team will create. These simulations will test the motion and control of the quadruped so the gait controllers can be improved in an easily modifiable manner. This method will not risk damage to any machined pieces or subsystems. This also allows the team to work on the controllers without having a physical robot and to easily see the effects of any design changes. Using the CAD model, the team will manufacture and assemble the robot. This will likely be done in pieces with major subsystems, such as a leg, being built and tested multiple times before the final design. This allows the team to iterate on the design and find problems that do not appear in simulations as well as to ensure that all of the subsystems of the quadruped are working and meet the desired design parameters. All realistic goals regarding the mechanical design of the robot should be attainable for the scope of this project.

Once the design is finalized, the team will fine-tune the controllers to allow the quadruped to balance with assistance and eventually without any external support as well as do simple motions like squat or shift its upper body without moving any legs from the ground. These activities require the controllers to be fully tuned and operational and are a necessary step towards walking.

The last of the team's realistic goals is to achieve a simple crawl gait on a flat surface. This is a major increase in complexity from standing, however not overwhelming enough that the team believes it is not attainable. The team will start with the supported crawl gait to avoid possible damage to the robot and its components, but believes that it is also possible to attain an unsupported crawl gait.

The team's reach goals are goals that they hope to be able to accomplish but do not know if is realistic to attain them in the limited time period of this project. These reach goals include implementing an unsupported walking gait on a flat surface which requires two legs to be off of the ground at once, therefore creating a statically unstable situation, a task that is very complicated to properly control. An even further reach goal is to incorporate mapping and navigation of a flat environment. It is highly unlikely that the team will reach this point due to the requirements of vision and mapping software and the equipment necessary.

3.1.1 Work Schedule

Because this project is still ongoing, any future scheduling is a plan. As time goes on, this section will be updated with what happened but it should be fairly similar to the initial plan.

The team spent time over the summer and the beginning of A-Term to do research and information gathering. This include familiarization with past MQP projects that relate to the topic of quadruped platforms as well as researching existing platforms currently that have been developed or are currently being developed in the robotics industry. This gave us an understanding of the benefits and shortcomings of different quadruped designs so that the team could approach this project in the most efficient way possible and avoid the same problems that other designs have faced.

Over the course of A-term, the team focused on determining the feasibility and requirements of the quadrupedal platform through calculations, simulations, and research. This constituted looking at the exact design parameters of other quadrupedal platforms, running dynamic, torque, and speed analyses outlined in Sections 4.2.1 through 4.2.3. Using the results of these analyses, the team selected motors that would fulfill our needs and create an initial mechanical design of the robot. Additionally, the team created the high level designs for the system architecture, controllers, and construction method of the robot (described in Sections 4.3, 4.4, and 4.1 respectively) to ensure that all components would work together. The team reached out to potential sponsors for different components of the robot with the goal of receiving discounts or partnerships to allow us to acquire the desired components. We designed for mostly off-the-shelf products to allow for lower lead times, something that often causes a problem with past MQPs, which forced us to make some sacrifices. The main deliverables for A-Term were a draft of the MQP report that described the results of the team's research and the initial design gathering as well as preparation for the Preliminary Design Review (PDR) in the beginning of B-Term.

In B-term, the team's main goals were to complete the full design of the robot, from a complete SOLIDWORKS rendering to all of our PCB and battery specs. We also wanted to create a proof of concept leg prototype, but ran into a problem with the delivery time of the desired motors. In order to create a full scale prototype before our final motors came in, the team had to go back to the drawing board and create all of the leg components out of lightweight materials (wood and 3D printed parts) so that we could use REV Robotics motors that had significantly less torque. Once the leg design was redone, the initial prototype was built. We used the initial prototype to fix design issues with the series elastic actuators and some issues we found with clearances. The manufacturing of the prototype was very useful for us to see where we would run into problems in the implementation stages of our design and to learn from them when we moved to the final product. The prototype was also used as a test-bed for the controllers in B-term, but most of that work will take place in C-term while the rest of the final design pieces come in and are assembled. Towards the end of the term, the team also ordered the required materials to create the final robot during C-term.

The main goals of C-term are to improve upon the design based upon the results of our initial testing and complete the testing of the control systems. This would entail testing of the combination of the control and mechanical systems for each leg as well as testing the combined motion of the individual legs to create planned motion of the entire platform

(a.k.a. walking). The mechanical construction of the robot should be completed within this term so physical testing can begin.

These tests may carry into D-term, but if there is enough time the team hopes to implement a navigation aspects of the project using off-the-shelf technology. The navigation portion of the project is a reach goal and the team hopes to get to it during D-term, however it is possible that D-term will be spent making the robot capable of walking and standing unsupported.

4 Design

4.1 Mechanical Design

For this project the team used SOLIDWORKS for the design of all mechanical components of the robot. When designing parts, the team made sure that it feasible to manufacture parts with the resources available. Over the course of the project, the identified manufacturing resources available are access to a waterjet cutter provided by Hydrocutter, a sponsor of this project, and the CNC machinery of WPI's Washburn Shop (a HAAS 4-axis Mini Mills, a 5-axis VM2, and an ST-30 Lathe). The team's mechanical design of the platform kept these resources in mind to avoid any difficulties that may occur if the design was created without regulation.

The choice of the size and dimensions of the team's quadrupedal platform is driven by different elements. One of the most important requirements the team had was that the robot can be safely operated by two people without a lot of extra equipment. This is so the robot could be easily used without much setup. At the same time, the robot has to be large and powerful enough to carry the desired sensor equipment and on-board power with the possibility of future expansion. These contradictory goals, in combination with our desire to use commercial off-the-shelf components for the motor and electronic components, impose tight size and weight restrictions. This is what led the team to decide for the entire platform to weight under 30kg and fit within a 1x1x1 meter cube so that it would be feasible to manually transport the platform. The team also decided that the system should be designed to attain a top speed of .5 meters per second with a walk gait to allow future projects to analyze higher speed motions.

4.1.1 Body Design

The team decided that the motor and gearbox for θ_1 would be placed on the body itself while the motors and gearboxes for θ_2 and θ_3 would be mounted as high up on the leg as possible as seen in Figure 4.1 below.

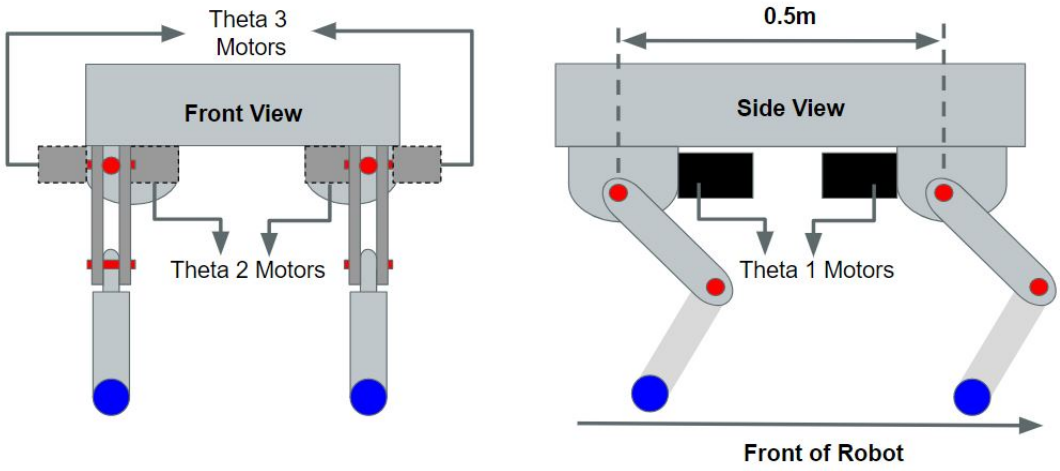


Figure 4.1: Diagram for Initial Body Design Plan

These decisions were made to avoid any drawbacks that come with additional mass being attached to the leg, such as increased inertia and the additional torque that is required to move it. The dimensions of the body were determined by dimensions of the legs to prevent any collisions during a normal crawl gait and to allow the legs to rotate 10 degrees towards center of the platform without colliding into each other. The width was slightly increased beyond this point to improve the stability during the crawl gaits despite that necessitating a larger side-to-side motion during the walk gait to maintain stability. The team was okay with this sacrifice due to us focusing on the crawl gait. Additionally during the design process of the body, space for electrical components such as batteries and controller boards was considered.

For the actual construction of the body, the team decided to go with 1x1 inch hollow aluminum tubing supported by eighth inch aluminum plate. To save weight, the aluminum plates had portions removed while ensuring it retained its structure. Since the team has access to a water jet cutter and CNC machinery, we felt that both materials would be easy to manufacture. The use of rectangular tubing connected by gussets would also allow the team to easily change the length of the robot if that became necessary. The final design for the body can be seen in Figure 4.2.

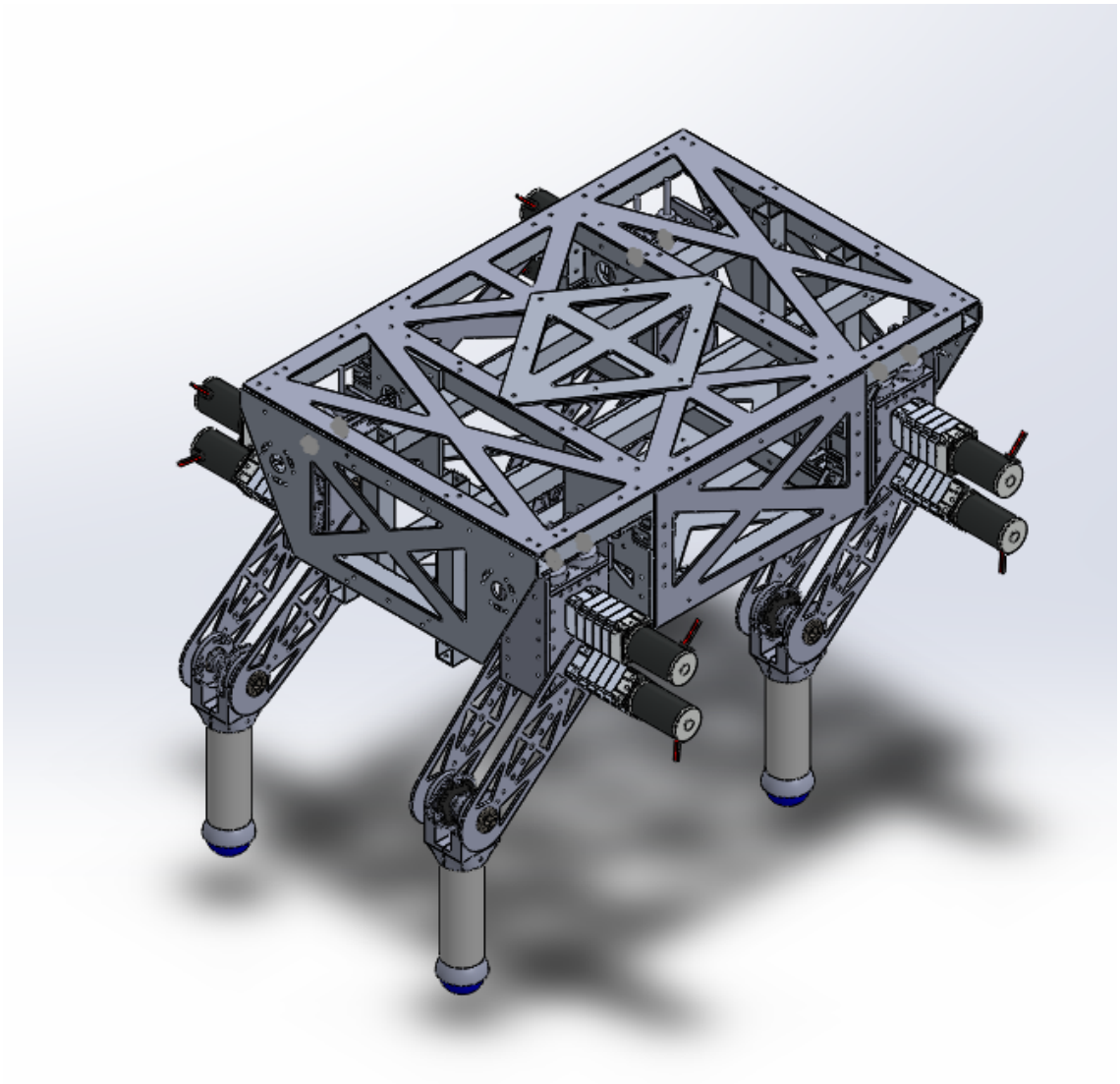


Figure 4.2: Final Body Cad Isometric View

4.1.2 Leg Design

After researching other quadrupedal platforms the team decided that the legs of the platform should have 3 degrees of freedom. This choice would allow for the platform to have a larger range of motion, making it easier to traverse different terrains. The team also decided to use motors to actuate each joint rather than linear actuators or hydraulics due to the weight and space requirements that the other actuation systems bring. This choice would also increase the platform's mobility and allow experimentation of more gaits due to the motion of a motor not being restricted by the design of the actuator itself like that of the other two options. However, because motors were chosen, direct drive of θ_3 would cause the weight of the motor and gearbox to heavily affect the rotational inertia of the leg. To limit this, the team decided to make the θ_3 joint chain driven with the motor located by the upper joint so that it does not significantly contribute to the inertia of the leg. The biggest downside to this design choice is it increases the width of the leg by requiring the use of

sprockets, therefore increasing the inertia of the leg.

Originally, the team decided to place the motors for the θ_2 and θ_3 in the coaxial configuration seen in Figure 4.1. This is the ideal placement for the motors due to them both rotating around the axis of rotation of the upper link as the leg moves. This significantly decreases the rotational inertia of the leg as the motors and gearboxes are a significant portion of the mass of the leg. However, as the leg design progressed, the coaxial design led to complications with supporting the axles and the team being concerned by the potential for deformation of those axles leading to increased slop in the system. This led to the design being modified to the motor for θ_3 being mounted above the motor for θ_2 with an idler sprocket being mounted on the shaft for θ_2 to run the drive train to the lower link. This increases both the size and the inertia of the leg, but the team felt like it was a needed design choice. The final design of the leg can be seen in Figure 4.3.

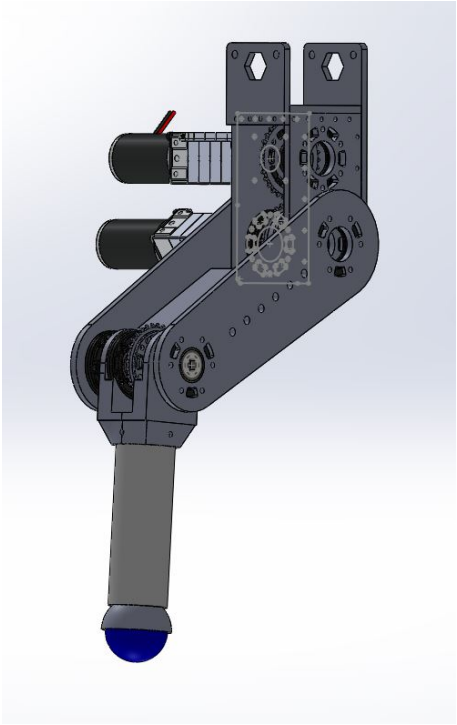


Figure 4.3: Final Leg Cad Isometric View

4.1.3 Foot Design

The team also decided for each foot to be a spherical-like shape to constantly have only one point of contact with the ground per leg. This simplifies all associated calculations and control-schemes as each foot will be considered a point rather than a surface of a certain area. If the latter method was chosen, it would have made the stability calculations of the platform much more difficult.

In order to sense the forces experienced by the foot, a force sensor on the end of each leg was required. The sensor needed to be able to withstand large dynamic forces, fit in a constrained budget, and have as little mass as possible to reduce the torque requirements on

the leg motors. Due to these design requirements, a novel foot force sensor with barometric pressure sensors was designed for the quadruped. Specifically the LPS25HB barometric pressure sensor, shown in Figure 4.4 below, was chosen for its large and unprotected sensing membrane compared to to other comparable barometric pressure sensors.

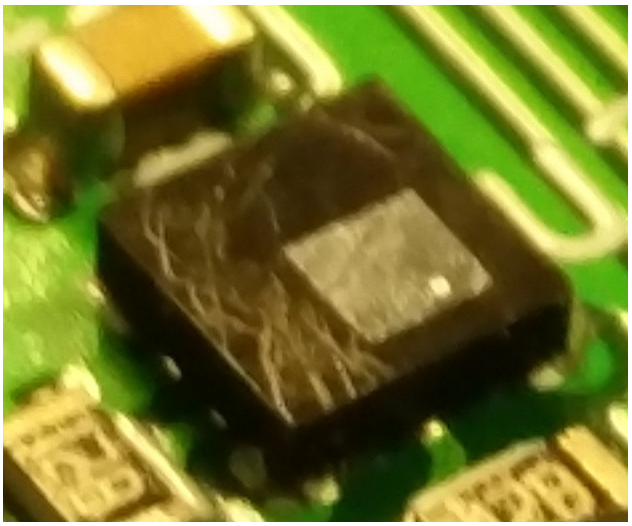


Figure 4.4: Barometric pressure sensor, the membrane is the shiny grey square

To sense the force on the bottom of the foot, the barometric pressure sensors should be placed on a PCB in an array. By using an array of sensors, the foot would record more data, allowing for more precise calculations, and also allowed for the calculation of components of the force vector not normal to the foot sensor's surface. It was determined that the largest and most dense array would be desirable, but the budget constrained the size of the array to be a 3×3 grid.

Communicating with all of the sensors in the foot sensor array required a digital communication bus. Due the relatively slow update rate of the LPS25HB sensors and the number of sensors in the array, I2C was chosen as the communication bus since it could meet timing requirements, transit over relatively long distances, and not need additional control pins for each sensor. However the LPS25HB sensor only had 2 possible I2C addresses so an I2C switch was required to resolve address conflicts. The TCA9548A I2C switch, shown below in Figure 4.5, was chosen because of its wide spread use, availability, cost, and total number of sub-channels. The VQFN package was used because the TSSOP package took up too much space.

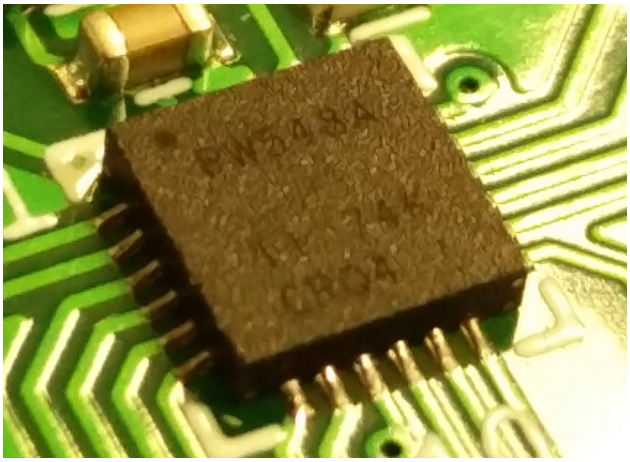


Figure 4.5: I2C switch, VQFN package

The array of pressure sensors, I2C switch, and all other necessary electronics then needed to be mounted on a PCB. Kicad was chosen as the software to design the foot sensor because it was open source, easy to learn, and was a complete electronics design package. First, a schematic of the foot sensor was created to form all of the electrical connections for the PCB, shown below in Figure 4.6.

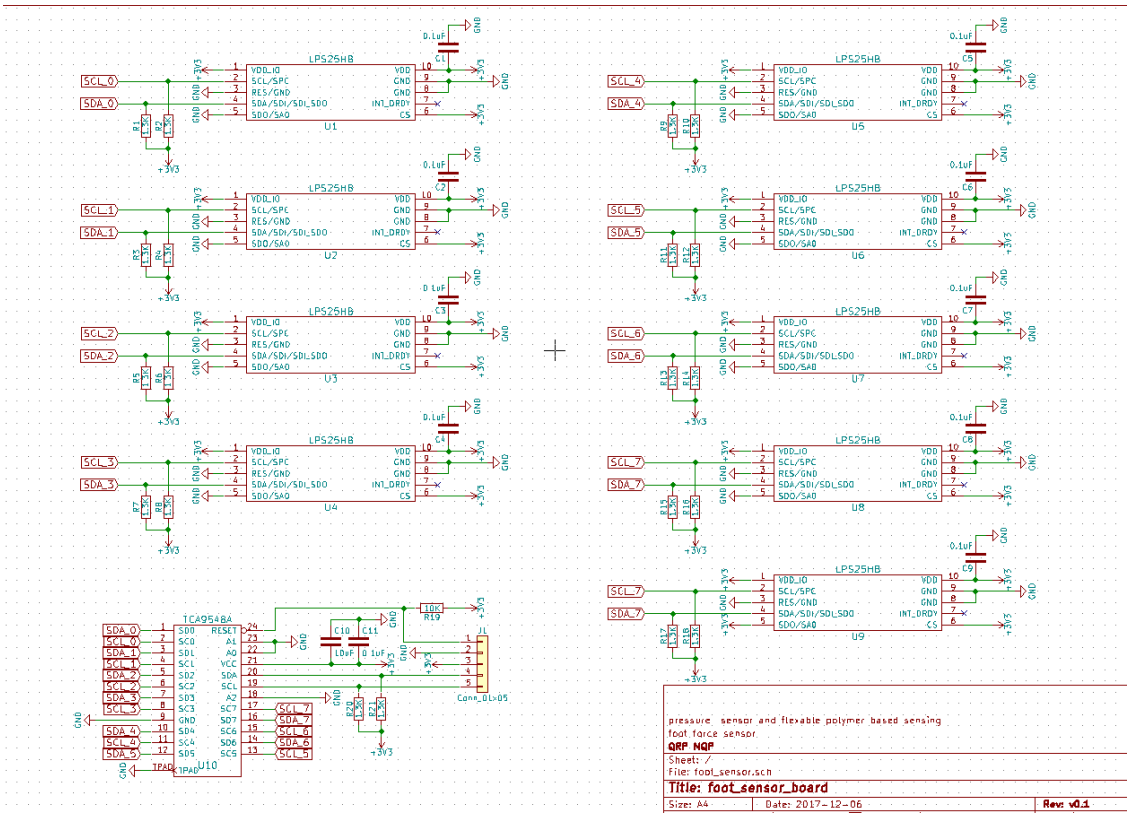


Figure 4.6: Foot sensor schematic

After the schematic was designed, the spacial layout and routing of the traces need to specified. The board editor was used to make a 2-layer PCB, shown in Figure 4.7.

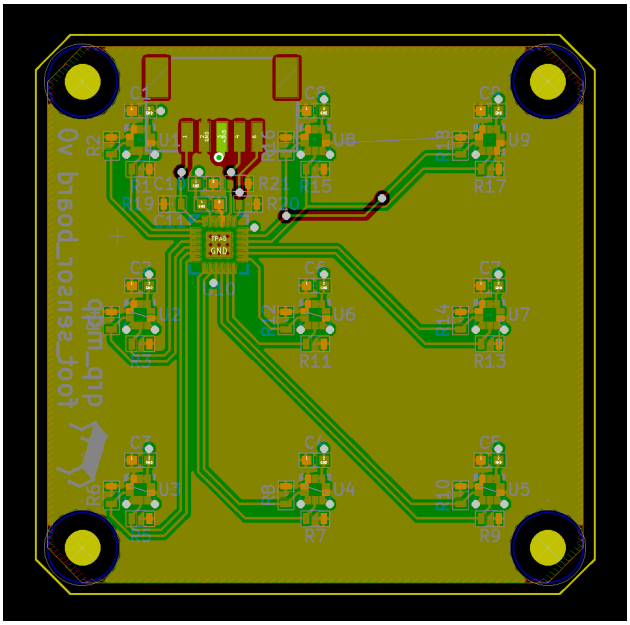
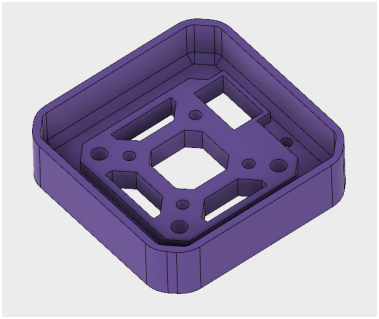


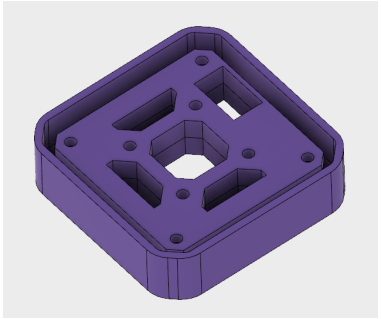
Figure 4.7: Foot sensor PCB board design

After the PCB was designed, it needed to be completely encased in a flexible but strong material. This design allows the deflections in the encasing material caused by the external forces to be transmitted through the material to the interface between the pressure sensor and the material. At this location, the deflection of the material also deflects the membrane of the pressure sensor, causing the pressure measurement to change and provide a reading proportional to the external force applied. Because the casting material needs to hold the shape of the foot while also deflecting enough for the sensors to obtain readable values, a urethane polymer known as Vytaflex was chosen due to its desirable material properties and the ability to order many different shore hardnesses of the same material.

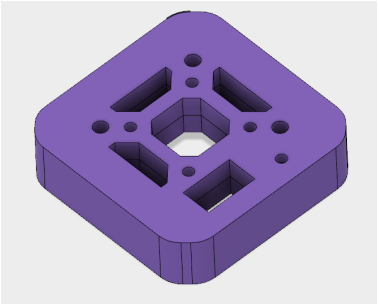
The preliminary CAD design of the foot testing prototype and final shape are seen below in Figure 4.8. The foot was designed so that the it would be machinable, withstand side to side forces (not normal to the foot), and able to be cast with available materials. Resistance to side forces was added by designing a cavity beneath the lip of the PCB to be filled in by the polymer. The ability to machine the foot as one piece was desired to increase its strength, but to accommodate the cavity for side to side force resistance, the PCB holder had to be made as a separate part. To mold the outer shape of the foot, a removable top piece was designed to slide on to hold the urethane in place until it was cured.



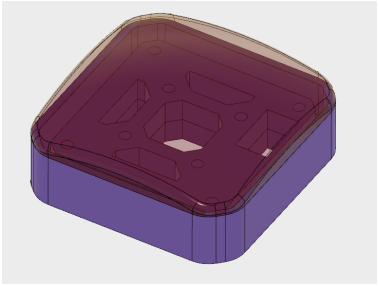
(a) leg mounting and bottom half of cavity



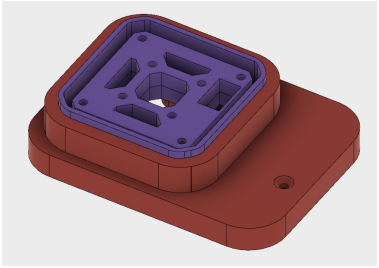
(b) pcb mount and top half of cavity



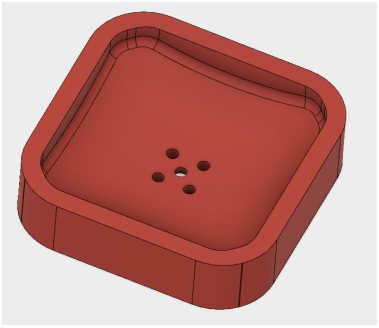
(c) back of leg mounting area



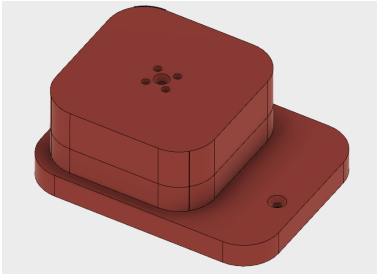
(d) spherical urethane cast section (translucent)



(e) alignment holder and urethane injection channel



(f) spherical mold top piece



(g) complete mold assembly (all)

Figure 4.8: Foot sensor CAD components

The initial calibration and proof of concept versions of the sensor had flat casts to help validate the sensor model design.

4.2 Modeling and Simulation

In order to verify the feasibility and requirements of the robot, the team modeled and simulated specific aspects of the design.

4.2.1 Dynamic Analysis

MATLAB is a numerical computing environment that provides powerful matrix manipulation, data plotting, algorithm implementation, and graphical interface capabilities. The

software is intended to be used for numerical computation because it visualizes and declares everything as a matrix. It is widely used due to its abilities to manipulate large quantities of data and visualize them, being used in industries from economics to engineering as well as research in academic institutions.

For these reasons, MATLAB was chosen to create simulations of different aspects of the robot. Use of the software was critical to determining the required torques for our desired weights and speeds. We created homogeneous matrices using MATLAB to determine the location of each link on the leg and plot the distance between links. Using these we could create graphs of different leg positions for visualization purposes by specifying different angles of the joints and lengths of the leg. Our team expanded upon this by applying force vectors to different parts of the leg to determine static joint torques, which is expanded on in Section 4.2.2.

To determine the dynamic leg joint torques, a more complicated method was needed than static MATLAB analysis. The method we decided on was a dynamic analysis for a serial robotic manipulator. When performing this analysis we used Maple to assist in complex computations. We found the forward position kinematics of each joint position and took derivatives to determine the velocity kinematics. These equations allowed us to determine the potential and kinetic energies and form the Lagrangian equation. Various derivatives of this equation let us find the torques on each joint given leg lengths, joint positions, joint velocities, and joint accelerations.

4.2.2 Torque Analysis

As described in Section 4.2.1, we were able to calculate the static torques required for each leg joint. These calculations only took into account the position of the legs and the force upon the foot, not the weight nor the motion of the leg links themselves. The force upon the foot was calculated as 75 percent of the weight of the entire fully loaded robot (75 percent was chosen due to this being attainable in a walk gait with most of the weight developed towards one side). This calculation threw out any leg orientation where the end of the lower link was above the position of the third joint of the leg, an unattainable position on flat ground. This was used to find the maximum torques that each joint would feel.

This method has its drawbacks in that it did not take into account the weight of the legs nor the inertia of the leg links during motion. However, due to not having a full design of the leg at the time of these calculations, we decided to deal with this by including a 1.5 times multiplier on the required torques when we chose our motors and gearboxes. We also believe that adding the multiplier will allow for the motors to handle the dynamic loading of the walk gait.

4.2.3 Speed Analysis

Besides the required maximum torques of the motors, the other characteristic we needed to know was how fast the leg links should to rotate. In order to solve this, we used the parameters of a quadruped's gait defined in Section 2.2 and the inverse kinematics from the dynamic analysis to find the required angular velocities of the joints based upon the leg lengths of the quadruped.

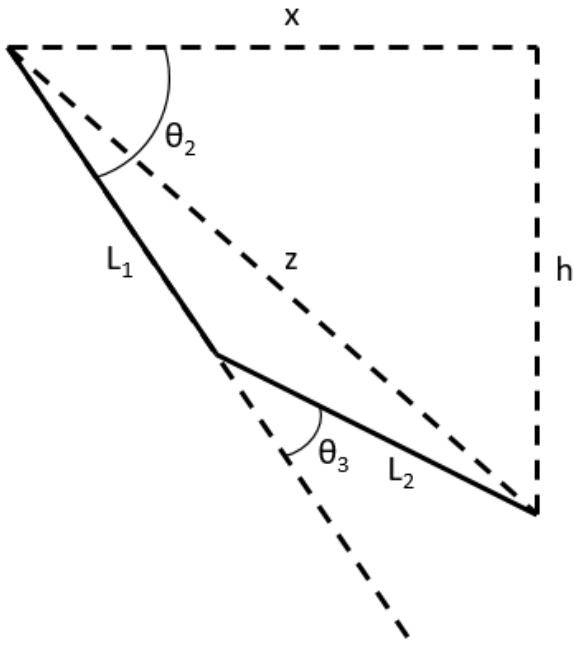


Figure 4.9: Diagram of the position of a leg upon plant

To calculate the speed, we used the geometry of the leg position at the time that the leg touches and lifts off the ground (the two extremes of its motion during a cycle), seen below in Equations 4 to 10 to generate the equations

$$z = \sqrt{L_1^2 + L_2^2 - 2 * L_1 * L_2 * cos(\pi - \theta_3)} \tag{4}$$

$$\theta_2 = sin^{-1}\left(\frac{h}{z}\right) + sin^{-1}\left(\frac{L_3 sin(\pi - \theta_3)}{z}\right) \tag{5}$$

$$x_{plant} = L_2 cos(\theta_{2_{plant}}) + L_3 cos(\theta_{2_{plant}} + \theta_{3_{plant}}) \tag{6}$$

$$x_{lift} = L_2 cos(\theta_{2_{lift}}) + L_3 cos(\theta_{2_{lift}} + \theta_{3_{lift}}) \tag{7}$$

where we define h, $\theta_{3_{plant}}$, $\theta_{3_{lift}}$, L_2, and L_3.

These locations can be used to calculate the stroke of the leg, R,

$$R = x_{plant} - x_{lift} \tag{8}$$

which defines the stride, γ, for a known duty cycle, β,

$$\gamma = R/\beta \tag{9}$$

which in turn is used to calculate speed of a quadruped, v, for a known period, T,

$$v = \gamma/T \tag{10}$$

4.2.4 Leg Length Selection

This was paired with an analysis that took in a determined shoulder height of the hip joint and calculated the possible link length combinations. For each these leg lengths, the maximum attainable torques and speed were calculated using the aforementioned calculations. This data were combined to create an average of the joint torques to speed ratio

$$\left(\frac{\frac{\tau_{2max}+\tau_{3max}}{2}}{v}\right) \tag{11}$$

seen in Figure 4.10, for use in determining the best leg lengths for the speeds we want to attain.

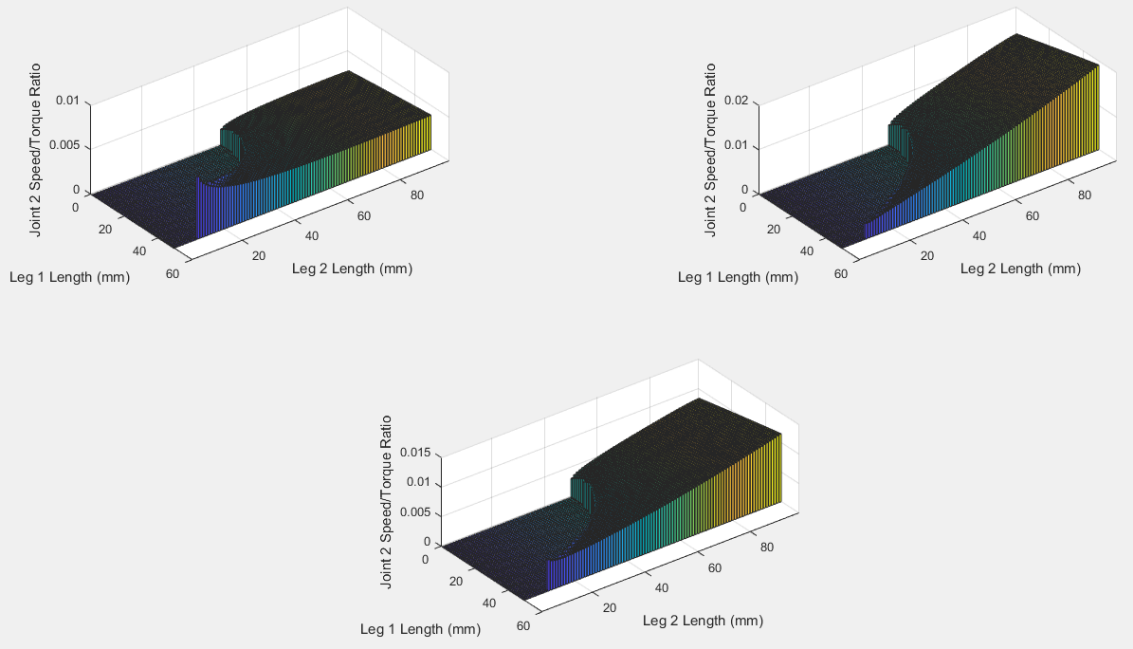


Figure 4.10: Torque-Speed Ratio Analysis (h=.5m, β=.75, T=4s)

The results of this analysis were not what we desired. As can be seen in Figure 4.10, the analysis heavily favors the higher speeds rather than the lower torques. This did not work for us because we could not afford incredibly strong motors to support the higher torques required for the speeds. Additionally, the maximum speed we needed for a walking gait is only .5 m/s so a significantly higher speed is not necessary nor helpful.

In order to discount higher torques more, we decided to use the average of the torques squared

$$\left(\frac{\tau_{2max}^2+\tau_{3max}^2}{2}\right) \tag{12}$$

as the comparison point for each leg combination. To further sort through this analysis, we discounted any combination that did not have a max speed close to the desired .5 m/s. An

Table 1: Leg Angle and Length Analysis

Lift and Plant Angles	-20,-40	-25,-40	-30,-40	-35,-45	-40,-45	-45,-45	-50,-50
Speed (m/s)	.52196	.50152	.4929	.53837	.51584	.48848	.60165
Joint Lengths (m)	.20, .14	.20, .14	.20, .14	.20, .15	.20, .15	.20, .15	.20, .17
Joint 2 Torque (Nm)	43.83	41.45	39.56	43.23	39.92	35.91	44.22
Joint 2 RPM	-18.33	-17.32	-16.22	-17.48	-16.20	-14.77	-17.75
Joint 3 Torque (Nm)	25.46	26.55	27.47	31.83	32.42	32.78	40.33
Joint 2 Speed/Torque	.0119	.0121	.0125	.0125	.0129	.0136	.0136
Joint 3 Speed/Torque	.0205	.0192	.0179	.0169	.0159	.0149	.0149
Avg Sqr Speed/Torque	.000281	.000257	.000239	.000221	.000210	.000204	.000204
Avg Sqr Torque	1284.55	1232.32	1159.93	1440.98	1322.59	1182.19	1791.15

excerpt of such an analysis is seen below in Table 1.

This analysis was run for shoulder heights of .2m through .5m in .05m increments and it was found that while the maximum speed increased significantly with the increase in shoulder height, so did the torques. Additionally, the shoulder heights under .3 m could not attain the speeds that we desired. Due to us needing to go for lower torques, we chose to design for leg lengths of .3m and .4m, which we found produced optimal leg length combinations. With this analysis, we found are the optimal link lengths were .25m and .09m at a .3m shoulder height and .35m and .08m at a .4m shoulder height.

We took these numbers and moved onto the leg design, but later found that series elastic actuation design for the lower link required a minimum leg length of .1m. So we redid the calculations, the actual output of which is displayed in Table 1, to throw out anything with a lower leg length of less than .1m which resulted in link lengths of .2m and .15m at a .3m shoulder height.

The team's initial plan for the design of the diagram can be seen in Figure 4.11 below which does not show the later changes mentioned in Section 4.1.2. It is important to note that this diagram is only illustrating θ_2 and θ_3 as θ_1 rotates along a different plane.

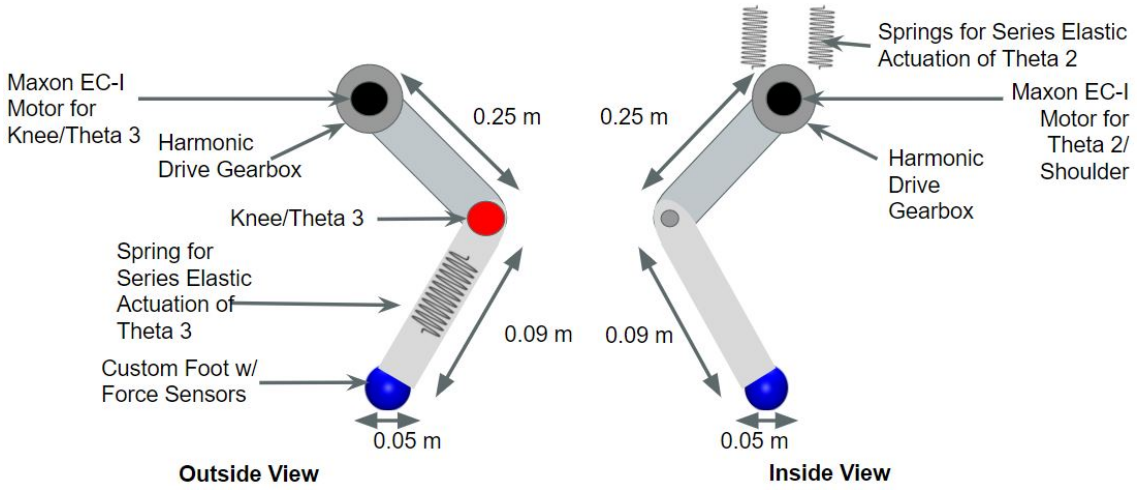


Figure 4.11: Diagram for Leg Design Plan

This design was then modeled in SOLIDWORKS to design the actual parts that needed to be machined and to check whether the motion of the system was as expected. The exact design went through many iterations with minor improvements to fix clearances. The only major item that changed over the course of the design iterations was the width of the leg, but the leg lengths described earlier were maintained.

4.2.5 Series Elastic Actuation Design

Figure 4.11 also illustrates the team's intention to utilize series elastic actuation for θ_2 and θ_3. This is important as it will give the robot a certain degree of compliance when walking which helps to protect the motors and gearboxes from some of the harmful effects of the expected dynamic loading.

How this series elastic actuation will be implemented is different for each rotational joint. For θ_2, the motor will be mounted to the plate of the upper link to rotate the link around the drive shaft, rather than the usual grounded motor connected to the drive shaft which is rigidly attached to the link. While this is a significantly less efficient way to drive the link, it was the simplest way the team could devise to decouple the motor's rotation from the movement of the spring, a critical component of SEA. For the actual elastic portion of the design, a sprocket will be attached to the drive shaft with a length of chain around it running through a structural plate and through two springs mounted above it. A bolt will then be run through the spring, tightened with a nut and washer, and attached to the chain so that when the chain is pulled on one side, the chain will compress one of the springs. This provides the desired compliance where the link can rotate slightly without having to back-drive the motor. The final implementation of the SEA for the upper link can be seen in Figure 4.12.

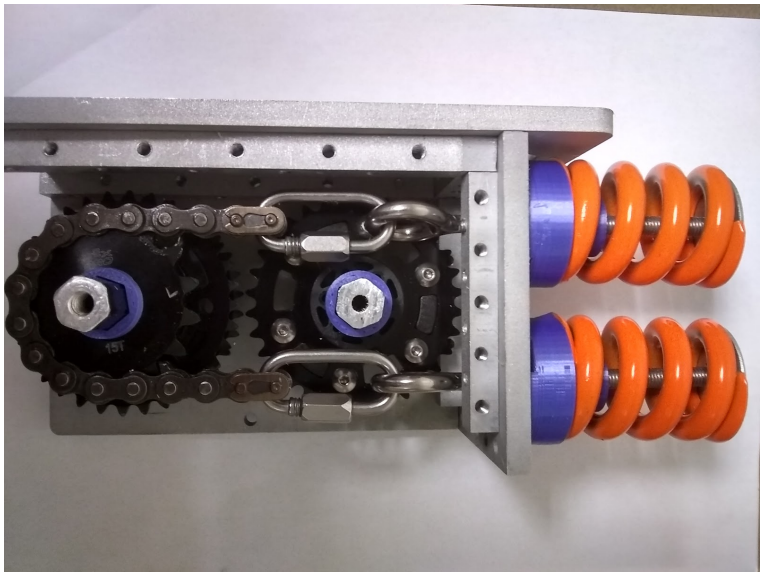


Figure 4.12: Implementation of Upper Link SEA

For θ_3, both the drive sprocket and the lower link are designed to rotate freely around the shaft, which causes the link to be pulled in the desired direction by the motor's interactions with the spring. The same sprocket, bolt, and chain system used for the $theta_2$ joint will be used here, but in this design both sides of the chain will be pulled through the same spring due to limited space in the lower link. Although the spring cannot be seen in Figure 4.13 the implementation can be seen.

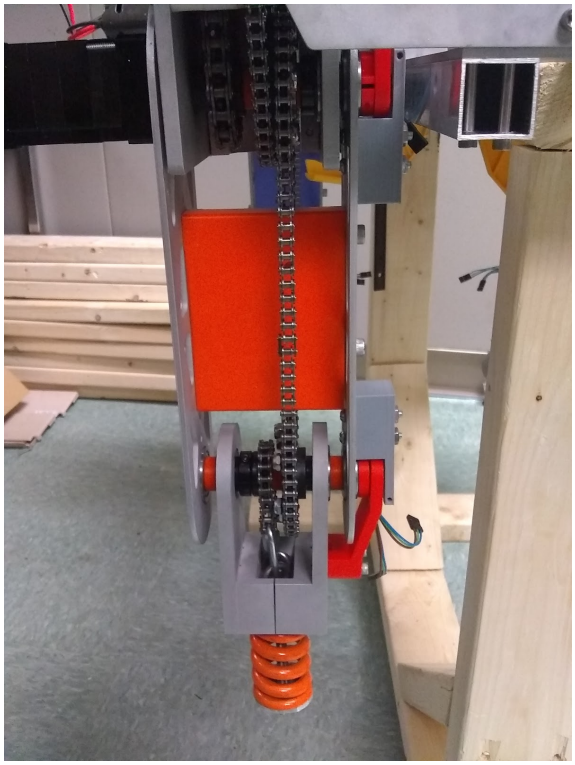


Figure 4.13: Implementation of Lower Link SEA

Both of these designs were heavily based on the design of StarlETH explained in their paper and modified to work with our design scheme.

The most important part of the design of the SEA is the selection of the springs themselves. If the spring if too stiff, it will not provide the desired compliance, but if it is too weak then there will be an unnecessary amount of deflection in the joint when it is under load. Additionally, a stiffer spring is beneficial for decreasing the oscillation while moving the leg without a direct load applied to it, such as when it is not touching the ground. The spring for each joint was designed to allow a spring deformation of 1/8th of an inch under static loading with at least a max load of two times the expected static load at that joint. This allows for an acceptable amount of deformation under static loading while also providing the required force to function under the expected dynamic loading. The equation for finding the required spring constant is seen in Equation 13 where τ_{joint} is in in*lbs and $R_{sprocket}$ is in inches.

$$k_{spring} = \frac{\tau_{joint}}{*R_{sprocket} * .125} \tag{13}$$

The required spring constant for the θ_1 and θ_2 springs were calculated to be 2866 and 1433 pounds per inch, respectively.

4.2.6 Actuator and Gearbox Selection

Based on the team's calculations and simulations, the torque requirements for all situations the quadruped is expected to face was determined. Using these torques, the team had

estimated an approximate peak torque of 50 Nm and a nominal torque of approximately 40 Nm. These numbers are subject to change as the simulations accounted for the full range of motion of each leg when the robot is only expected to complete crawl gaits. Below is a table that contains details on identified motors that should meet the torque requirements of the quadruped.

Motor	Harmonic Drive FHA Mini	Harmonic Drive FHA-17C	Maxon RE 50 Motor	Maxon RE 65 Motor	ANYdrive Motor	VEXPro 775 Pro Motor
Nominal Voltage (VDC)	24	24	24	24	48	12
Average Speed (RPM)	60	48	150	200	114	9000
Max Torque (Nm)	28	57	45	70	40	0.71
Nominal Torque (Nm)	N/A	N/A	N/A	N/A	15	0.37
Weight (kg)	1.2	2.5	2.6	5.1	1.0	0.36kg
Size (mm)	75x70x100	128x128x78	60x60x150	81x90x160	100x95x90	45x45x66
Estimated Cost (USD)	1700	1200	1300	1500	8000	18
Lead Time	12 weeks	4-12 weeks	4 weeks	4 weeks	4 weeks	1 week

Figure 4.14: Motor Selection

The manufacturers of each motor were contacted for potential sponsorship. We initially reached out to Harmonic Drive who offered the sponsorship of four FHA Minis and a discount for any other motors that are purchased. These motors are rated for a lower torque than the team's assumed design requires so the design of the robot would have had to be scaled down if these motors were used. However, the lead time of these motors was unacceptable for this project and other options had to be looked into.

Our second choice for motors were Maxon EC-i 52mm motors which would be paired with Harmonic Drive gearboxes and EPOS4 Module 50/15 motor controllers. These were able to provide the required torque with relatively low current due to them operating at 24 V. We were able to get a 50 percent discount for the motors and motor controllers and Harmonic Drive offered the custom gearboxes for free, however, we could not raise enough money to make these purchases by the time we needed to order them.

This forced us to consider heavily modifying the design to use lighter materials; however, the team decided to go with a combination of VEXpro 775pro motors and a three stage VEX Versa Planetary gear-train with a total gear ratio of 324:1. For motor controllers, we selected Cross the Road Electronics' TalonSRX. Although this design stretches the strength of the planetary gear-train, we could slightly lighten the design and stay with most of our original design. The main downside of this combination is that the current draw of the 775pros is significantly higher than desired due to its lower efficiency (max of 75% compared to the 90% max efficiency of the Maxon motors) and 12V operating voltage.

The ratio of 324:1 was chosen as a balance between . With the motor data provided by Vex shown in Figure 4.15 and assuming a gearbox efficiency of approximately 90%, we can calculate the current required for the 775pro to produce the required torque. With a 324:1 gear ratio, the torque required from the motor is $35\text{Nm}/(0.9 \cdot 324) = 120\text{mNm}$. From the provided motor data, this corresponds to a current draw of 23.3A.

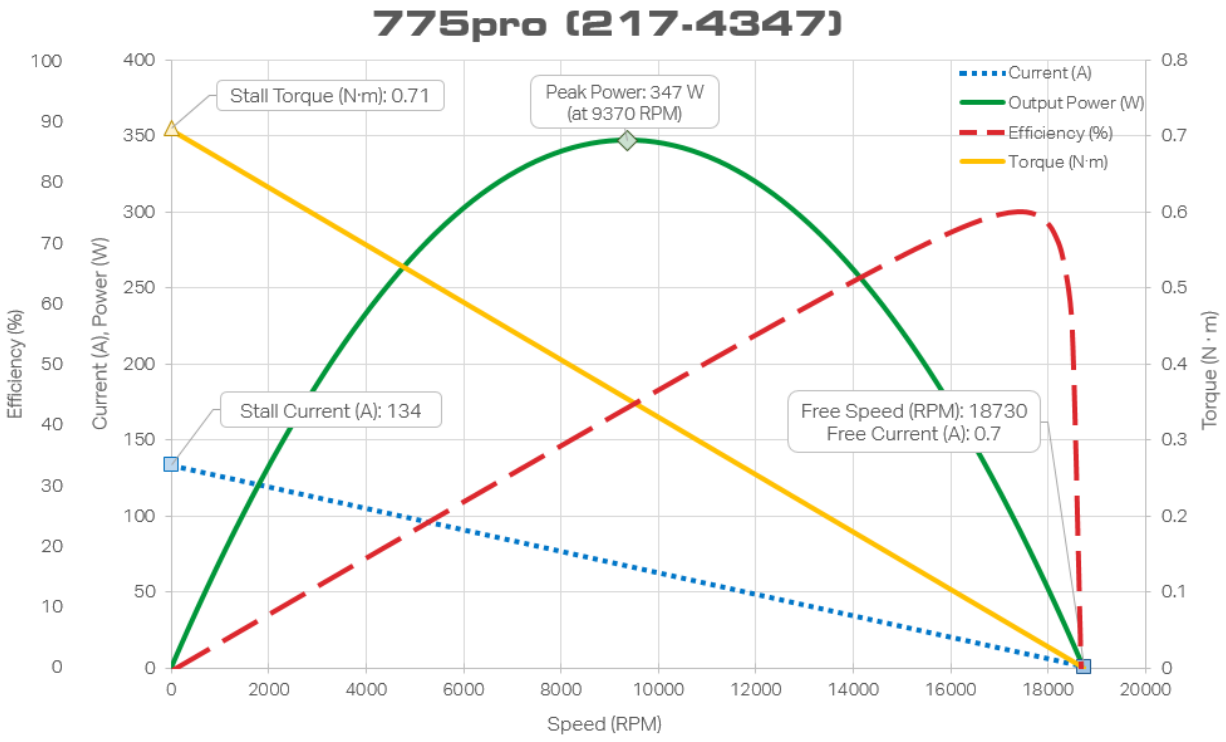


Figure 4.15: 775pro motor data[1]

[1]Motor data from http://motors.vex.com/vexpro-motors/775pro

4.3 Hardware Architecture

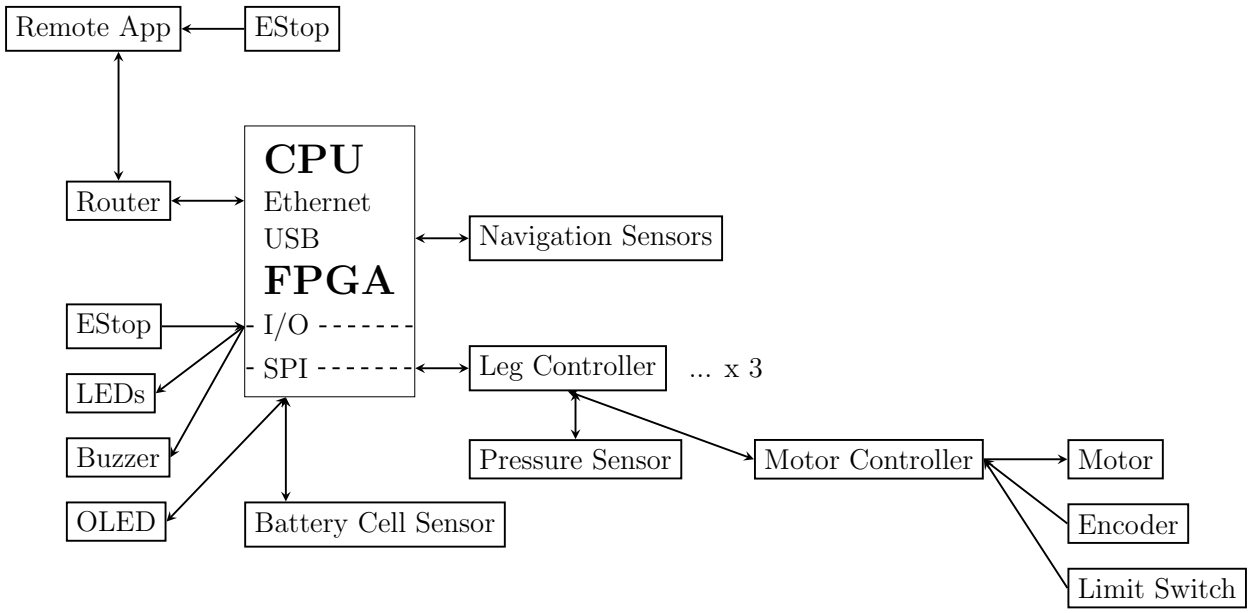


Figure 4.16: General Hardware Architecture

To control the quadruped legs our team decided to use a distributed control scheme as shown in Figure 4.16 and 4.17 to have each leg have its own micro-controller that is then controlled by a main processor. By using this topology the processing requirements for the overall system are divided among smaller processors. This reduces the computational load on the main processor allowing more time to be spent on path planning and gait planning. This also reduces the number of wires and complexity at the hardware interface of the main processor. However this complicates communication within the system. In this topology there are several distinct computational units detached from the main processor that need to communicate and relay all relevant information requested. Debugging the system will also be more complicated as the code is spread across multiple nodes that need to be interfaced with, updated, and kept synchronized.

The main processor will be responsible for path planning, gait planning, controlling the individual leg nodes, reading navigational sensors, and reading general purpose proprioceptive body sensors. Our team has decided to use a SoC development board with an ARM CPU and FPGA logic cells. This has the advantage of giving us developmental flexibility to use the traditional computational model with CPUs and to also augment our system with programmed logic hardware for communication interfaces or accelerator functions. Once the FPGA portion of the board has been designed and programmed it is possible to load the configuration from nonvolatile storage and interface with the CPU on-chip without getting involved with FPGA development.

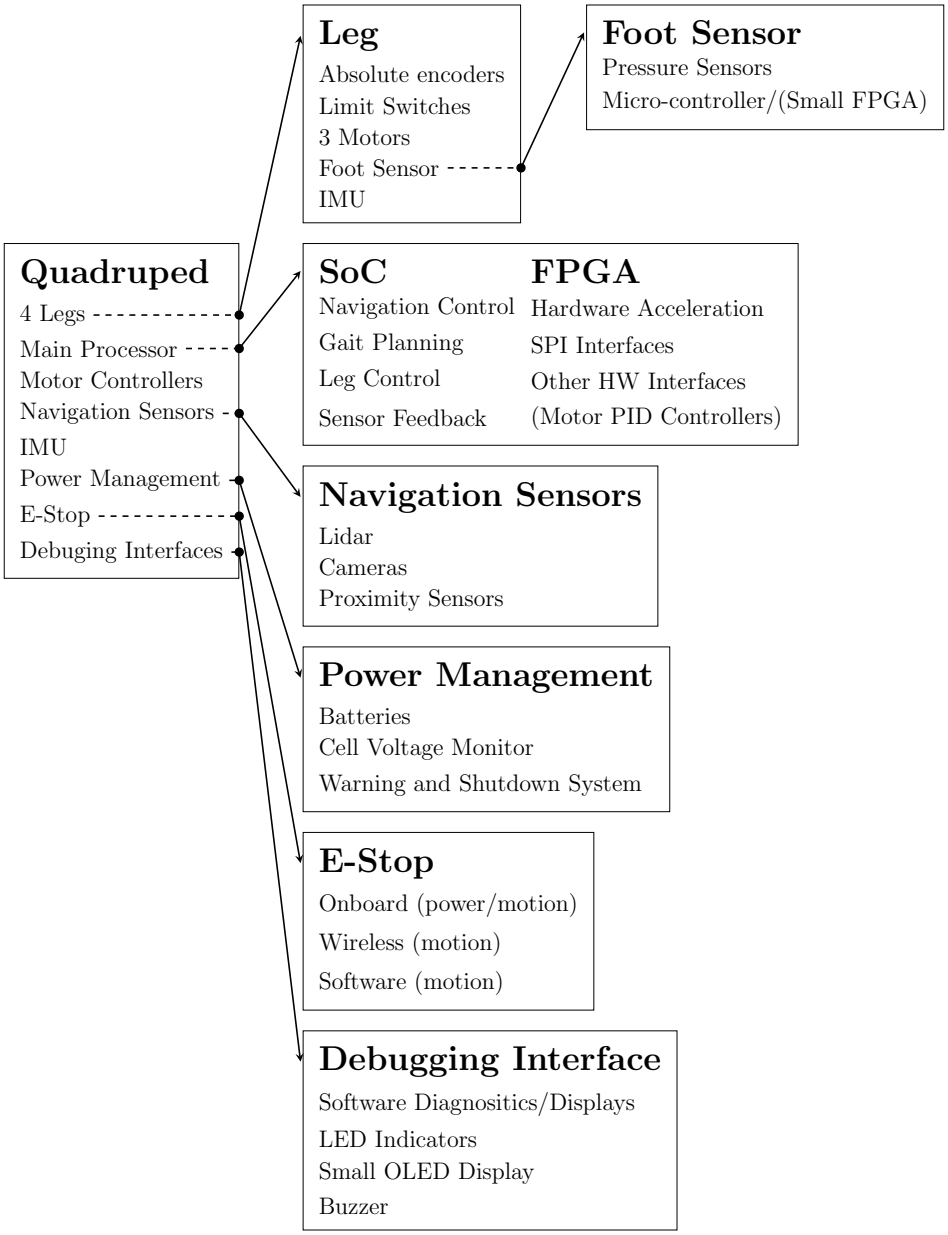


Figure 4.17: Detailed Hardware Architecture

The quadruped will be powered with an on-board battery and ideally also have the ability to be tethered. Figure 4.18 shows a high level overview of the planned power distribution system. To ensure the battery does not drop below a required voltage, for our system to function and for the battery's health, we plan to have a battery cell monitor to measure the voltage of the battery cells to disable motor function. The main power source will then be connected to breakers to power on and off the robot and to be a safety system if too much current is drawn from the power source. This will then supply power to any of the motors through an estop and to a bank of voltage regulators to supply power to on board electronics. The estop will allow the motors to be shut off without turning off the entire robot and stop and torque generation capability.

After deciding on the Vex 775pros and determining the maximum expected current draw we started evaluating options for implementing the power distribution as described above. We decided to use Cross the Road Electronics' Power Distribution Panel, as we already had one on hand. The PDP is connected to the battery through a 120A thermal breaker, and each motor controller has a 40A snap action breaker. However, to ensure that we can safely power all 8 motors simultaneously even if they were drawing the expected max current of 23.3A (a total current draw of 186.4A), we decided to use 2 PDPs, this way each PDP only needs to source 93.2A. To further ensure safe operation, we will use the current limiting feature of the Talons to keep the current draw of each motor to under 30A.

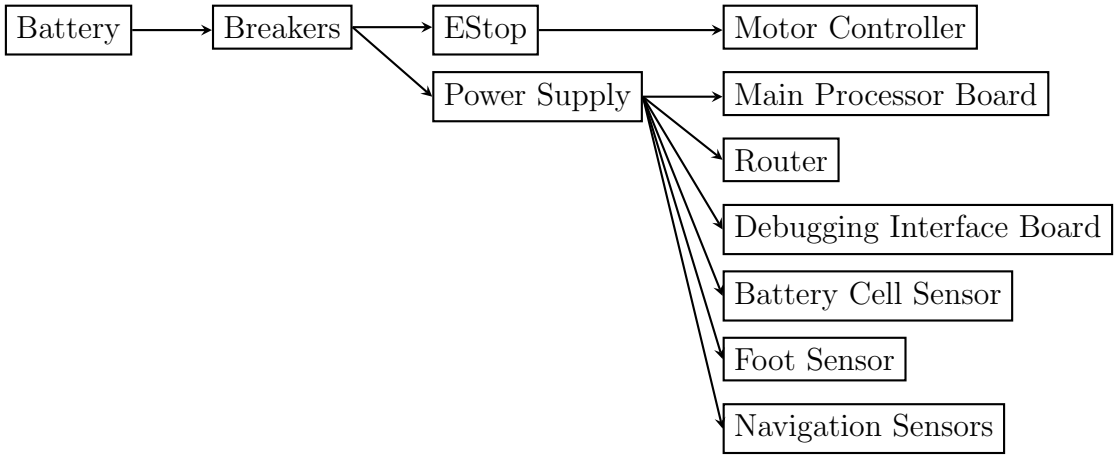


Figure 4.18: Power Distribution

In addition to autonomous controls we plan to implement a debugging and remote control system to control and monitor the quadruped. For the debugging interface, we plan to have LEDs, buzzers, and a small OLED screen for messages on-board the robot to indicate the status of different processes. Then for remote control and monitoring we plan to use a Wi-Fi router to connect to the robot wirelessly and to send information over ROS messages over TCP to ensure reliability. On the controller side we plan to have a client with a GUI to control and monitor data. In addition to regular functions we plan to have a software E-Stop to stop all robot motion.

4.3.1 MicroZed Breakout PCB

Because this PCB is relatively simple in that it serves mostly as a breakout for the MicroZed, and to save on the cost of getting the board manufactured, we decided to design it as a 2 layer board. The board has a few major components, connectors for the MicroZed, the power circuitry, CAN transceivers, and GPIO breakout. The schematic for the board can be found at the end of this section.

The MicroZed has two Amphenol BergStak connectors (61082-101400LF). This board has two appropriately positioned connectors which mate with the connectors on the MicroZed (61082-101400LF).

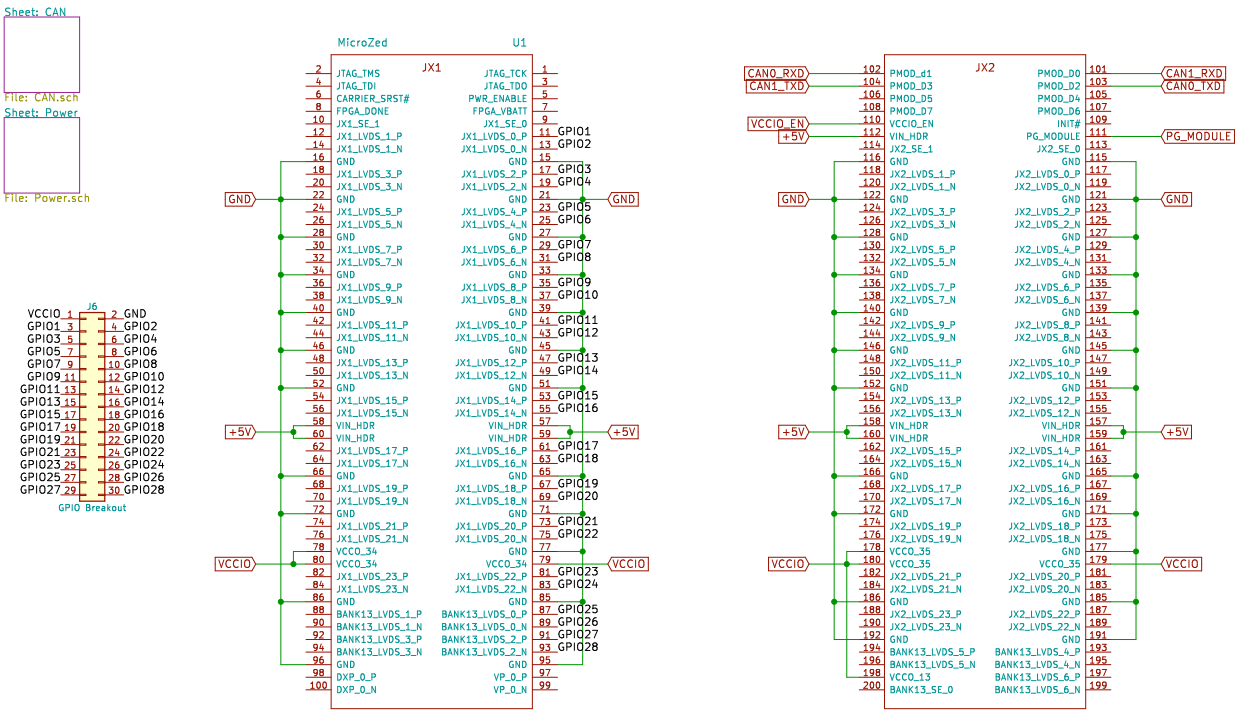
The power circuitry regulates the input 24V down to 5V (which is required to power the MicroZed board) and 3.3V, the logic level used for IO. The regulators used in this design are

two TI LMR16030PDDAR switching regulators, which are rated for input voltages of up to 60V and output currents of up to 3A. Based on the application notes and guidelines in the datasheet, a switching frequency of 500kHz was selected. The input to each regulator has decoupling and bypass capacitors of 10uF and 0.1uF, and the outputs each have two 47uF capacitors. For the 5V regulator, an 8.2uH inductor was selected, and for the 3.3V regulator a 5.6uH inductor was selected. Again, these values are based on the guidelines from the regulator datasheet.

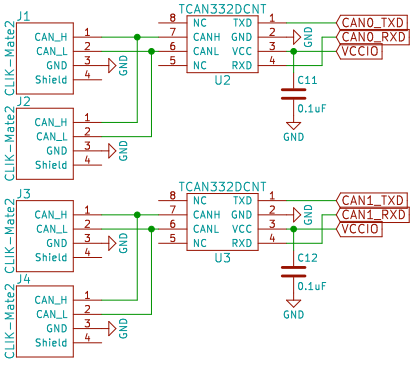
The CAN transceivers take the logic level receive and transmit CAN signals from the Zynq CAN hardware and transforms them to the appropriate levels defined by the CAN standard. For this purpose, we use two TCAN332DCNT chips, each of which handles one of the CAN bus drivers on the Zynq. The physical layer side of these transceivers are each wired to two connectors, providing a total of four CAN bus ports, one for each of the leg controller boards that the MicroZed needs to communicate with.

The GPIO breakout serves as a way to connect any other arbitrary devices that may be useful in the future. In total, 28 GPIO pins, ground and 3.3V are broken out onto a 2x15 female 0.1 inch header. Of the GPIO pins, 24 are from the Zynq's bank 34, and four are from bank 13. The pins that are broken out were selected mostly based on the ease of routing them to the header.

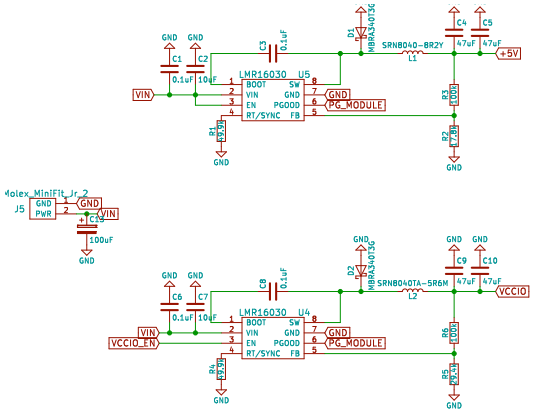
Figure 4.19 shows the schematic for the board. The design after layout and routing is shown in figure 4.20. Top layer fills and traces are shown in red, and bottom layer traces are shown in green.



(a) Microzed and GPIO connectors



(b) CAN components



(c) Power components

Figure 4.19: Microzed breakout schematic

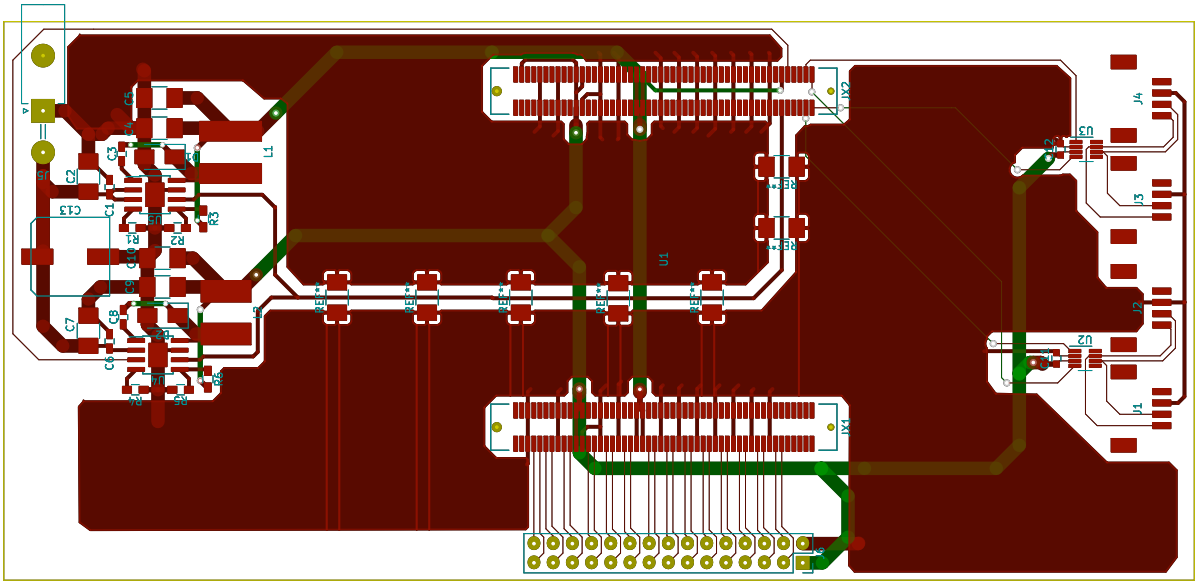


Figure 4.20: Microzed breakout board layout

4.4 Software Architecture

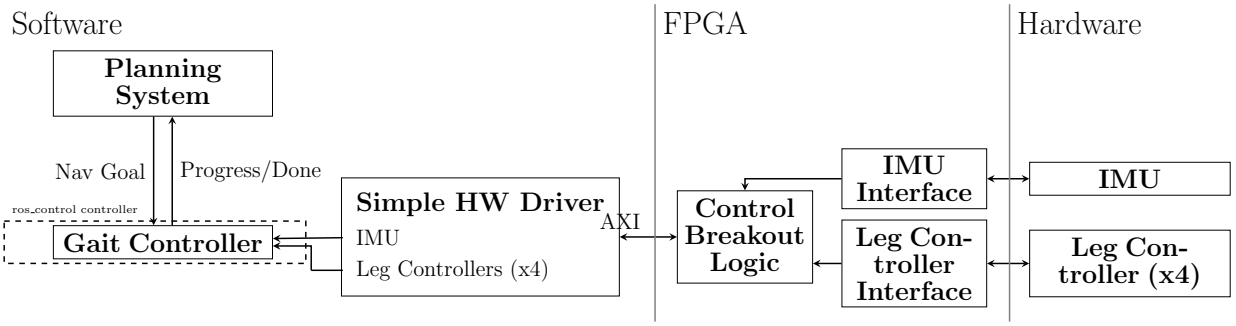


Figure 4.21: Detailed software architecture

Figure 4.21 shows an overview of the software architecture for the software running on the Microzed. It shows the two major components of the system: the planning system, responsible for making high level decisions about what to do and where to go, and the gait control system, responsible for producing leg trajectories to achieve the desired robot motion. The diagram also shows the limited use of the on-chip FPGA, which will mostly be limited to implementing the necessary protocols such as SPI. The software communicates with the FPGA through an AXI bus and there will be a simple driver on the software side that handles that communication and provides a simple interface for interacting with the sensors and leg controllers.

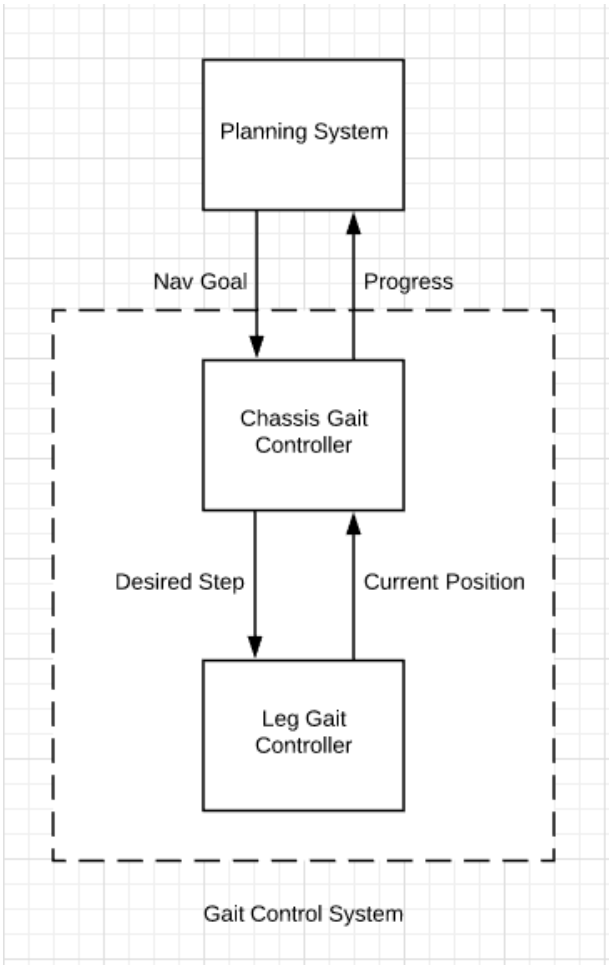


Figure 4.22: Diagram for Gait Control Interface

The gait control system shown in Figure 4.22 is divided into two parts. An individual leg controller and a chassis gait controller. Our team decided that this design was the best choice because walking in a crawl gait involves careful consideration as to where each leg is so that the robot does not tip over or become misaligned. This makes it important for the robot to know exactly where all of its feet are at all times and how much weight is being supported at each foot. By using our chosen design, each leg is abstracted from the chassis which makes computing the zero moment point much simpler.

The leg controller operates on individual legs and is used to generate the path of points for a single foot to travel. These points are represented in a Cartesian coordinate frame where the leg attachment point is the origin. This controller uses inverse kinematics to calculate the ideal angles for each joint to move to. Additionally, velocity control is applied to these equations so that the foot of the leg can smoothly transition through its step. The path of points generated will change depending on the length of the stride and the landing spot of the foot.

The chassis gait controller itself relies on each of these four separate leg controllers to track the position of each foot. By using interrupt timers to determine where each foot is positioned and the foot sensor to determine how much weight is on each foot, the zero

moment point of the robot can be obtained. This information is then passed to the navigation system to determine where the next step should land to keep the robot from tipping. This ensures the crawl gait is smooth and reliable.

The gait controller also calculates the stability of the platform using the force data supplied by the foot sensors. This information is used to calculate the ZMP, which was chosen over the COM method due to it already factoring in the inertia of the system. Unlike the method described in the background where the ZMP is calculated by simulating the body, the gait controller works backwards from the force information to allow for faster calculations. Working under the assumption that the robot is walking on completely flat ground, the equations

$$a_{ZMP} = \frac{z_1 a_1 + z_2 a_2 + z_3 a_3 + z_4 a_4}{z_1 + z_2 + z_3 + z_4} \tag{14}$$

$$b_{ZMP} = \frac{z_1 b_1 + z_2 b_2 + z_3 b_3 + z_4 b_4}{z_1 + z_2 + z_3 + z_4} \tag{15}$$

$$c_{ZMP} = 0 \tag{16}$$

describe the location of the ZMP where each foot has a force acting on it with components x_i, y_i, z_i and the location of each foot and ZMP are denoted by (a_i, b_i, c_i). One downside to this approach is that is does not allow for easy estimation of the future location of the ZMP. To compensate for this, the team added padding around the calculated support polygon so that if the ZMP approached instability, the gait controller would compensate by changing the joint angles in the legs to move the ZMP away from the edge of the support polygon. This design works as long as the robot is not moving fast enough that its inertia would cause instability before the controller can react.

The planning system determines the goal location of the robot and how it will get there. This is done by sending the navigation goal to the chassis gait controller and receiving the current progress as feedback. By using this feedback, the planning system can ensure that the robot reaches the correct position at a reasonable speed. This system can also overwrite and recalculate the goal if a new position is requested.

The central Microzed communicates with each of the Teensys that act as leg controllers over the CAN bus. The software running on the Teensys was designed to be as configurable as possible so that they do not have to be reprogrammed often, as updating the firmware on the Teensys means reprogramming each Teensy individually. To this end, the leg controllers listen for a number of configuration messages such as which control loops should be running, what the parameters of those control loops should be, which joints should be enabled, etc. This allows different programs running on the Microzed to perform different tasks without the need for reprogramming the leg controllers. This greatly simplifies the normal workflow of working with the robot especially given that the Microzed can be programmed and debugged over a WiFi connection. The firmware was also designed with extensibility in mind such that changes like adding control over the third joint of each leg would be easy.

4.4.1 TalonSRX Driver

One problem with using the Talon as a motor driver is that the protocol used to control it over the CAN bus is poorly documented. Fortunately, Cross the Road Electronics has an implementation of the driver in C# hosted on their Github account from which the protocol could be reverse engineered[2]. Based on this implementation, we implemented our own driver for the Talon. While the protocol itself is quite simple, because there was incredibly little documentation and most of the information came from the C# implementation, implementing our own driver did take a fair amount of effort. What follows is an overview of the protocol as we understand it.

The Talon communicates on the CAN bus at the maximum standard rate of 1Mbps and uses extended identifiers. The low 6 bits of the message ID are used as a device ID so that multiple Talons can be on the same can bus. The rest of the bits are used to indicate a particular type of message. For example, a message with the ID `0x02040005` is for the Talon with ID 5 and is a `CONTROL_1` message.

Under normal operation, the Talon expects to receive certain control messages at partiular rates; for example, it expects to receive a `CONTROL_5` message every 10ms. This control message contains information such as the current setpoint, control mode, which sensor to use in closed loop control modes, etc. If the Talon stops receiving these control messages, it assumes the controlling device is no longer present and stops the motor. The Talon also broadcasts several status messages at various intervals. The `STATUS_1` message for example is broadcast with a 10ms period and contains on the current control mode of the Talon, the closed loop error, and the voltage currently being supplied to the motor; the `STATUS_2` message is broadcast with a 20ms period and contains the position and velocity data from the selected feedback sensor and the current sense reading. There are several other status messages as well.

In addition to these status and control messages which contain data which must be kept up to date, the Talon has messages to support the getting and setting of parameters which change infrequently, such as PID gains for closed loop control, or the periods of the status messages. To set such a parameter, a device sends a `PARAM_SET` message that contains the index of the parameter to set as well as the parameter value. To get a parameter from the Talon, a device sends a `PARAM_REQUEST` message containing the index of the parameter and waits for the corresponding `PARAM_RESPONSE` message which contains the current value.

For a more detailed description of the different message types and what each one contains, see Appendix A.

[2]https://github.com/CrossTheRoadElec/Phoenix-netmf-lib

5 Results and Discussion

5.1 Hardware Setup Prototyping

To validate the team's initial hardware design, a prototype of the leg controller was tested on a bread board. This allowed the team to verify our initial electronics design before designing and manufacturing PCBs. A picture of the breadboard system can be seen in Figure 5.1. Using this setup the team was able to verify the functionality of our initial leg controllers software environment setup and test peripherals on the microcontroller. The most important test was verifying that the CAN interface and libraries on the teensy microcontroller and the MicroZed board were able to send data between each other.

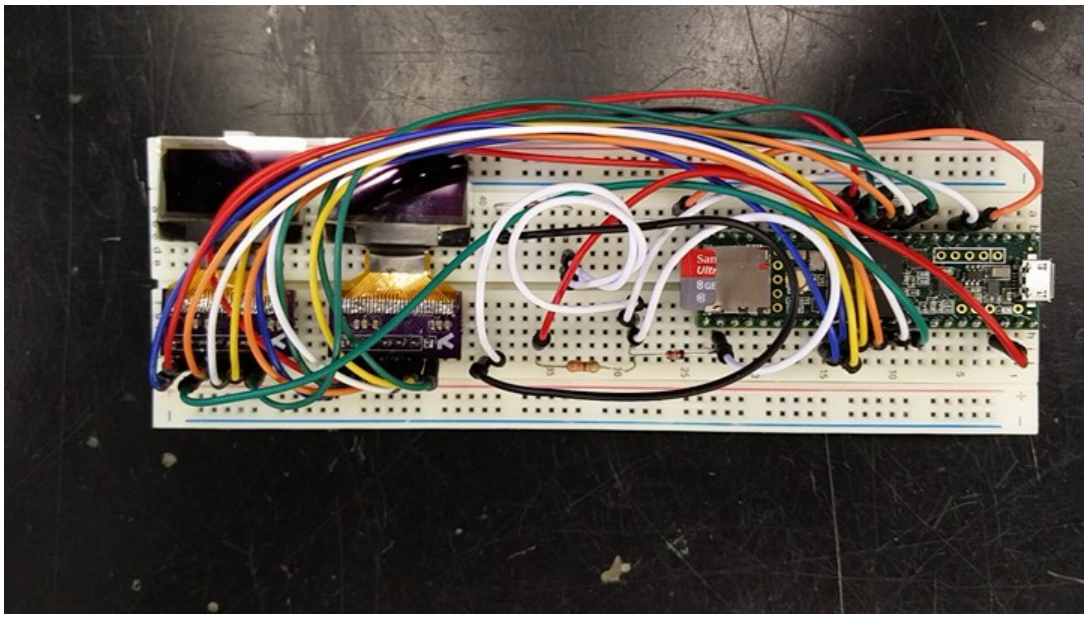


Figure 5.1: Hardware Prototyping

Once the Microzed breakout PCB was manufactured, we immediately began testing it to verify its functionality. The assembled breakout board is shown in Figure 5.2. The breakout board works as intended, however, after manufacturing and testing the breakout board, a few minor flaws have been noted in its current design. During the design process, the layout was changed several times and at one point we forgot to remeasure the clearance for a few of the ports on the Microzed. Because of this, while the Microzed is socketed into the breakout board, the micro USB port and SD card slot are inaccessible. Fortunately this is only a minor inconvenience because the typical workflow for working with the Microzed requires only an ethernet connection to the Microzed, and the ethernet port is accessible. The other oversight was the lack of mounting holes, both to mount the breakout board to the robot, as well as holes to secure the Microzed to the breakout board. All of these issues could be solved by relatively minor changes to the design, and will be implemented if another iteration of the board must be manufactured.

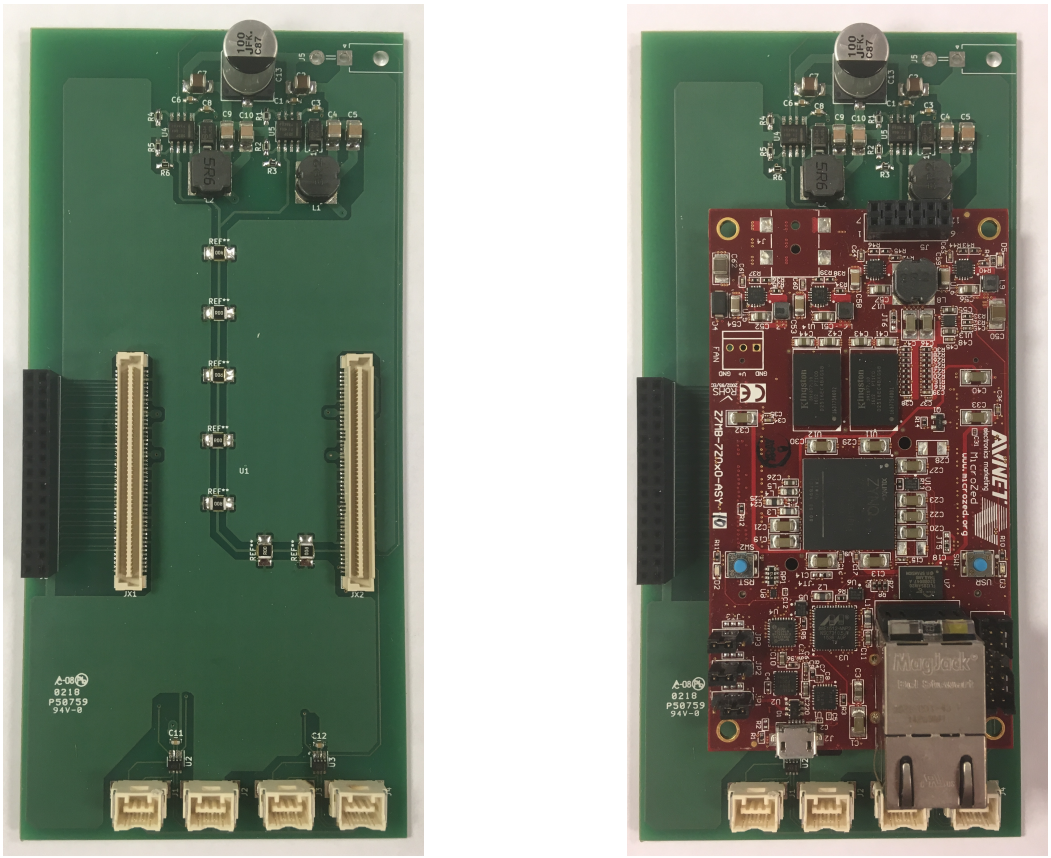


Figure 5.2: The assembled Microzed breakout board

5.2 Initial CAD

As mentioned in previous sections, the team decided to design the entire platform using the SolidWorks CAD software. The initial design drew inspiration from existing quadrupedal robots, particular the StarlETH robot from ETH Zurich. Renders of the first full CAD designs can be seen in the following figures:

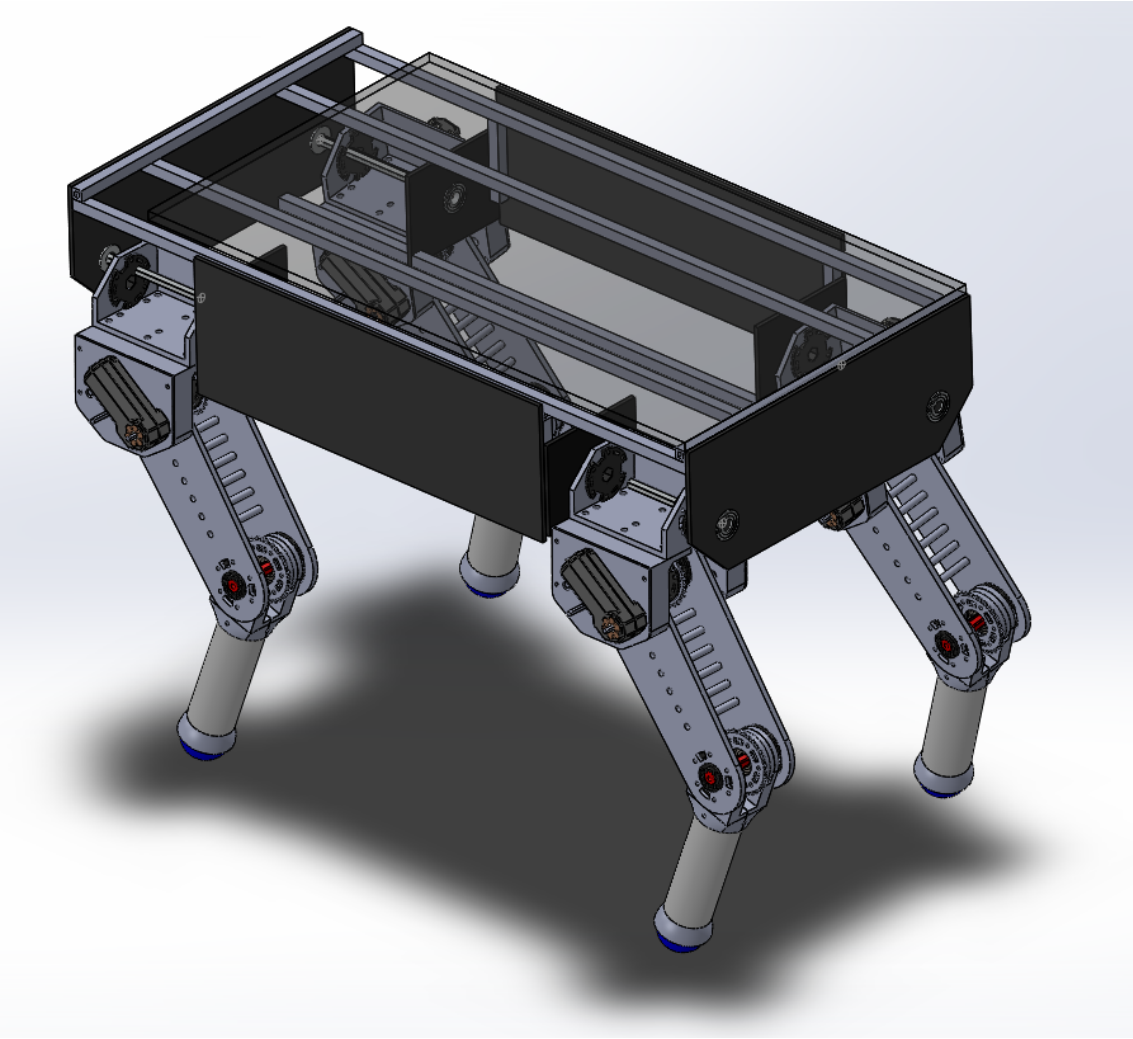


Figure 5.3: Initial CAD of Body Design Isometric View

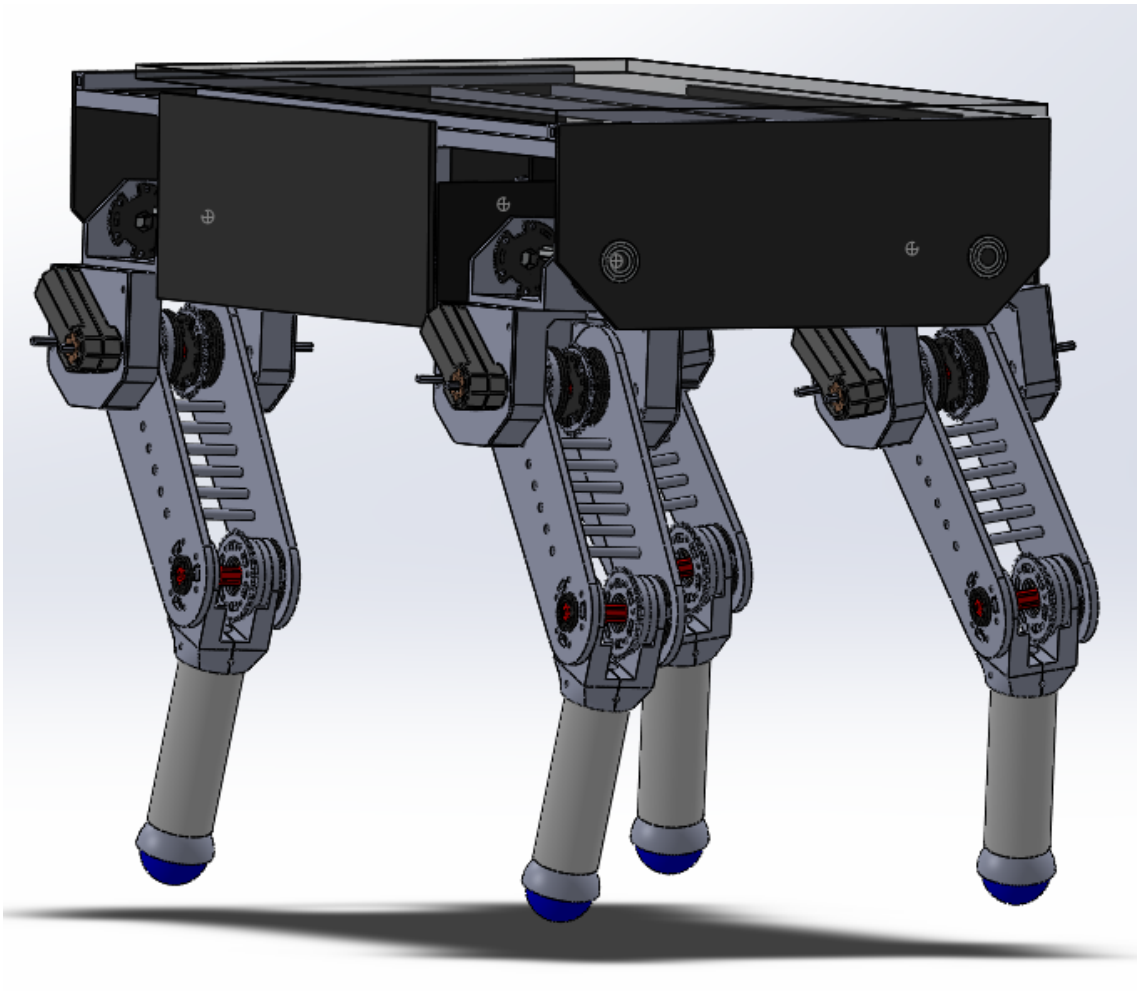


Figure 5.4: Initial CAD of Body Design

This initial CAD model was very basic and allowed us to verify our design methods and initial plan without a significant time investment.

5.3 Initial Leg Prototype

To verify the leg design, which can be seen in Figure 5.5, we created an initial leg prototype.

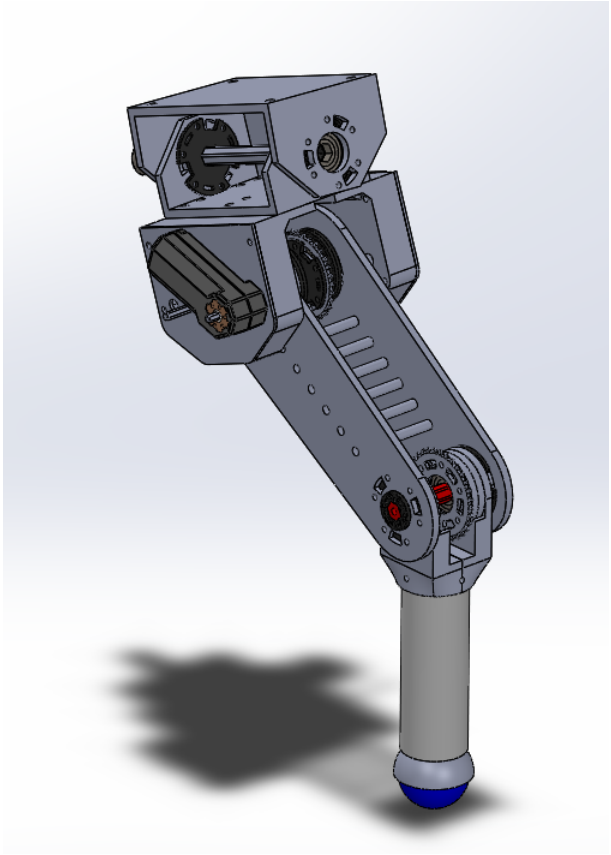


Figure 5.5: Leg Design CAD

This initial leg prototype, seen in Figure 5.6, was built to the same scale of the intended design to assist with validation of the initial CAD design. Since the use of metal for prototyping had the potential to be rather expensive, this prototype was constructed using cheaper materials as substitutes. Instead of water-jet sheet metal plates, laser cut plywood was used. Additionally, instead of machined aluminum blocks, 3D printed components were used. For the lower link, an ABS pipe was used as a substitute for an aluminum pipe. To lower the cost even further, motors that were readily available, namely a combination of REV Robotics and Pololu motors, were utilized rather than the desired motors (which were not yet available at the time of assembly) to drive each joint. The usage of these motors rather than the desired motors did not take away from the realism of the prototype to the final design despite their differences. This is because the shafts of the motors used were adapted to the 1/2" hex shaft of the final design in the custom gearbox we made for the prototype.

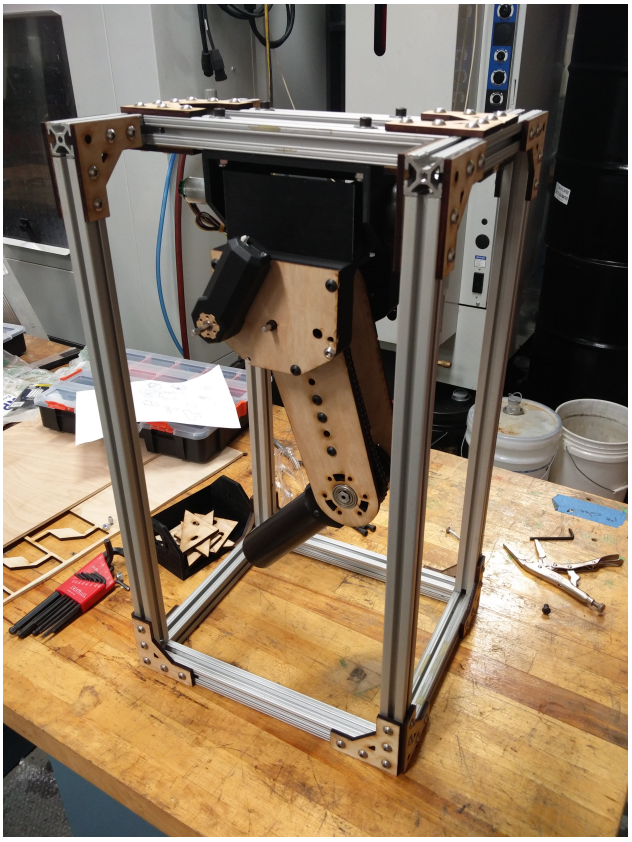


Figure 5.6: Leg Design CAD

After the prototype was assembled, the team noticed a few problems with the alignment of the leg design. Most notably, we found that we had forgotten to account for the thickness of the screw heads between the upper link and the housing. We also realized that the series elastic actuation for the lower link had been designed wrong due to a misunderstanding of the StarlETH design.

Although the team originally intended to use this prototype as a test bed for the structure of the leg control system, we never drove the motors on this initial prototype. This was mostly due to the significant difference in control between the motors placed on this prototype and the desired motors for the final product. Rather than implement a different control structure for this prototype, which would need to be changed later, the team decided to focus on improving the design and creating a second prototype with the full scale motors.

5.4 Second Leg Prototype

After making the changes to the design from the initial prototype, we made a full scale version of our leg with the selected VEX-Pro 775 Pro motors which can be seen in Figure 5.7. This leg was made out of aluminum instead of the weaker plywood, which allowed for extensive testing of the 775 Pro motors and our leg control structure. Additionally, the series elastic actuation of the joints was added so that position control with the springs could be tested and tuned.

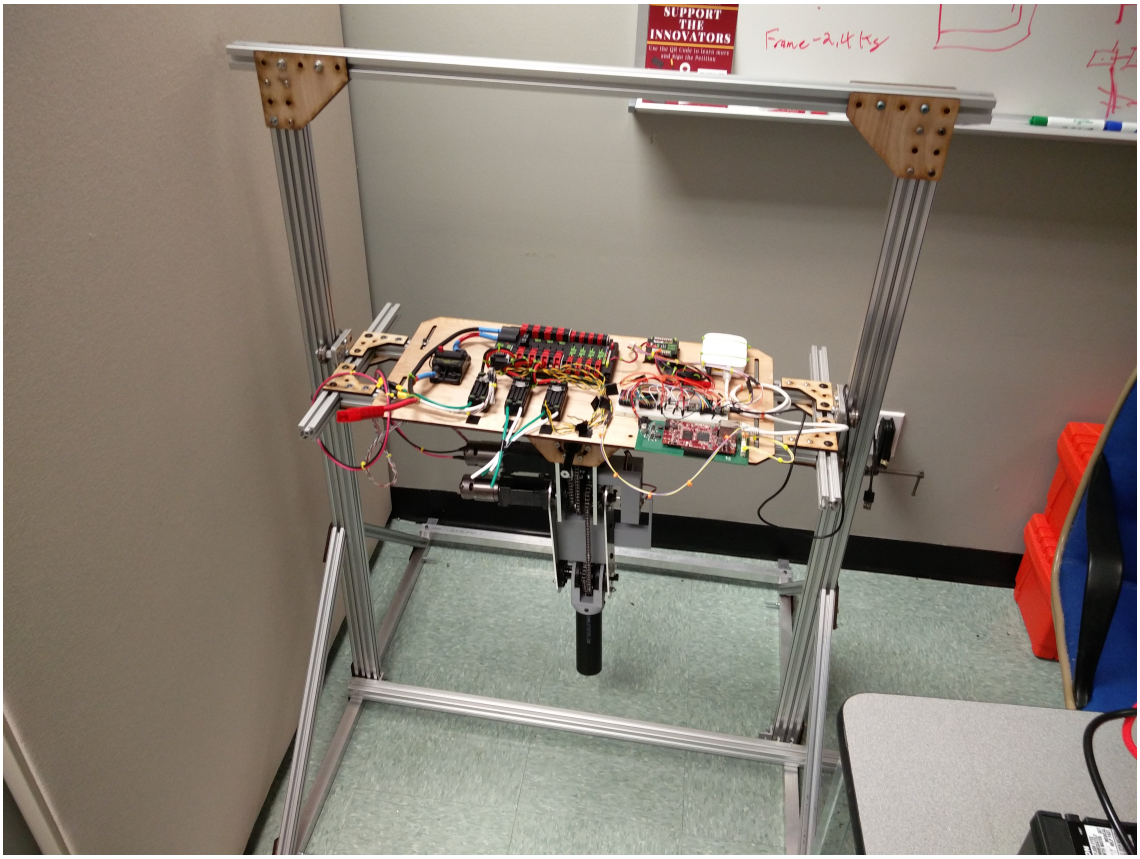


Figure 5.7: Leg Prototype 2

This prototype leg was mounted to an 80-20 frame just like it would mount to the chassis of the platform. A plywood board to house all necessary electronic components for the leg to move was also attached to the top of this frame to make control and wiring of the test rig easier. This assembly was in turn mounted to two linear sliders so that the leg could either move freely in the air or push off from a surface and bear weight for testing of the design under a simulated load.

During testing of this prototype, the team found that some of components that we planned to use for the series elastic actuation were not strong enough and needed to be switched out for stronger items. This caused issues with the clearances of the series elastic actuation components and necessitated making the leg less wide. To counteract this decrease in size, the team also decided to move to press-fit bearings rather than the bearing hubs we had been using, which would also make the leg slightly lighter.

Additionally, when strength testing our 775 Pro motors, our team broke an internal gear for our gearbox. This was due to the small gear diameter of the sun gear on a 9:1 stage gearbox which led to the gear shearing as it was handling the highest level of torque in the gearbox. We were able to fix this problem by moving the 4:1 stage to the end of the gearbox and the two 9:1 stages below it. After fixing this, we retested our motors without failure. While this proved that our motors and gearboxes could handle the loads that we require of them, we are very close to the failure point of the gearbox so our safety factors are not as high as we desired.

5.5 Final CAD and Product

Figures 5.8 and 5.9 show the team's final CAD of the robot. The biggest changes of this design were cutouts in each of the sheet metal plates to save weight and changes in the leg design. These leg changes were based upon the testing of the second prototype and involved making the leg design more compact.

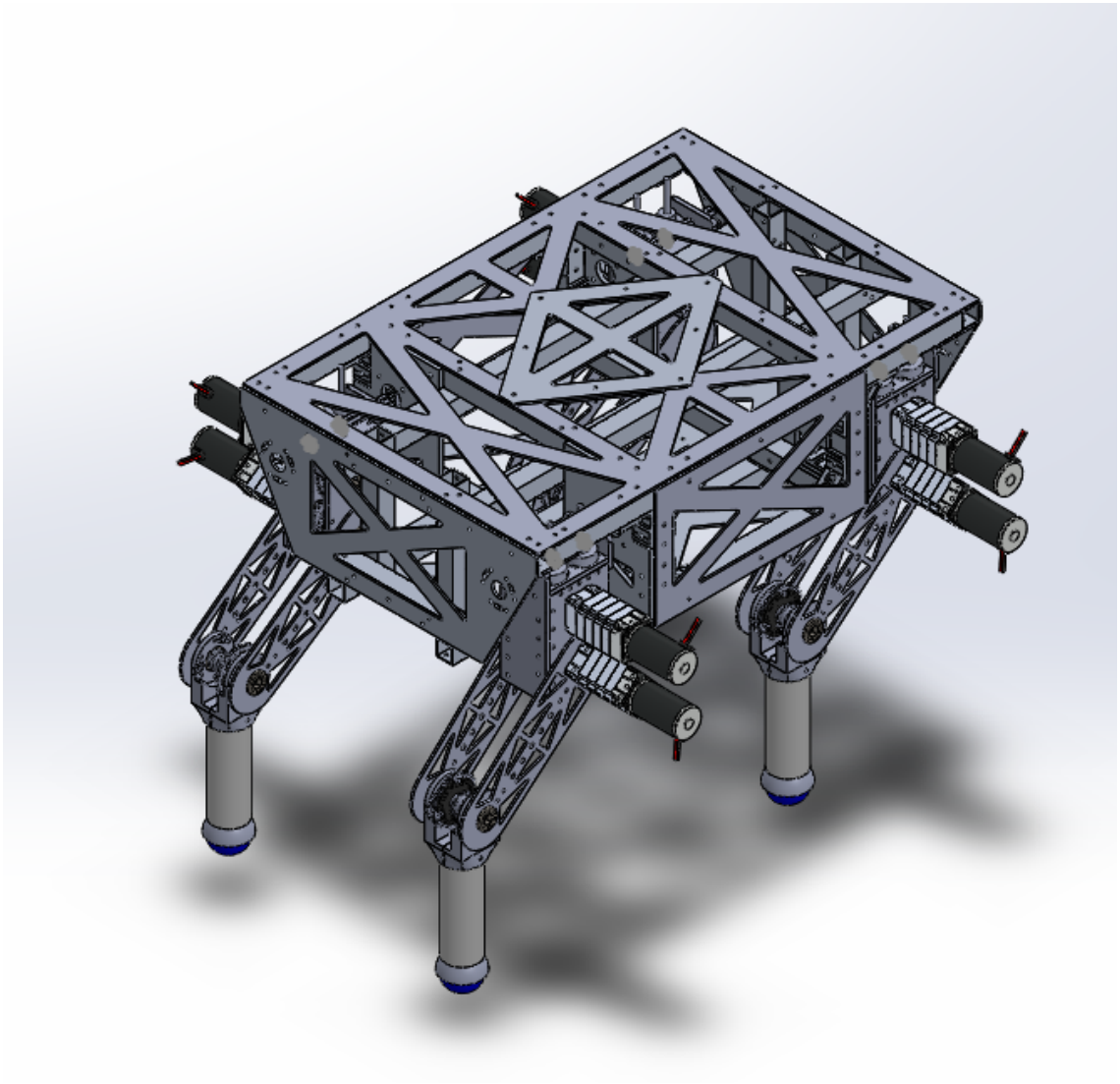


Figure 5.8: Final CAD of Body Design

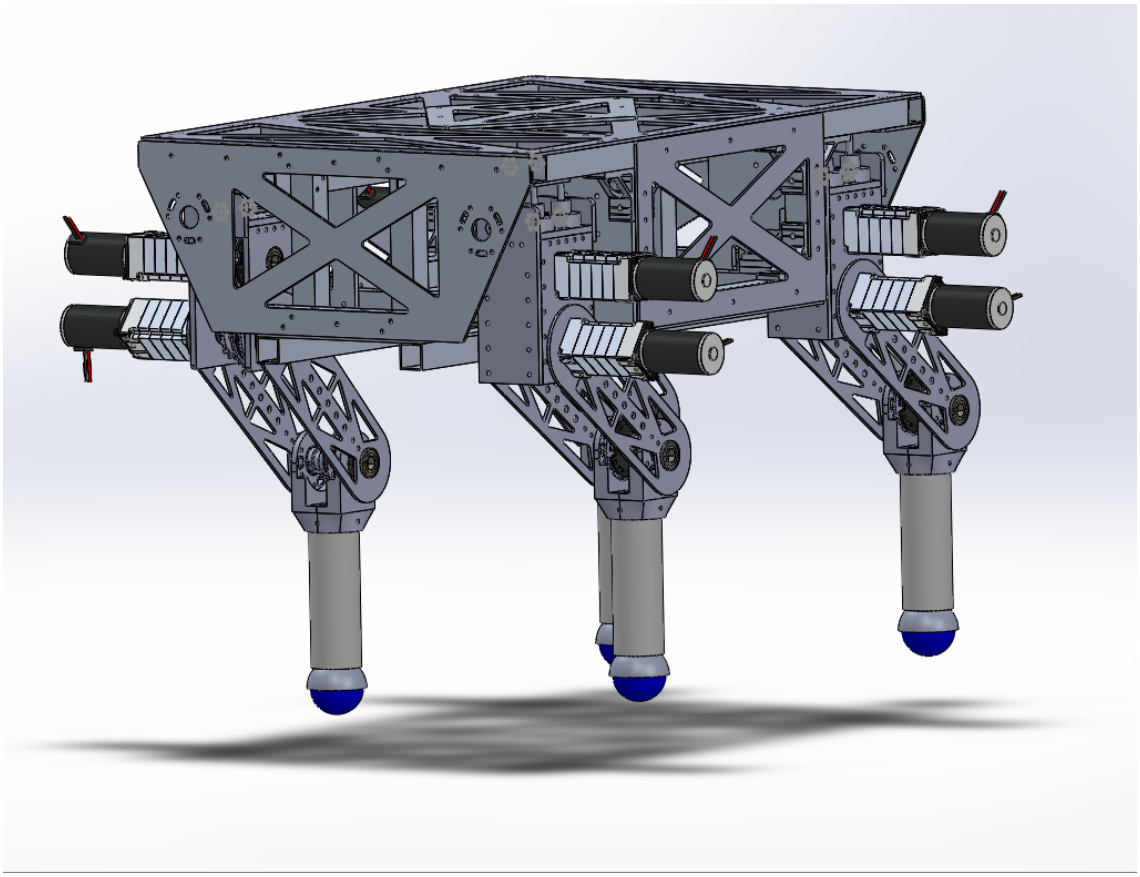


Figure 5.9: Final CAD of Body Design 2

The Final CAD was also used to design and implement the electronics mounting stack that was placed in the center of the robot to hold the leg controllers, main controller, router, and power distribution hardware.

The final platform can be seen in Figure 5.10.

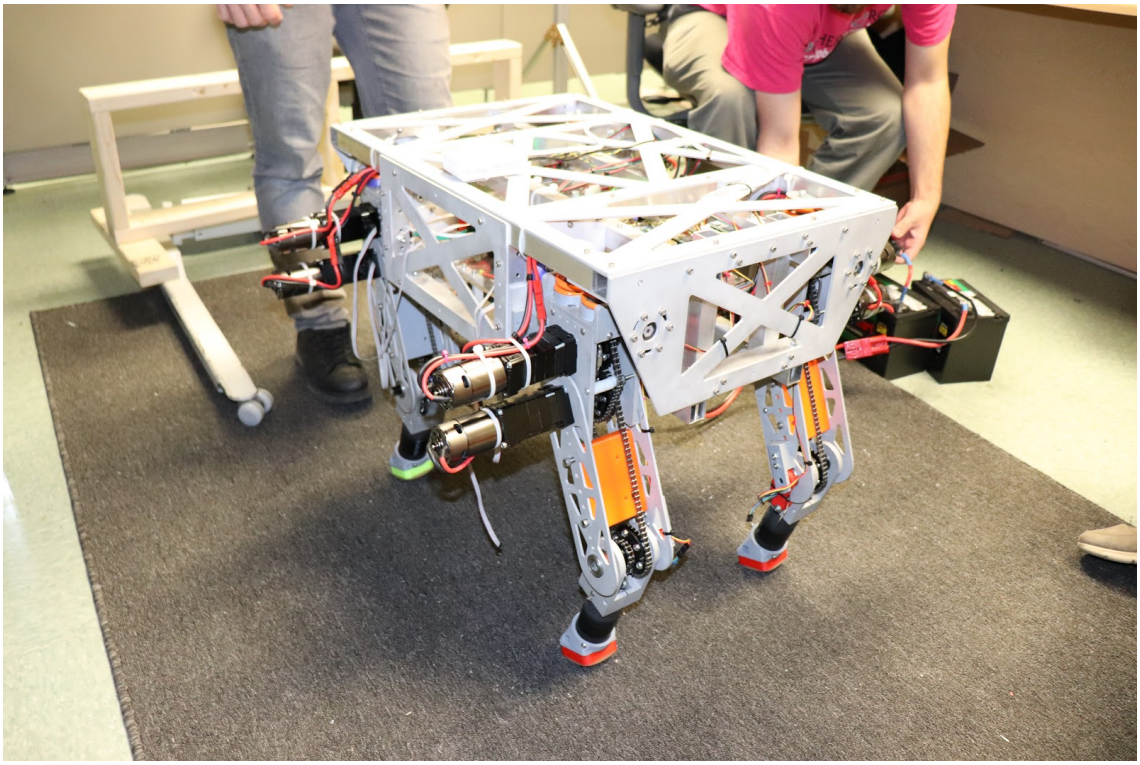


Figure 5.10: Final Product

Due to a significant amount of design changes that had to be made due to budgetary restraints, the manufacturing and assembly of the final robot started a few weeks later than was originally desired. Additionally, manufacturing the parts took a week longer than was desired due to the team not being very familiar with Fusion CAM software and scheduling issues. This added up to the final platform not being completed later than desired which ran into the scheduled testing and software implementation time.

The overall product had relatively few issues with the final design that had to be fixed. One of the major changes that was necessary was that the design of the original potentiometer mounts caused the wires running from the potentiometer to collide with the other half of the mount during normal operation which would break the potentiometer. This was easily fixed by redesigning the mounts to run the wires along the leg rather than out the side. The team also found that during testing it was quite common for the quick links connecting the springs to the chain would come unlatched and break. We believe that this could be easily solved by increasing the size of the chain for the lower link to be a #35 chain so that a larger size quick link could be used and adding Loctite to the threads of the quick link during assembly.

5.6 Foot Sensor Prototypes

After the design of the foot sensor and the mould was complete, the team started testing the moulding methods and materials we had chosen with the boards we had made. We started by casting just the board in a flat mould created with a 3D printer and found that the surface finish of the mould was not desirable where the material touched the surface.

This confirmed to us that the final cast had to be machined rather than 3D printed. However, it also meant that the calibration of the chips would not be possible without a better surface finish. We noticed that the part of the material that was exposed to the air had the desired surface finish, so the mould design was inverted to expose the bottom surface to the air during casting.

When the team went to cast the rounded mould, however, the open air method was not possible. The team originally tried pouring the material through the top of the mould, but found that the material was too viscous and caused an air bubble at the top of the mould despite the addition of air vents. To solve this problem, the team decided to pipe the material into the mould through a tube to the bottom of the opening and letting it fill to the top. Although this method was harder to pour and took longer, it avoided the air bubble problem by allowing the air to leave through the ventilation holes at the top of the mould while the material filled in slowly.

Once the moulding methods were proven to work, a fully functional PCB was cast to verify that the readings obtained were useful. This test cast used the flat casting method since the point of the experiment was just to verify that the sensors could be reasonably calibrated to provide good readings. The results of this calibration test can be seen in Figure 5.12.

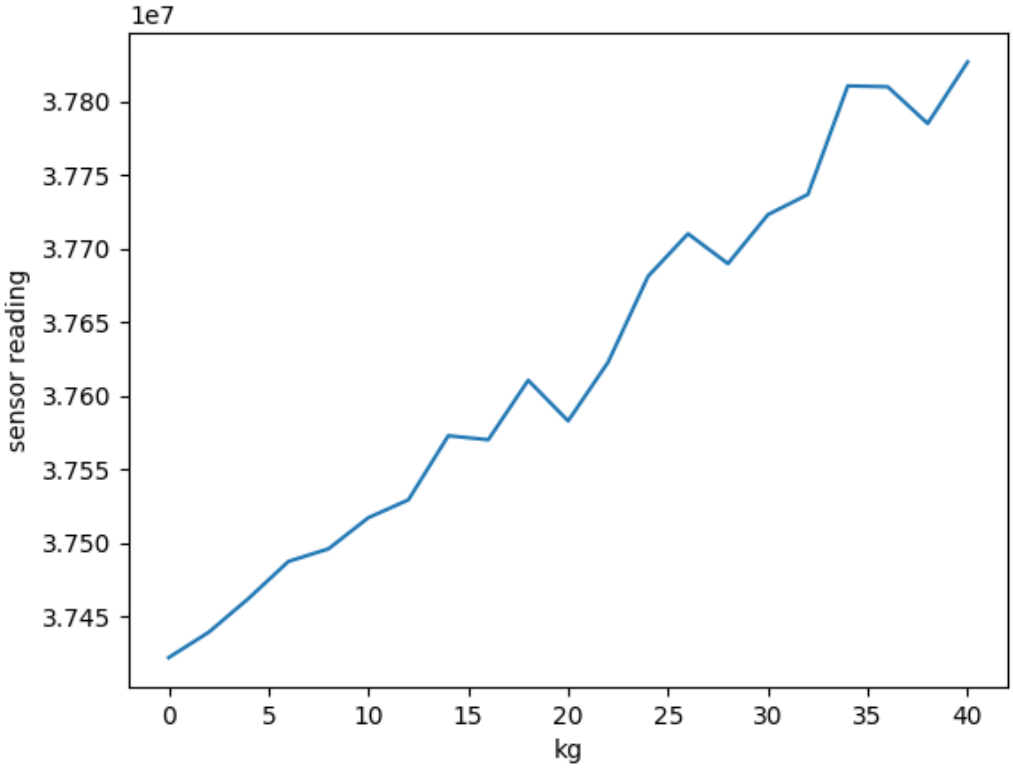


Figure 5.11: Foot Sensor Calibration Test

While the results of this test were not ideal, there is a clear linear correlation between

the normal force applied to the sensor and the resultant sensor reading. The team attributes the spikes in the graph to the poor testing method that we had available due to the lack of a proper force gauge with which we could read and apply exact force vectors. We concluded that while the test proved the design worked, the calibration method would have to be greatly improved to obtain a useful sensor reading. The team also decided to use a polymer with a lower shore hardness on the final foot sensors to allow for increased sensor readings and resolution on the final product.

While we were able to cast the final foot sensors and attach them to the legs, we did not have time to calibrate and implement them into the final system. This removed our ability to calculate the stability of the system.

5.7 Implementation and Testing

Once the first final leg was assembled, the team began testing to ensure that it would meet the requirements, and to find any problems before manufacturing the other three legs.

Among the tests performed with the leg were individual calibration of both the motor velocity control loop and joint angle control loop, as well as multi-joint position control tests. We tested position control of the leg both for motion through the air and for motion while under weight.

For testing under weight, the frame from Figure 5.7 was used. Because the leg was mounted to the linear slide, this test was only able to asses the ability to control vertical motions, however it did give us confidence that the leg was capable of supporting the weight necessary, and that we were able to control it appropriately while it was under that weight.

During the testing of the leg, a small number of design changes were made: better chain tensioning was added for the chain that controls the lower link, a slot was cut out of one of the plates to make room for the chain to prevent it from limiting the leg's range of motion, and the design of the spring pre-loading was slightly modified to use larger quick links. It was also discovered that the joints had more slop than desired, even after improving the chain tensioning. The team attributes much of the slop to the three-stage gearbox and the adapter from the gearbox output shaft to the hex shaft it drives. Although the slop was not an issue when the leg is under weight, it was found to severely limit the bandwidth with which the leg can be controlled for movements through the air, and makes transitions when lifting a leg off the ground or setting it down more delicate and prone to causing instability.

Because the team had neither the time nor budget to find a proper solution to the undesirable slop, we decided to deal with the problem by limiting the speed of the legs to something achievable despite the slop. Satisfied with the performance of the single leg, the team proceeded with manufacturing the remaining three legs.

Once the robot was fully assembled, we were finally able to begin testing the system as a whole. This testing started with commanding each leg through the same motion in the air to ensure that they were all working properly and were synchronized. From there, our tests slowly built on top of one another towards our final goal of unsupported walking. During these tests we showed that the robot was able to stand up and support it's own weight, control the position of the chassis to move the center of mass of the robot, and from there pick up a leg and balance stably on the remaining three legs. This particular test was

repeated to ensure that the robot was able to successfully balance on any combination of three legs.

While it was the one remaining major goal left, we unfortunately ran out of time trying to get the robot to complete a stable crawl gait. We believe that the main shortcoming of our implementation of the crawl gait was it's simplicity, it consisted of a precomputed trajectory that each leg followed (different legs each followed the trajectory with a 25 percent offset so that as one leg was beginning the swing phase of the trajectory, another was completing it). This meant that the robot would not make adjustments if picking up a leg would cause it to become unstable. We also encountered a weakness of the current implementation of inverse kinematics which was that the exact length from the lower joint to the contact point with the ground varies with the contact angle. Our inverse kinematics implementation did not take this variation into account and so led to planting a foot at the wrong height when finishing the swing phase and contributing to the robot becoming unstable. Most of these problems could have been solved by the proper implementation of the foot sensors to sense when a foot was on the ground before moving onto the next foot and to allow for more precise foot placement. However, as mentioned in Section 5.6, the team ran out of time implementing this feature.

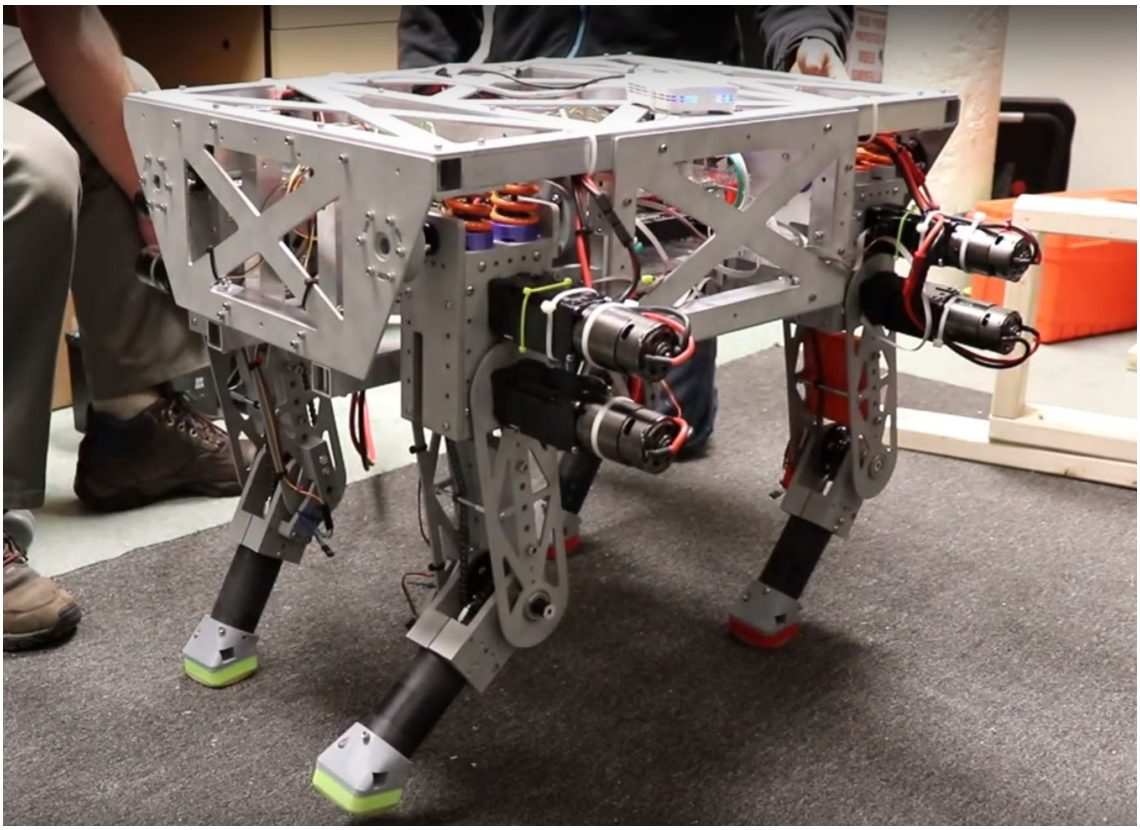


Figure 5.12: Unsupported Standing on 3 Legs

Based on our previous tests we fully believe that the robot is capable of attaining a stable crawl gait, and that with more time a smarter algorithm can be developed that actually uses feedback from the foot sensors to ensure that the robot remains stable throughout the gait.

6 Future Work

While the nature of this project is to create a platform for future use and research, which inherently allows for expansion, there are a few improvements that can be made to the current system.

Acquiring and installing the stronger motors that were initially specified for the system would be a major mechanical improvement to the current platform. The team had issues with the slop in the final three-stage gearbox that would be significantly improved by using a single-stage gear train, increasing the ability to precisely control the system. This would improve both the stability control of the system and the movement of the leg through the air during a gait, allowing for faster walking speeds. Additionally, the desired motors operated at 24V rather than 12V which decreases the required current draw by half, something that would be an integral step towards allowing battery rather than tethered operation.

On another hardware-related note, improving the calibration method of the foot sensor would allow for more accurate sensing of the force vectors on the feet and improved stability control. This would require recasting the foot with the original 3x3 grid of sensors and calibrating the outputs. The outputs of these calibrated foot sensors could then be paired with the gait software to ensure that the platform would always maintain stability throughout the gait of the robot and would be integral in planning foot placement on uneven terrain and during the walk gait. The improved foot sensors could also be paired with implementing the third degree of freedom in the legs for more precise stability control throughout a gait.

The team briefly was able to delve into the control of the crawl gait and the transition from standing to crawl gait, but this is a major source of future expansion, in addition to implementing a walking gait (which would likely require implementation of some of the above improvements).

This platform can be used to further examine the effects of different types of series elastic actuation on a quadrupedal system. Such research could analyze the effects that different strength springs and spring setups have on the control of SEA joints as well as their response to impulses or different nature. This analysis has many implications on the stability control of systems such as this and could potentially lead to a redesign of the SEA system implemented in the legs to be more ideal for the desired operation.

7 Conclusions

This project was successful at reaching most of the design goals our team set. Over the course of one academic year, this team designed, built, and programmed a quadrupedal robotics platform for use in future research. The end result weighed below 40kg and fit within a 1x1x1 meter cube as required by our design specifications. This robot was able to stand unsupported on all four legs, or any of the three if desired. Our team was able to shift the weight of the robot while standing from front to back. This type of control exhibits that the platform is capable of achieving the unsupported crawl and walk gaits as intended. Unfortunately due to time constraints we did not have time to fully achieve a crawl gait and the stability control we originally desired but the team believes that implementing these functions would not be too hard and expanding on our work is very doable and encouraged.

Appendix A Talon SRX CAN Bus Protocol

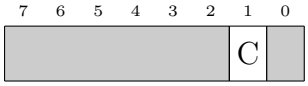
Talon SRXs communicate on the CAN bus at 1Mbps. All messages sent to and from the Talons use extended 29-bit identifiers, and the low 6 bits of these identifiers are used as a talon device ID. Following is a description of the different messages used by the Talon. The indicated message IDs have 6 bits of padding, so they can be bitwise or-ed with a device ID (for example a `CONTROL_5` message to a Talon with device ID 12 would have a message ID of 0x0204010C). Please not that this description is not a complete specification and only represents our understanding of the protocol based on the existing HERO board implementation.

A.1 `CONTROL_1` (ID: 0x02040000)

It is unclear from the HERO board C# implementation exactly what this message does, however it is sent with a value of all zeros at a period of 50ms so that is what our implementation does as well.

A.2 `CONTROL_3` (ID: 0x02040080)

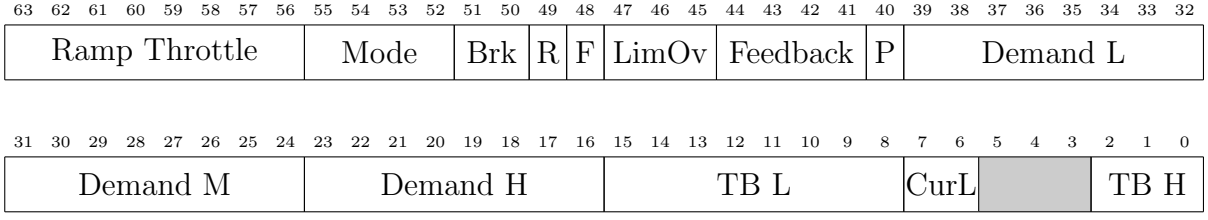
Unlike the other control and status messages, this message is not sent periodically, and seems only to be used to clear fault flags on the Talon.



Bit 1 **C**: Clear all sticky faults on the Talon

A.3 `CONTROL_5` (ID: 0x02040100)

The `CONTROL_5` is perhaps one of the most important messages for controlling the talon as it contains the setpoint and control mode. As with most of the control and status messages, it is intended to be sent periodically. A typical period for this message is 10ms.



Bits 63:56 **Ramp Throttle**:

Bits 55:52 **Mode**: Select the control mode to use
- 0000: Percent Vbus
- 0001: Position
- 0010: Speed
- 0011: Current
- 0100: Voltage
- 0101: Follower
- 0110: Motion Profile
- 0111: Motion Magic
- 1111: Disabled

Bits 51:50 **Brk**: Set the behavior for neutral throttle
- 00: Use default setting from flash configuration
- 01: Coast
- 10: Brake

Bit 49 **R**: Closed loop throttle reverse
- 0: Motor throttle not reversed in closed loop modes
- 1: Motor throttle multiplied by -1 in closed loop modes

Bit 48 **F**: Feedback sensor direction
- 0: Feedback sensor readings preserved
- 1: Feedback sensor readings multiplied by -1

Bits 47:45 **LimOv**: Overide limit switch enable settings
- 001: Use default settings from flash configuration
- 100: Disable both forward and reverse limit switches
- 101: Disable forward limit switch, enable reverse limit switch
- 110: Enable forward limit switch, disable reverse limit switch
- 111: Enabel both forward and reverse limit switches

Bits 44:41 **Feedback**: Select the feedback device to be used for cloosed loop control modes
- 0000: Quadrature Encoder
- 0010: Analog Potentiometer
- 0011: Analog Encoder
- 0100: Encoder (Rising Edge)
- 0101: Encoder (Falling Edge)
- 0110: CTRE Magnetic Encoder (Relative)
- 0111: CTRE Magnetic Encoder (Absolute)
- 1000: Pulse Width

Bit 40 **P**: Profile select
- 0: Use profile 0
- 1: Use profile 1

Bits 39:16 **Demand L**, **Demand M**, **Demand H**: Low, medium, and high bytes of the demand or setpoint.

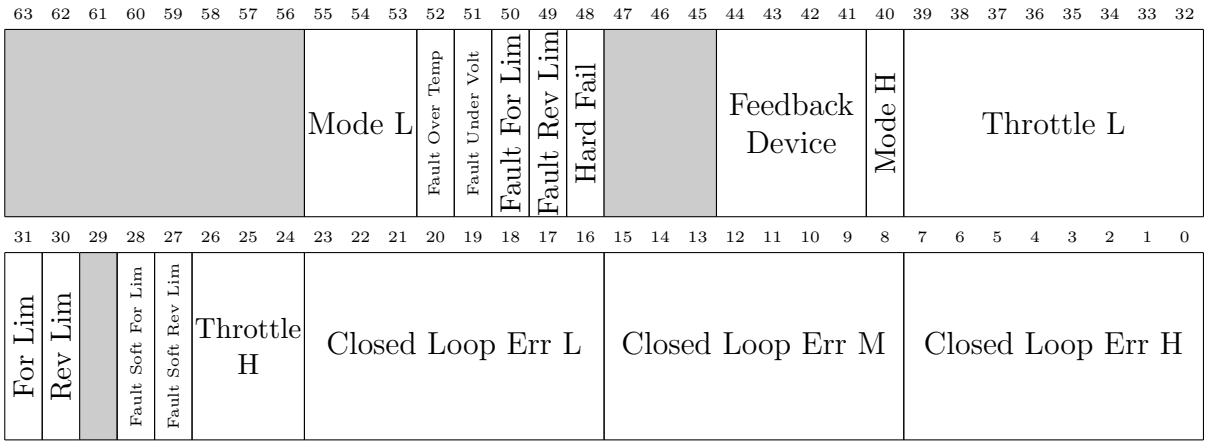
Bits 15:8 **TB L**: Low 8 bits of throttle bump value (it is unclear exactly what this value does however)

Bits 7:6 **CurL**: Current limit enable
- 00: Disable current limiting

01: Enable current limiting

Bits 2:0 **TB H**: High 3 bits of throttle bump value (it is unclear exactly what this value does however)

A.4 STATUS_1 (ID: 0x02041400)



Bits 55:53 **Mode L**: Low bits of currently selected control mode. When combined with the high bit in **Mode H**, represents one of the following:

0000: Percent Vbus
0001: Position
0010: Speed
0011: Current
0100: Voltage
0101: Follower
0110: Motion Profile
0111: Motion Magic
1111: Disabled

Bit 52 **Fault Over Temp**: Fault flag for over temperature

Bit 51 **Fault Under Volt**: Fault flag for under voltage

Bit 50 **Fault For Lim**: Fault flag for forward limit switch

Bit 49 **Fault Rev Lim**: Fault flag for reverse limit switch

Bit 48 **Hard Fail**: Fault flag for hardware failure

Bits 44:40 **Feedback Device**: The device currently being used for feedback in closed loop control modes

0000: Quadrature Encoder
0010: Analog Potentiometer
0011: Analog Encoder

0100: Encoder (Rising Edge)
0101: Encoder (Falling Edge)
0110: CTRE Magnetic Encoder (Relative)
0111: CTRE Magnetic Encoder (Absolute)
1000: Pulse Width

Bit 40 **Mode H**: High bit of currently selected control mode (see **Mode L**)

Bits 39:32 **Throttle L**: Low bits of the currently applied throttle. When combined with the high bits in **Throttle H**, represents the currently applied throttle as an integer from -1023 to 1023 where 0 is neutral throttle and either extreme is applying bus voltage in that direction.

Bit 31 **For Lim**
0: Forward limit switch not closed
1: Forward limit switch closed

Bit 30 **Rev Lim**
0: Reverse limit switch not closed
1: Reverse limit switch closed

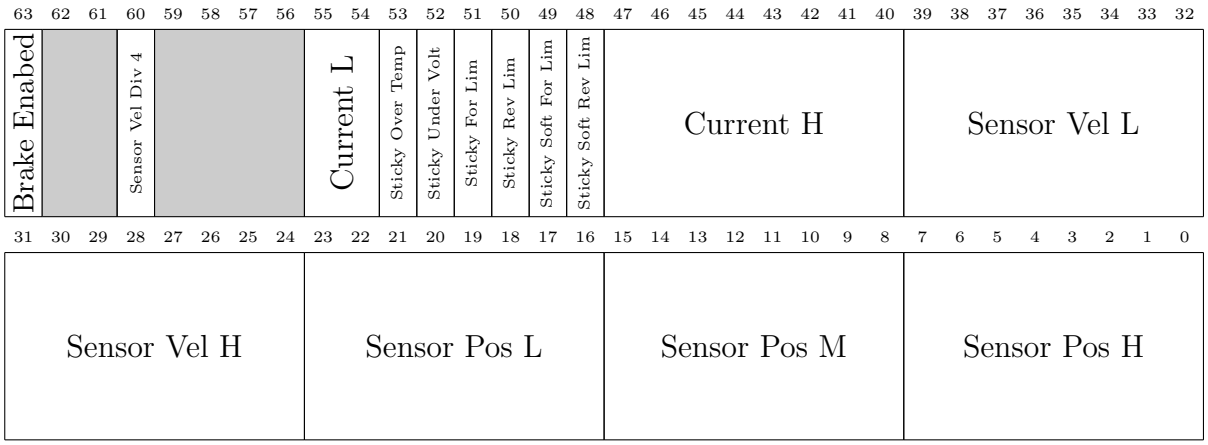
Bit 28 **Fault Soft For Lim**: Fault flag for soft forward limit

Bit 27 **Fault Soft Rev Lim**: Fault flag for soft reverse limit

Bits 26:24 **Throttle H**: See **Throttle L**

Bits 23:0 **Closed Loop Err L**, **Closed Loop Err M**, **Closed Loop Err H**: Represents the low, medium, and high bits of the current closed loop error.

A.5 STATUS_2 (ID: 0x02041440)



Bit 63 **Brake Enabled**

Bit 60 **Sensor Vel Div 4**: If this bit is set, then the sensor velocity in **Sensor Vel L** and **Sensor Vel H** should be multiplied by 4 to get the actual velocity.

Bits 55:54 **Current L**: Low bits of the current reading. When combined with **Current H** represents the current reading in Amperes in unsigned 7.3 fixed point notation.

Bit 53 **Sticky Over Temp**: Fault flag for sticky over temeprature fault

Bit 52 **Sticky Under Volt**: Fault flag for sticky under voltage fault

Bit 51 **Sticky For Lim**: Fault flag for sticky forward limit fault

Bit 50 **Sticky Rev Lim**: Fault flag for sticky reverse limit fault

Bit 49 **Sticky Soft For Lim**: Fault flag for sticky soft forward limit fault

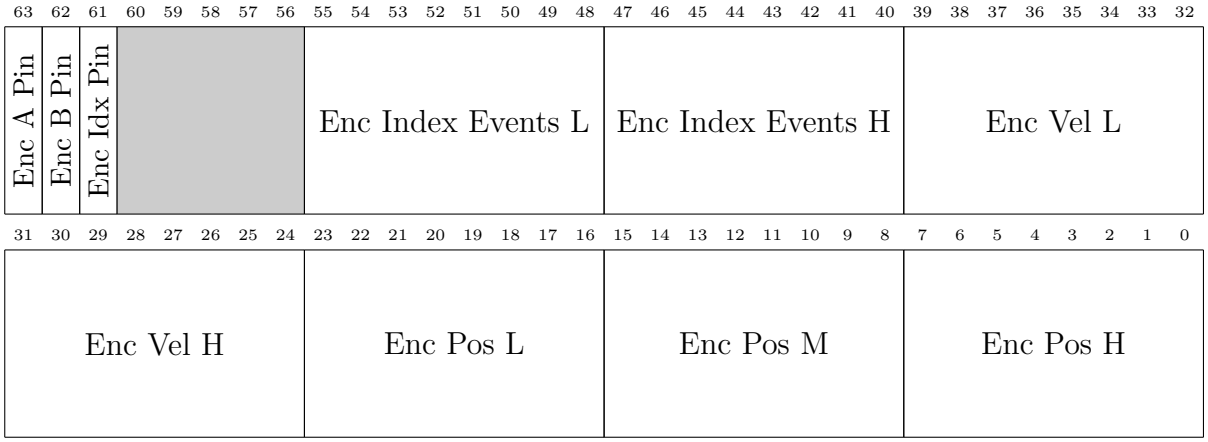
Bit 48 **Sticky Soft Rev Lim**: Fault flag for sticky soft reverse limit fault

Bits 47:40 **Current H**: See **Current L**

Bits 39:24 **Sensor Vel L**, **Sensor Vel H**: The low and high bits of the sensor velocity reading. See also **Sensor Vel Div 4**

Bits 23:0 **Sensor Pos L**, **Sensor Pos M**, **Sensor Pos H**: Low, medium, and high bits of the current sensor position reading

A.6 STATUS_3 (ID: 0x02041480)



Bit 63 **Enc A Pin**: Current status of the Talon's quadrature encoder A pin

Bit 62 **Enc B Pin**: Current status of the Talon's quadrature encoder B pin

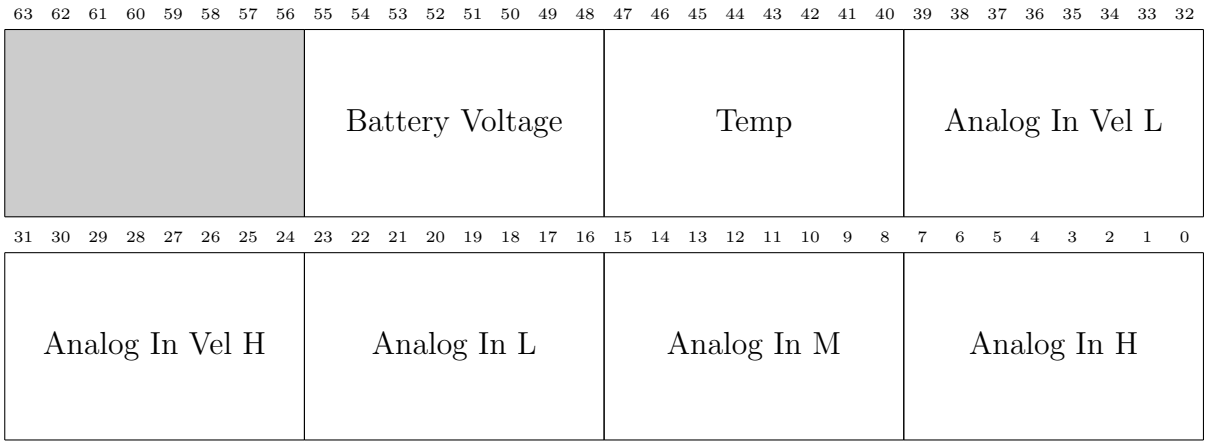
Bit 61 **Enc Idx Pin**: Current status of the Talon's quadrature encoder index pin

Bits 55:40 **Enc Index Events L**, **Enc Index Events H**: Low and high bits of the count of rising edges that have been counted on the index pin

Bits 39:24 **Enc Vel L**, **Enc Vel H**: Low and high bits of the current encoder velocity reading

Bits 23:0 **Enc Pos L**, **Enc Pos M**, **Enc Pos H**: Low, medium, and high bits of the current encoder position reading

A.7 STATUS_4 (ID: 0x020414C0)



Bits 55:48 **Battery Voltage**: Input voltage to the talon. To convert this value to volts use the following conversion $0.05 \cdot raw + 4$

Bits 47:40 **Temp**: Current temperature of the talon. To convert this value to degrees celsius, use the following conversion $0.645161290322581 \cdot raw - 50$

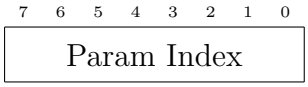
Bits 39:24 **Analog In Vel L**, **Analog In Vel H**: Low and high bits of the rate of change of the analog in sensor reading

Bits 23:0 **Analog In L**, **Analog In M**, **Analog In H**: Low, medium, and high bits of the analog sensor reading. The ADC in the talon is only 10 bits; the upper 14 bits of this reading track overflows and underflows

A.8 PARAM_REQUEST (ID: 0x02041800)

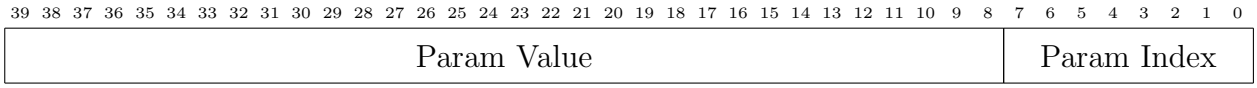
This message is sent to a Talon to request the current value of a parameter. The Talon should then respond with a PARAM_RESPONSE message that contains the current value of the

parameter.



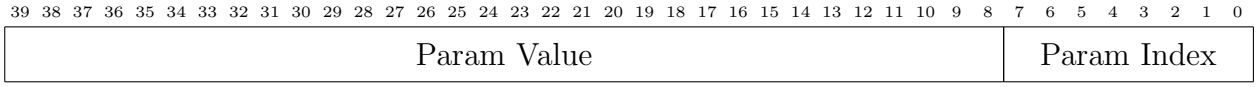
Bits 7:0 **Param Index**: Integer identifying the parameter requested (see A.11 for a list of parameters)

A.9 PARAM_RESPONSE (ID: 0x02041840)



Bits 39:8 **Param Value**: Value of the parameter (see A.11 for how to interpret these bits)
Bits 7:0 **Param Index**: Integer identifying the parameter the value is for (see A.11 for a list of parameters)

A.10 PARAM_SET (ID: 0x02041880)



Bits 39:8 **Param Value**: Value of the parameter (see A.11 for how to interpret these bits)
Bits 7:0 **Param Index**: Integer identifying the parameter the value is for (see A.11 for a list of parameters)

A.11 Parameters

The following is a list of all the parameters from the HERO board Talon implementation. Unless otherwise specified, the format of a parameter is a 32 bit integral value. Other possible parameter formats are unsigned 10.22 fixed point (u10.22), signed 10.22 fixed point (s10.22), unsigned 0.8 fixed point (u0.8), and unsigned 8.8 fixed point (u8.8).

Parameter Name	Index	Format	Units
eProfileParamSlot0_P	1	u10.22	
eProfileParamSlot0_I	2	u10.22	
eProfileParamSlot0_D	3	u10.22	
eProfileParamSlot0_F	4	s10.22	
eProfileParamSlot0_IZone	5		
eProfileParamSlot0_CloseLoopRampRate	6		
eProfileParamSlot1_P	11	u10.22	
eProfileParamSlot1_I	12	u10.22	
eProfileParamSlot1_D	13	u10.22	
eProfileParamSlot1_F	14	s10.22	
eProfileParamSlot1_IZone	15		
eProfileParamSlot1_CloseLoopRampRate	16		
eProfileParamSoftLimitForThreshold	21		
eProfileParamSoftLimitRevThreshold	22		
eProfileParamSoftLimitForEnable	23		
eProfileParamSoftLimitRevEnable	24		
eOnBoot_BrakeMode	31		
eOnBoot_LimitSwitch_Forward_NormallyClosed	32		
eOnBoot_LimitSwitch_Reverse_NormallyClosed	33		
eOnBoot_LimitSwitch_Forward_Disable	34		
eOnBoot_LimitSwitch_Reverse_Disable	35		
eFault_OverTemp	41		
eFault_UnderVoltage	42		
eFault_ForLim	43		
eFault_RevLim	44		
eFault_HardwareFailure	45		
eFault_ForSoftLim	46		
eFault_RevSoftLim	47		
eStckyFault_OverTemp	48		
eStckyFault_UnderVoltage	49		
eStckyFault_ForLim	50		
eStckyFault_RevLim	51		
eStckyFault_ForSoftLim	52		
eStckyFault_RevSoftLim	53		
eAppliedThrottle	61		
eCloseLoopErr	62		
eFeedbackDeviceSelect	63		
eRevMotDuringCloseLoopEn	64		
eModeSelect	65		
eProfileSlotSelect	66		
eRampThrottle	67		
eRevFeedbackSensor	68		

eLimitSwitchEn	69		
eLimitSwitchClosedFor	70		
eLimitSwitchClosedRev	71		
eSensorPosition	73		
eSensorVelocity	74		
eCurrent	75		
eBrakeIsEnabled	76		
eEncPosition	77		
eEncVel	78		
eEncIndexRiseEvents	79		
eQuadApin	80		
eQuadBpin	81		
eQuadIdxpin	82		
eAnalogInWithOv	83		
eAnalogInVel	84		
eTemp	85		
eBatteryV	86		
eResetCount	87		
eResetFlags	88		
eFirmVers	89		
eSettingsChanged	90		
eQuadFilterEn	91		
ePidIaccum	93		
eStatus1FrameRate	94		ms
eStatus2FrameRate	95		ms
eStatus3FrameRate	96		ms
eStatus4FrameRate	97		ms
eStatus6FrameRate	98		
eStatus7FrameRate	99		
eClearPositionOnIdx	100		
ePeakPosOutput	104		
eNominalPosOutput	105		
ePeakNegOutput	106		
eNominalNegOutput	107		
eQuadIdxPolarity	108		
eStatus8FrameRate	109		ms
eAllowPosOverflow	110		
eProfileParamSlot0_AllowableClosedLoopErr	111		
eNumberPotTurns	112		
eNumberEncoderCPR	113		
ePwdPosition	114		
eAinPosition	115		

eProfileParamVcompRate	116	u0.8	V/ms
eProfileParamSlot1_AllowableClosedLoopErr	117		
eStatus9FrameRate	118		ms
eMotionProfileHasUnderrunErr	119		
eReserved120	120		
eLegacyControlMode	121		
eMotMag_Accel	122		
eMotMag_VelCruise	123		
eStatus10FrameRate	124		ms
eCurrentLimThreshold	125		
eCustomParam0	137		
eCustomParam1	138		
ePersStorageSaving	139		
eClearPositionOnLimitF	144		
eClearPositionOnLimitR	145		
eNominalBatteryVoltage	146	u8.8	V
eSampleVelocityPeriod	147		
eSampleVelocityWindow	148		

Table 2: List of Talon SRX parameters

References

A. Shkolnik and R. Tedrake, "Inverse kinematics for a point-foot quadruped robot with dynamic redundancy resolution." DOI: 10.1109/ROBOT.2007.364146. (2007).

C. RunBin et al, "Inverse Kinematics of a New Quadruped Robot Control Method," International Journal of Advanced Robotic Systems, vol. 10, (1), pp. 46. (2013).

D. Belter et al, "Dynamic simulation of legged robots using a physics engine." (2014).

D. Pongas, M. Mistry and S. Schaal, "A robust quadruped walking gait for traversing rough terrain." DOI: 10.1109/ROBOT.2007.363192. (2007).

F. L. Moro et al, "Horse-like walking, trotting, and galloping derived from kinematic Motion Primitives (kMPs) and their application to walk/trot transitions in a compliant quadruped robot," Biological Cybernetics, vol. 107, (3), pp. 309-320. (2013).

H. Hwang and Y. Youm, "Dynamic crawl gait algorithm for quadruped robots." DOI: 10.1109/IROS.2008.4650779. (2008)

J. E. Pratt and B. T. Krupp, "Series Elastic Actuators for legged robots," in Unmanned Ground Vehicle Technology VI (G. R. Gerhart, C. M. Shoemaker, and D. W. Gage, eds.), vol. 5422 of Proc. SPIE, pp. 135–144, Sept. 2004.

J. Lee and S. Song, "Path planning and gait of walking machine in an obstacle-strewn environment," Journal of Robotic Systems, vol. 8, (6), pp. 801-827, (1991).

J. R. Rebula et al, "A controller for the LittleDog quadruped walking on rough terrain." DOI: 10.1109/ROBOT.2007.363191. (2007)

J. Rudy, "Zero-Moment Point Walking Controller for Humanoid Walking Using Darwin-OP." Dep. Aer Mech Eng, U. Notre Dame. (2014).

M. H. P. Dekker, "Zero-moment point method for stable biped walking." Eindhoven University of Technology. (2009).

M. H. Raibert, "Trotting, pacing and bounding by a quadruped robot," Journal of Biomechanics, vol. 23, pp. 79,83-81,98. (1990).

R. Kurazume, K. Yoneda and S. Hirose, "Feedforward and Feedback Dynamic Trot Gait Control for Quadruped Walking Vehicle," Autonomous Robots, vol. 12, (2), pp. 157-172. (2002).

S. Ivaldi et al, "Tools for simulating humanoid robot dynamics: A survey based on user feedback." DOI: 10.1109/HUMANOIDS.2014.7041462. (2014).

X. Chen et al, "A Real-Time Kinematics on the Translational Crawl Motion of a Quadruped Robot," Journal of Intelligent and Robotic Systems, vol. 29, (2), pp. 111-131. (2000).

Tenzer, Yaroslav, Leif P. Jentoft, and Robert D. Howe. 2014. "The Feel of MEMS Barometers: Inexpensive and Easily Customized Tactile Array Sensors." IEEE Robot. Automat. Mag. 21 (3) (September): 89–95. doi:10.1109/mra.2014.2310152.

M. Y. Chuah, M. Estrada and S. Kim, "Composite force sensing foot utilizing volumetric

displacement of a hyperelastic polymer," 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, 2012, pp. 1963-1969. doi: 10.1109/IROS.2012.6386239

S. Youssefian, N. Rahbar and E. Torres-Jara, "Contact Behavior of Soft Spherical Tactile Sensors," in IEEE Sensors Journal, vol. 14, no. 5, pp. 1435-1442, May 2014. doi: 10.1109/JSEN.2013.2296208

M. Y. Chuah and S. Kim, "Enabling Force Sensing During Ground Locomotion: A Bio-Inspired, Multi-Axis, Composite Force Sensor Using Discrete Pressure Mapping," in IEEE Sensors Journal, vol. 14, no. 5, pp. 1693-1703, May 2014. doi: 10.1109/JSEN.2014.2299805