4-29-2008

Final Report: Tree Climbing Robot

Stephen Arnold
Trinity University

Robert Lewis
Trinity University

Matthew Meador
Trinity University

Amanda Richter
Trinity University

Matthew Saunders
Trinity University

TRINITY UNIVERSITY

Final Report

ENGR-4382

4/29/2008

Tree Climbing Robot Group

Stephen Arnold
Robert Lewis
Matthew Meador
Amanda Richter
Matthew Saunders

Dr. Farzan Aminian, Advisor

ABSTRACT: The motivation for this project was to minimize the injury and fatality rate of loggers across the world due to inclement weather, structurally unsound tress, and bulky equipment used to trim trees by designing a robot with the ability to climb trees and trim branches. Due to budget and time constraints, the scope of the project was limited to autonomously climbing and descending a tree trunk while avoiding branches; however, the possibility of future modifications was considered in the design. Design criteria included maneuverability, reversibility, simplicity, vertical speed, the ability to climb a tree of fixed diameter, and the ability to carry extra weight. The final design was a "caterpillar"-style robot, and the major modular subsystems of the design included clamping, extension, rotation, and branch detection. The final design, construction, and testing of each of these subsystems is discussed, as well as the testing of the system as a whole. Ultimately, the design was deemed a success, with the exception of the failure of the rotation system.

1 Executive Summary

The success of this project depended on the robot's capacity to autonomously climb trees while detecting and maneuvering around branches. Motivation for the project stemmed from an attempt to minimize injuries sustained by loggers due to traversing dangerous heights in structurally unsound trees and harnessing bulky equipment during possible inclement weather conditions such as wind and rain. Limitations in budget and time constrained the project to be designed for a tree diameter of thirteen inches, and even though the diameter for this project yields no flexibility for the diversity a natural forest, it was felt that this project would provide motivation for future projects similar to the group's tree climbing robot. Other criteria included maneuverability around branches, the ability to reverse the direction of climbing, simplicity of design, vertical speed, and the ability to carry extra weight.

The mechanical construction of the robot consisted of three unique subsystems. First, a simple clamping system was devised using a linear actuator to lever the arms of each clamp, allowing the robot to grip the tree. Next, the extension system incorporated another linear actuator with a different extension length and force than the clamping actuators that would extend and contract the two identical clamping subsystems. Finally, a gear and chain system was implemented to rotate a tread and roller system on each clamp. A motor, which coordinated with a controlling circuit and program, rotated through a gear box, which increased the torque to overcome the friction between the robot and the tree. These treads conformed to the tree and allowed the entire robot body to roll around the tree trunk, while at the same time maintaining enough frictional force to stay attached to the tree.

The electrical design met the criteria for an automated robot with the capacity to navigate the tree in both directions. Control circuits applied voltages to the actuators and motors, allowing them to move autonomously. Sonar detectors were placed on the top clamp of the robot, allowing for the detection of branches as the robot climbed the tree, as well as a real time map that allowed the robot to navigate the tree in reverse. The rotation system's motors also followed inputs provided by programming and circuits designed to rotate the robot when a branch impeded the upward and downward motion of the robot. Finally, all circuits implemented communicated with a master computer which oversaw the delicate movements of the robot.

Each subsystem was tested modularly as it was constructed, and then the subsystems were put together to form the robot. The entire robot was then tested to ensure that it met the working criteria. The testing showed that all of the subsystems except rotation worked properly, and the design was ultimately deemed successful.

2 Table of Contents

3 Table of Figures

4 Table of Tables

5 Introduction

The current logging industry requires loggers to trim branches off of large trees to prevent both wind damage to the trunks and fall damage when they are cut. Motivation for this project stemmed from an attempt to minimize injuries sustained by loggers due to traversing dangerous heights in structurally unsound trees and harnessing bulky equipment during possible adverse weather conditions such as wind and rain. Due to time and budget constraints, the scope of this project was limited to the development of an autonomous robot capable of scaling a tree and avoiding collisions with branches. Ultimately, the purpose of the project was to design, build, and test a robot that was able to traverse a tree trunk to a height preset by the operator and then return to the ground, all the while maneuvering around obstacles in its path. This robotic platform could then later be modified to accommodate various modifications, such as the addition of a saw, allowing it to perform tasks that currently put a human at risk of injury or even death.

At the beginning of the design process, several criteria for success were identified. The robot needed to be agile enough to maneuver around branches, yet robust enough to support its own weight and an additional weight of 15 pounds. This extra payload was incorporated to ensure that the design would be capable of accommodating future modifications. The robot needed to be able to climb a tree of a fixed diameter at a reasonable speed, as well as reverse its climbing and maneuver back down the tree; if it were to climb to the top of a tree and not be able to get back down, a person would have to climb up after it, defeating the purpose of the robot. Due to the limited budget and timeline of the project, the design also needed to be as simple as possible, while still encompassing all of the goals. The weighting of each of these criteria is shown in

Table 1.

Table 1. Weighted Working Criteria

Working Criteria	**Weight (%)**
Maneuverability around branches	30
Fairly simple design	20
Climb fixed diameter trunk	15
Reversible climbing	15
Extra weight	15
Vertical speed	5

To achieve these goals, the group settled on a basic concept of two separate clamps connected by a center system that was extended and contracted in a caterpillar-like motion, allowing the robot to alternately push and pull itself up the tree. A sonar array and a system of treads and rollers gave the robot the ability to detect branches or obstacles in its path and rotate around the tree to avoid them. All of the subsystems were then integrated together to be controlled by a master computer.

The robot design was broken into four modular subsystems: clamping, rotation, vertical extension, and branch detection. After coming up with an initial design for each subsystem, the group constructed, tested, and revised each module separately. Once each module was working properly, they were combined to form the final system. The robot as a whole was then tested to ensure that the design criteria were met.

6 Construction and Module Testing

As with any project, once the construction phase began the initial prototype design began to change. Both the mechanical and electrical components of the design went through extensive evolution throughout the process of testing and revisions. The final design ended up looking similar to the original design, as seen by a comparison of Figure 1 andFigure 2, but the goals of each module were accomplished in slightly varied forms.

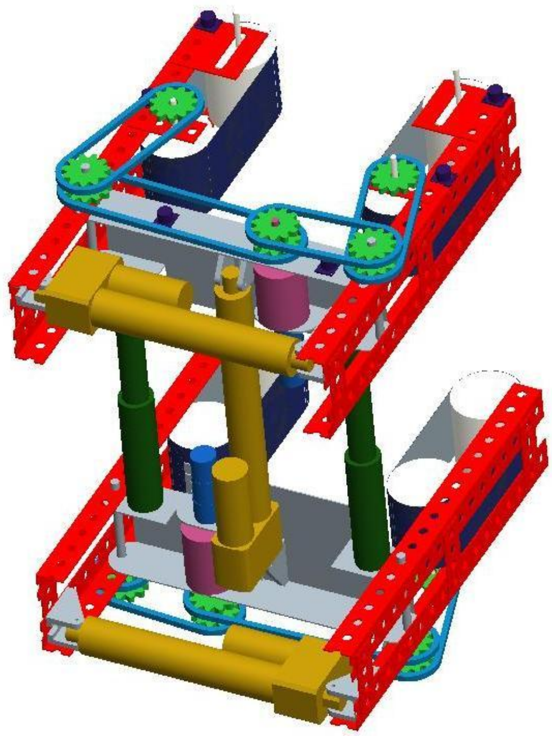


Figure 1. Original Project Model in ProE

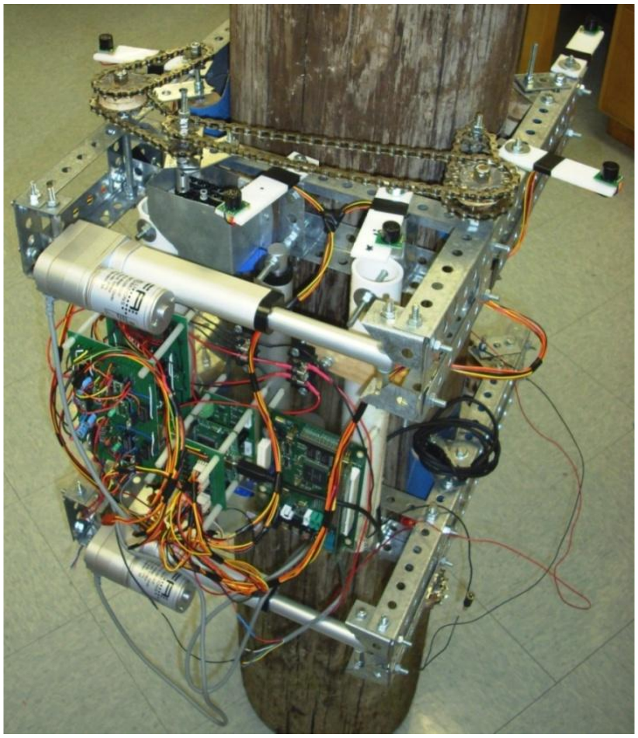


Figure 2. Final Robot Design

6.1 Clamping System

A strong clamping system was necessary for the robot to be able to support its weight as it climbed up the tree. The final design decision was based on the system's simplicity and its ability to conform to the tree and maximize the surface area of the robot in contact with the tree.

6.1.1 Final Design

On a basis of simplicity and cost effectiveness, the group decided that a linear actuator provided the best solution for the clamping system. The ideas of using pneumatics or hydraulics were briefly considered, but were disregarded due to the cost, construction time, and additional weight that would be added to the robot by the required supply subsystems. Each clamp consisted of two arms connected by a linear actuator at one end and also fixed to free rotating shoulder joints, so that when the actuator mounted behind these joints extended, it caused the arms to rotate about the shoulder joints and compress and grip the tree, as demonstrated by the arrows in Figure 3.

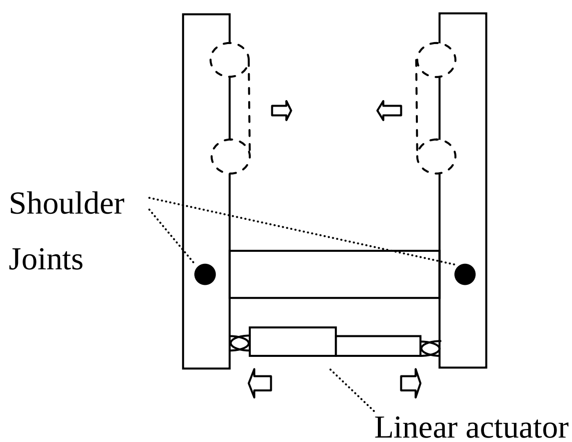


Figure 3. Prototype clamping system

This actuator was sized using basic force and moment balances based on geometry and the estimated weight of the robot to ensure that the clamp would be strong enough to grip the tree without slipping. The applied force was increased to an acceptable level by lengthening the distance between the shoulder joint and the point of applied force from the actuator, as well as increasing the strength of the actuator to amplify the moment. This shoulder to actuator length also governed how far back the arms could swing away from the tree, which was an important factor in the design. Refer to Appendix A for detailed mathematical modeling code and solutions.

The final clamping design is shown in Figure 4 and Figure 5.

Figure 4. Final Clamping System, Single Clamp

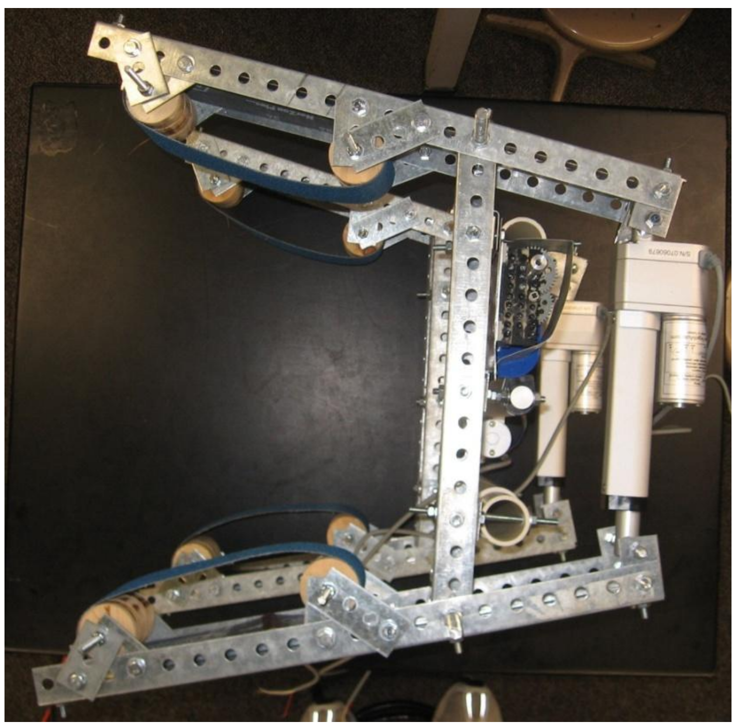


Figure 5. Final Clamping System, Both Clamps

The two clamps were constructed so that they were mirror images of each other in order to maintain system symmetry when the clamps were attached to each other via the extension system. The arms and base were constructed out of “L” shaped metal with predrilled holes spaced one inch apart. Both legs of the "L" were one inch wide. They were supported by flat pieces of metal which were one inch wide with predrilled holes half an inch apart. The shoulder joints were constructed with pieces of 3/8"-16 threaded rod. Lock nuts were used to fasten the threaded rod in place. These served to properly secure the shoulder joints in place without having to tighten the nuts so tight that the joint was unable to hinge. Each joint was fastened by 1" long, 3/16" wide bolts and nuts.

6.1.2 Module Testing

The construction of the robot began with the clamping system, and upon completion the first clamp was tested to see if it could in fact attach to the tree. The actuator was attached to a variable power supply set to 12V to generate a clamping action. Once attached to the tree, it was found that the base of the robot came into contact with the tree. This was corrected by extending the length of each arm by two inches, with all the extended length added between the shoulders and the tread wheels. It was also found that tread's protrusion from the arms was greater than expected in the original design. In order to keep the joints at 90° when clamped, the distance between the arms had to be increased. To do this, extenders were designed to connect the actuators to the arms. This kept the strength ratio equal to the original design. The joint between the actuator and arm is shown in Figure 6.

Figure 6. Joint Between Arm and Linear Actuator

Once the revised clamp was fully tightened to the tree, a force meter was attached to the back plate and a downward force of 50 lbs was applied to ensure that the single clamp could more than adequately support the entire robot and additional weight of 15 pounds. Although the roller axles were not connected to a motor, the robot was manually rotated around the tree to ensure that the grip did not inhibit rotational motion and that there was minimal slip between the rollers and tread. The first arms worked very well and held considerably more force than was expected. This test was then repeated with the second clamp when its construction was completed. The second clamp's testing was also successful with no slipping occurring when additional weight was added to it.

Once the two clamps were connected via the extension system (to be discussed later in the report), the completed robot was once again subjected to a clamping test to ensure that it would not slip as a whole robot. The robot was first clamped to the tree with both clamps and manual rotation was once again applied to ensure that the robot would not slip during a rotation. The robot was able to rotate a full 360° without any noticeable change in gripping capability. The next full scale test conducted consisted of extending the robot to its full length and testing each clamp's ability to individually support the weight of the entire system and keep the robot from falling. This test was performed at full extension because that was when the rotational moment due to the loose clamp had the greatest effect on the robot. First the bottom clamp was tested to ensure that it could withstand the full weight of the robot from the top. This test showed a stable robot without any slippage downward on the tree. It also showed that there was not any leaning toward or away from the tree. Next the top clamp was tested. As with the bottom clamp's test, this showed that the top clamp was able to withstand any downward force and remain securely clamped on the tree. This test did however show the loose clamp on the bottom leaning into the tree. After re-gripping the tree, this problem rectified itself. The robot's clamping system proved successful with these tests.

6.2 Rotation System

The goal of the rotational module was to allow the robot to travel around the tree to position itself to avoid branches. The difficulty in designing this system was that there was an inverse relationship between the ability to rotate and the ability to stay clamped onto the tree; the more securely the robot was clamped to ensure no slippage, the more torque was required to rotate the robot around the tree. This tricky balance of forces was an integral part of the design throughout the project.

6.2.1 Final Design

After numerous debates between treads and wheels, it was decided that a compromise of two small tread arrangements on each set of arms would work best. As seen in Figure 7, an arrangement of bike gears and chain powered by a DC motor and gear box was designed to drive the rollers closest to the shoulders.

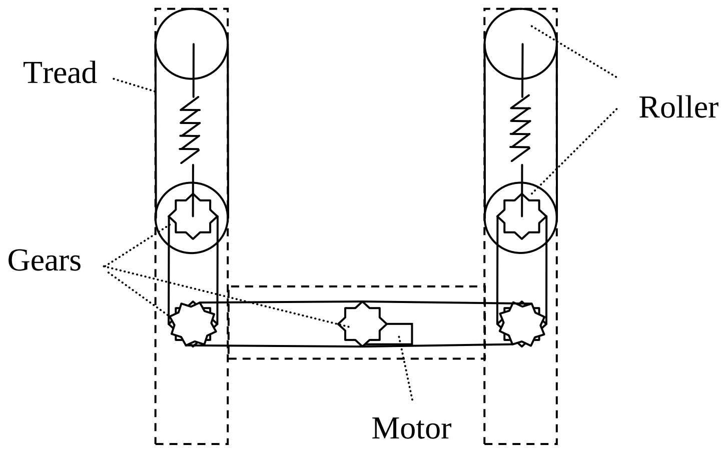


Figure 7. Prototype Rotation System

A gear box was used to amplify the torque from the DC motor because a specific motor had already been chosen due to familiarity and available feedback options. Two bike gears were attached on the same axis to the output shaft of the gear box. Separate bike chain loops connected these two gears to one of two more gears sharing a same axis mounted on each shoulder joint. The other gear on each joint turned a secondary chain loop, which powered a gear mounted on each of the closer rollers. The rotation of the roller then turned the tread, causing the robot to travel around the circumference of the tree. The intermediary gears on the shoulder joints were added to the design to eliminate the need for a device to keep the chain taut, since a direct motor to roller system would require different lengths of chain to remain taut as the arms were swinging to clamp or release.

During the design of this system, it was decided that treads had an advantage over wheels because of the difference in contact area between the robot and the tree, but the fallback for this application was that the contact area was cylindrical. To create this curved surface, the tread system was designed to shape to the tree as the arms were clamped. As seen in Figure 8, the tread systems were designed to conform to the curve of the tree, with the tread pulled taut by the distance between the two fixed sets of wheels. This allowed for a maximum surface area of the tread to be in contact with the tree.

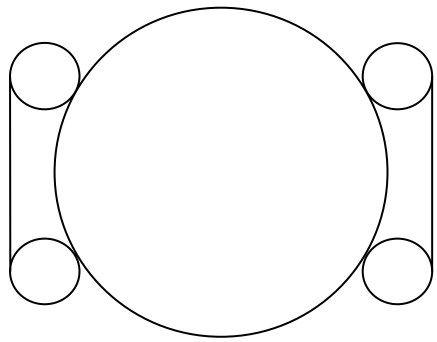


Figure 8. Compressed Tread Systems for Curved Contact Surface

The tread system was constructed with a series of three 2” diameter wooden cylinders with 1/4”-20 threaded rods through the centers of the cylinders. Nuts were placed between each cylinder in order to keep them from rotating extraneously from the rod. The tread around the two sets of cylinders is a piece of sandpaper designed for a belt sander. Because of this it is connected in a loop, and can fit around the inner and outer wheels of the tread system. The wheels were connected to the arms via the same flat metal as used to support the arms and shoulders. These were fastened to the arms loosely at first, but then two wheels were pulled apart as far possible in order to tighten the tread. Figure 9 shows the construction of one tread and roller system.

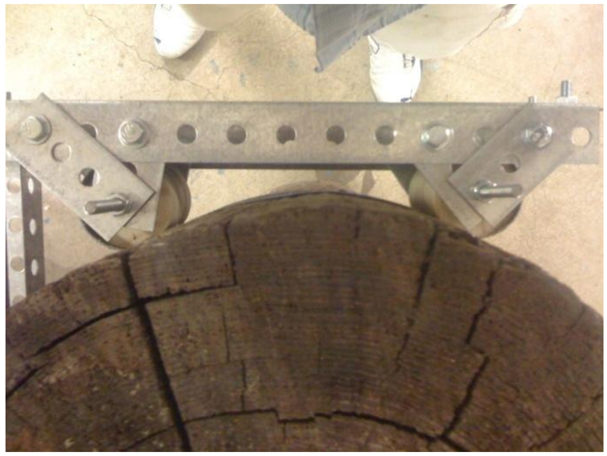


Figure 9. Fixed Axle Support and Tread Conformity

While being extended, the tread systems were fastened to the arms using the same bolts as used in the arms and base. Lock nuts were placed on the outside of the wheels in order to keep them from sliding up and down while being clamped. The bike gears were then fastened to the end of the threaded rod and held in place by a nut and jam nut combination.

The DC motor used to power the rotation system was originally planned to be mounted in the center of the robot. This task proved impossible, however, since the linear actuator used for the extension system had to be mounted in the center to ensure uniform extensions and contraction, and took up all of the space in this area. It was decided that the motors and their gear boxes be attached to the piping system's supports. Thus the motor was cantilevered to the body from its base. Having the rotation system off center did not affect the symmetry of the system, as the gear ratio remained the same between both arms of the clamp.

The biggest challenge in the rotational design was the way that the bike chain and gears were implemented throughout the system. After searching numerous bike shops around the area, no gears were found that were small enough to fit the rollers of the clamp. The smallest gear obtained would, if used, rub against the tree. To solve this problem, many alternatives were discussed. The easiest solution would have been to create larger wheels for the tread. This alternative was given so much credence that it was actually implemented briefly. However, it was not successful because the clamping system had been designed for the wheels already in place, and the larger wheels simply compromised the integrity of the clamping system. Ultimately, the only way to solve the problem within the project's budget was to custom make four bike gears small enough to fit the clamp. These gears, shown in Figure 10, have 8 teeth and were constructed by cutting out notches in washers to fit the necessary spacing to work correctly with a bike chain. The washers were then welded to ¼" nuts so that they could be fastened to the threaded rods on the tread system.

Figure 10. Custom Constructed Gears

Another challenge in the bike gear system was that there were still not enough other gears of any one size to complete the rest of the system. This problem was solved by connecting two of the same sized gears (13 teeth) to the gear box. Chains connect these gears to the gears on the shoulder joint of size 14 teeth. Each of these gears was fastened to one side of a wooden circle, with a 15-tooth gear fastened on the other side, as shown in Figure 11.

Figure 11. Rotation Gears

Finally these gears were connected by chain to the custom made gears which were mounted to the inner wheels of the tread system.

The original design called for the use of a hand drill gear box to step up the torque from the motor. Upon further research into this option, however, it was determined that this method would be too costly to implement, as it would have incurred a cost that was double the allocated funds.

The remaining type of gear boxes that were readily available were made for small motorized robots. This type of gear box would have worked as well, but upon closer inspection, it was found that these gear boxes were essentially constructed from Lego gears under a different name. The decision was made to construct gear boxes from Legos, since one of the group members offered to loan the necessary parts. This donation resulted in a much needed increase in budget for the project as a whole.

A gear train of 1:100 was constructed out of Legos, which were epoxied and screwed together, and mounted onto a sheet of steel along with the motor as seen in Figure 12.

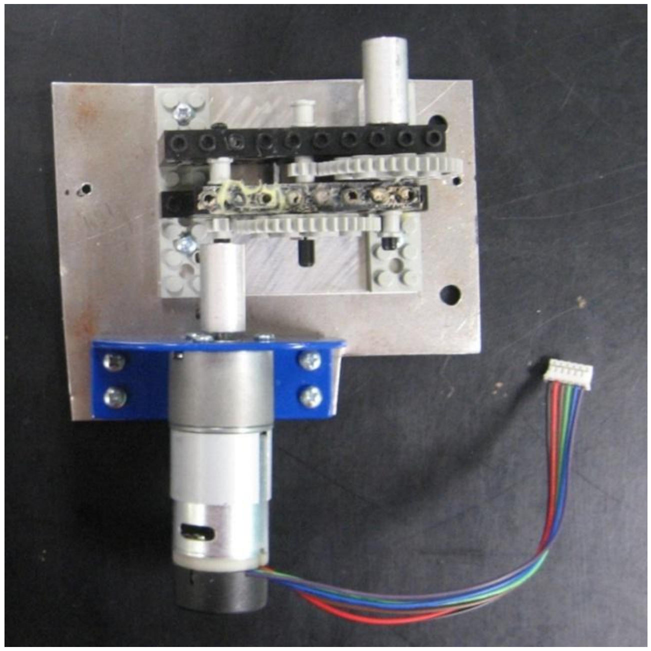


Figure 12. Constructed Gear Box

A coupling was made to attach the D shaft of the motor output to the input of the gear train. A secondary coupling was machined to convert the Lego shaft output of the gear train to a ¼"-20 threaded rod. Finally a sheet of aluminum was formed to create a casing to prevent any kind of debris from interfering with the gears.

The motor and gear box assembly was mounted to the robot as mentioned before. Another 1/4"-20 threaded rod was attached to the output of the gear box so that the bike gears could be attached. The system of one set of wooden cylinders and two bike gears was connected to the rod similarly to the 8-tooth gear mounted to the tread system. Regular nuts were fastened into the wooden circle, followed by jam nuts, followed by another set of regular nuts. This system was used to keep the gears from spinning independently from the rod. The gears located on the shoulder joints were the only gears that did not need to be securely fastened to the rod. In fact, a free spinning gear was preferred because the shoulder joint rod was used to support the robot's structure, and should not have had a torque applied to it. These gears were secured on the rod in the same way as the others, except that they were not tightened. This kept the gears in place, yet also allowed them to spin freely. Finally, bike chain was placed on the gears in the configuration mentioned before.

6.2.2 Module Testing

Unfortunately, testing on this system was never completed. During the construction of the system, it was found that the group had not been detailed enough in the design, leading to discrepancies in the calculations of chain length required to connect the gears. Bicycle chain can only be sized and connected in ½" increments, and it was found that the spacing between the fixed axles on which the gears were mounted did not line up correctly with the bike chain. The materials chose for the body, the L-bars with pre-drilled holes, did not allow for the flexibility required to place the gears at the correct distances from each other. Thus, when the chain loops were added to the system, they were either too tight and did not allow for any rotation, or they were too loose and slipped off the gears without allowing for any translation of motion. The group was never able to get the rotation system in working order.

6.3 Vertical Extension System

The purpose of the extension system was to allow the robot to climb up the tree, and then back down. This system allowed the robot to extend and contract so that it could climb in the same fashion as a caterpillar. The extension system had to be able to support half the robot's weight, as the other half was supported by the stationary clamp.

6.3.1 Final Design

After weighing several alternatives involving gears and scissor lifts, the use of electric linear actuators for the basis of the extension system was deemed the best solution to accomplish the module's goals. The final design centers on a linear actuator mounted perpendicularly between the two body sections, as shown in Figure 13.

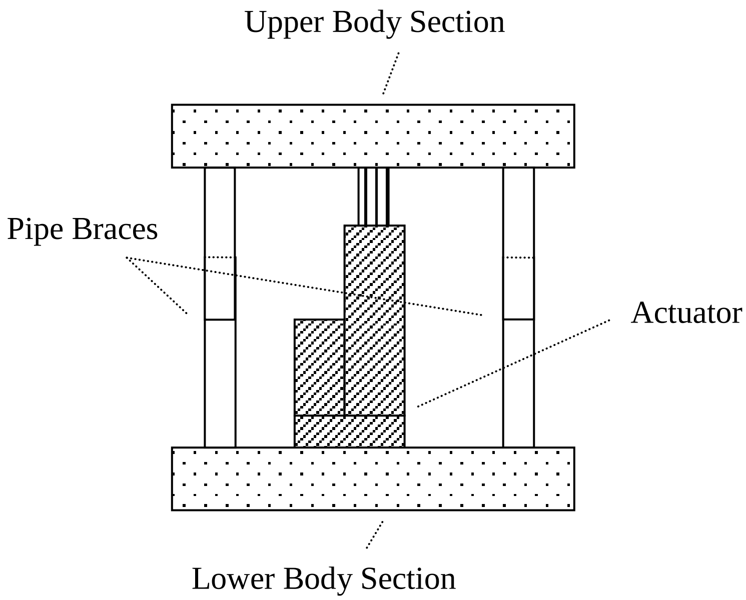
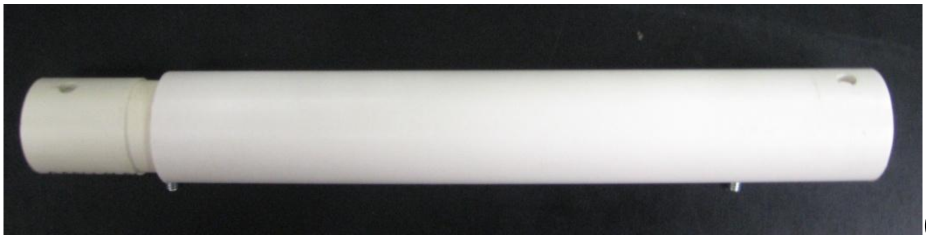


Figure 13. Prototype Extension System

Additional support was provided by a PVC pipe brace on either side of the actuator comprised of a larger diameter pipe with a slightly smaller diameter pipe inside to create a telescoping action. PVC was chosen for the support braces because of its relatively light weight and the ability to easily machine the tubing for increased structural support from the telescoped length. Using a rotary lathe, the outer radius of the one foot long segment of 1-5/8" outer diameter pipe was reduced by 2 mm to allow 10" to slide easily inside the slightly larger foot long section of 2" outer diameter pipe. Since the extension actuator extended a maximum of 8", the piping allowed for 2" of overlap at full extension. This machining created a tight telescoping support with relatively low friction. This telescoping is demonstrated in Figure 14.


(a)

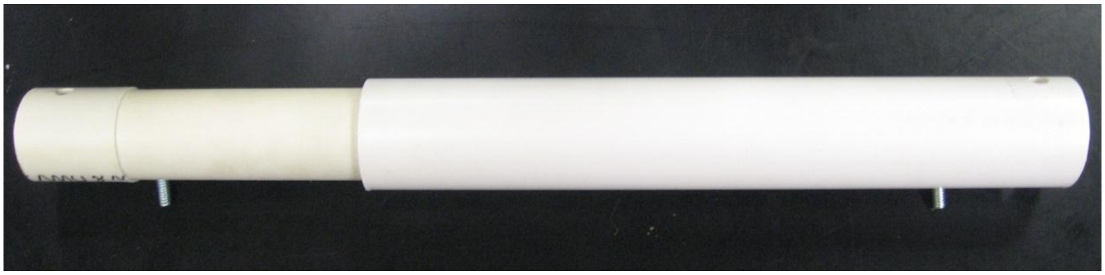

(b)

Figure 14. Telescoping Extension Support (a) Fully collapsed; (b) Partially extended

The final design of the extension system is shown in Figure 15.

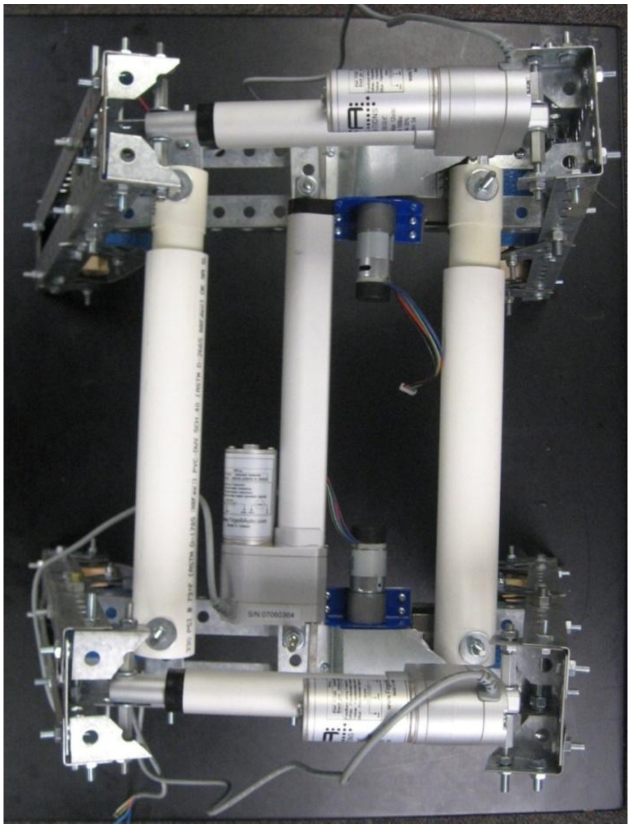


Figure 15. Final Extension System

The linear actuator and PVC braces were held firmly in place on both ends by “L” shaped brackets, lengths of ¼”-20 threaded rod, and nuts. The clamp structures were strengthened by adding support strips of “L” material to distribute the force from the actuator and piping to prevent the generation of a torque on the inner side of each clamp during extension. A Plexiglas backing to mount the electronics (not shown in the figure) was attached to the larger diameter piping by U bolts. It should be noted that the weight of the extension system was relatively balanced during construction by placing the weight of the actuator on the bottom clamp to offset the weight of the electronics attached to the top clamp. This balance of weight was crucial since the extension actuator had a strength of 20 lbs and each side weighed approximately 12 lbs.

6.3.2 Module Testing

During the design and comparison of the alternatives, the extension system was designed, but the actual connections were never thoroughly discussed. As a result of this, the joining of the clamps to the piping and actuator was created during construction and revised as the system was tested. Originally the pipes were secured on each end by only a longer threaded rod a piece, but during testing of the system it was discovered that this setup allowed the entire extension system to swivel back and forth. This problem was corrected by adding an additional “L” bracket to each side of each pipe.

The backing for the electronics was originally planned to be mounted directly to each clamp. This was quickly thrown out as the dimensions of the extension system increased and this method would no longer be feasible. The backing was ultimately attached to the larger brace pipes by U bolts, since there was no other solution that would have supplied enough strength and been easy to implement without interfering with the braces’ ability to telescope.

In the original design, the extension actuator was mounted to the outer "L" member of the clamp to shorten the overall height of the robot. This shorter height would potentially allow the robot to maneuver through more complicated branch arrangements by decreasing the minimum vertical distance between each branch. However, it was found during construction and testing that this solution did not allow for enough space to mount all of the electronics on the backboard when the robot was entirely compacted, and it was found that mounting the actuator to the inner members of the clamps was a better solution. This mounting method also opened up much needed space between the outer and inner members of the clamp for the gear boxes, which were actually mounted on the same threaded rods as the pipe supports.

Once the extension system was constructed and joined the two working clamps, rudimentary climbing was attempted by connecting a power supply to each actuator in turn. During this testing the robot was closely monitored to see if it extended as close to parallel to the tree as possible without tipping back away from the tree, which would eventually cause the robot to lose grip in a longer climb. Additionally, the clamping of the arms was watched for asymmetric gripping motion that could potentially generate a moment within the extension system. As was previously mentioned, the extension system twisted during the initial extension and prompted the addition of another "L" bracket attached to each end of each pipe. The same test after the extra support was added proved successful.

6.4 Vertical Actuator Controller

The vertical actuator controller was designed to be simple and straightforward. A slight increase in labor cost associated in creating a controller by hand was chosen over purchasing a prefabricated controller. This decision was made for budget conservation and simplicity, as the controller needed for this module was only ON/OFF. The controller went through a large number of revisions, all of which proved to be extremely useful later in the development of the other controllers. The end result was an elegant but rugged and simple controller that performed flawlessly.

6.4.1 Final Design

The vertical linear actuator control was decided upon based on the needs of the robot. Speed control of this linear actuator was not required, since the maximum speed of vertical lift was only 2” per second, switching currents was minimal, and the control design compensated for back electromagnetic force (EMF) from the motor. The robot weighed less than the rated force for the actuator (i.e. rated torque for the motor), therefore it held at full extension without harm to the actuator. Relays were used for simplicity and ease of digital control. The selected relays were OMIH-L with a 12VDC activation coil and were activated by open-collector buffers with pull-up resistors to 12VDC voltage regulators. This was due to the relatively large current demands for lower voltage activation (~100mA at 5VDC). Also, 5A fuses were added in-line with the motor to prevent permanent damage to system components.

As seen in Figure 16, the vertical linear actuator control consisted of a simple ON/OFF control with diodes to allow for back EMF to pull current and allow for the motor to ease off before the current was switched to improve system fluidness and longevity by allowing low switching currents.

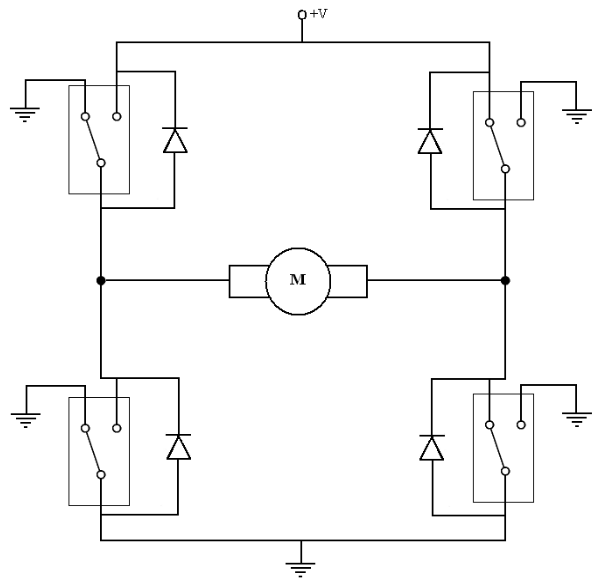


Figure 16. Vertical Linear Actuator Controller

Two specific relay positions worked in pairs and allowed for current to flow through the motor forward and backward.

Figure 17 shows the forward biased relay position of the vertical actuator, but this arrangement had potentially harmful positions which could ground the power supply (+V).

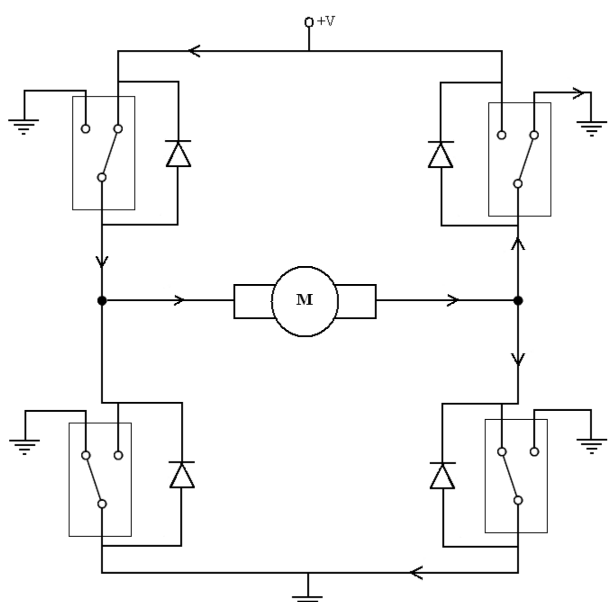


Figure 17. Forward Bias Relay Positions for Vertical Linear Actuator

There were however, two other acceptable positions as illustrated in Figure 18 andFigure 19, which illustrate the reverse bias and off position respectively.

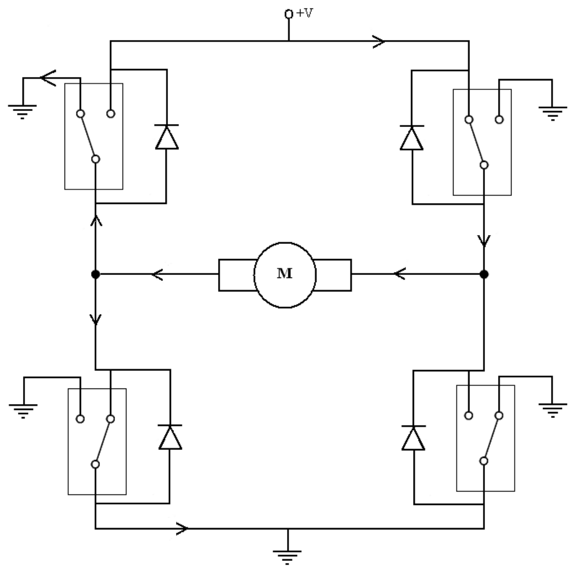


Figure 18. Reverse Bias Relay Positions for the Vertical Linear Actuator

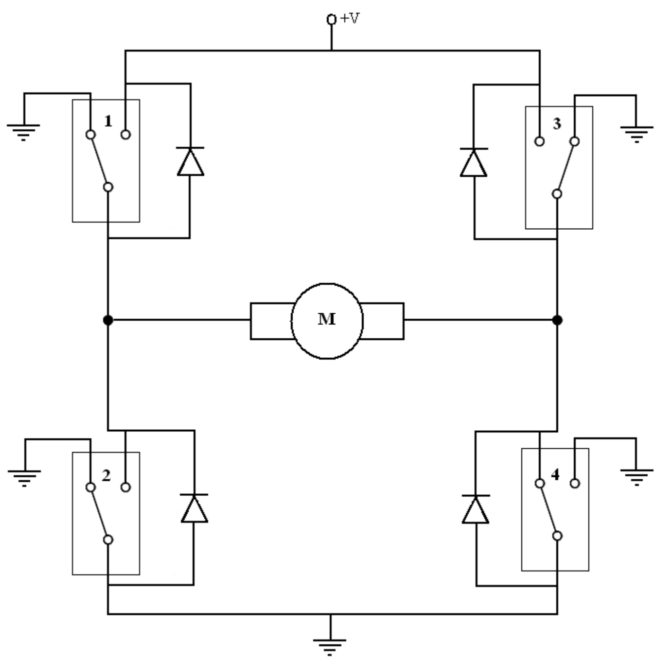


Figure 19. Off Relay Position for the Vertical Linear Actuator

To prevent destruction of the power supply by accidentally grounding it, a small logic circuit was implemented to interface with the system, only allowing for the outputs forward, backward, and off. This system had two inputs as there were three outputs requiring two bits of information. The outputs of the logic circuit to the relays (1, 2, 3, and 4) were decided by the inputs (A and B).

Applying digital logic design, it was found that these functions could be represented by two NAND chips. By utilizing NAND gates, OR and AND gates were constructed with two of the same integrated circuit (IC), instead of having to use two IC's of different types. This increased manufacturability by decreasing the types of IC's required for the design. These control logic outputs were the inputs to the added open-collector buffers as seen in Figure 20.

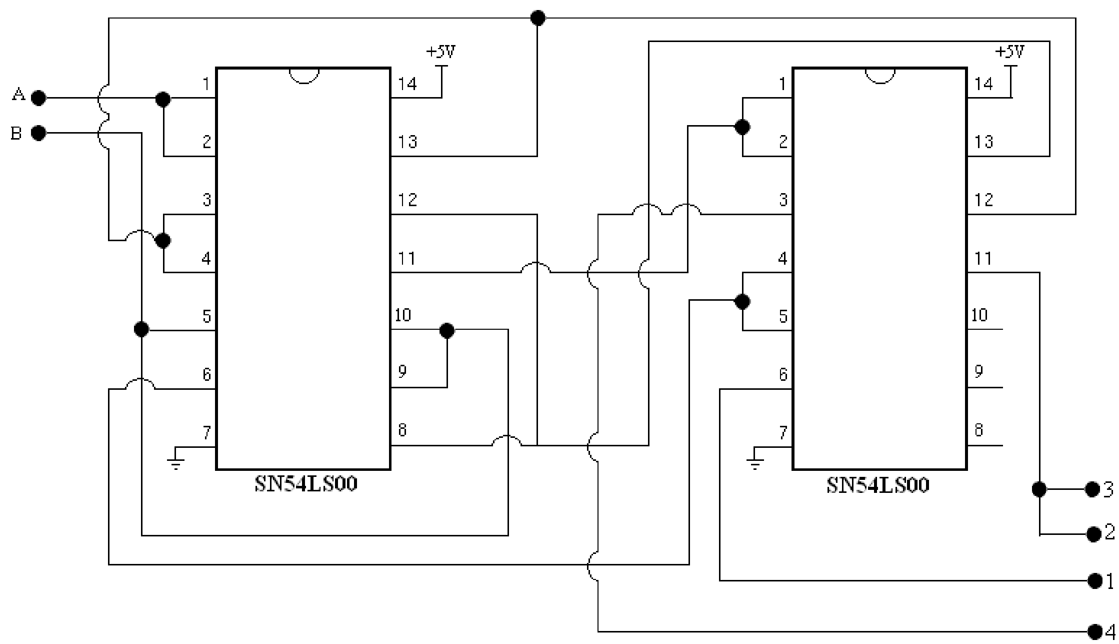


Figure 20. Simple Chip Logic Control for Vertical Linear Actuator

After many iterations, the final design of the controller added two IC buffers and a small fan (provided by a power supply) as shown in Appendix B.1. This made the controller slightly more complicated to construct, but still maintained its ease of programmability and use.

6.4.2 Module Testing

When this circuit was originally constructed, nothing worked; however, debugging led to several understandings. For starters, the inputs were out of specification, as the Diolan U2C provided insufficient current and nearly insufficient voltage (one input was ~2.8VDC when high, and the other was ~3.8VDC when high). When connected to a TTL chip and put to ground, the chip could not get below 1.5 VDC. The exact cause was never found, but the theory is that the low current cap on the I/O pins prevented the pin from being a true ground. After struggling with different solutions, the first attempt to correct this was a series of unitary gain operational amplifiers to bump the signal up to TTL logic levels in both current and voltage. This worked originally on a protoboard with ideal conditions, unfortunately when they were put under single polarity the low-level voltages were too high to turn off the TTL buffer they were attached to (low voltages were about 1VDC - 1.2VDC). This attempt at solving the plug and play problem only absorbed large amounts of time without providing any answers.

The next attempt at solving this problem luckily proved to be both a simple and favorable solution. While researching this problem, an inverting buffer was found that required only a micro amp or so to activate, and operated on TTL levels with a low-cutoff voltage of about 1.4 volts. Through double buffering the Diolan U2C inputs and outputting that to the logic, a layer of separation was created between the U2C and the rest of the circuit, and the plug and play problem was resolved. This addition to the design unfortunately made it more complex, but luckily made it even more robust.

Even with this problem solved, the circuit still did not function as planned. The current that was output from the TTL NAND gates was insufficient to power the OMI relays. After a quick search, another kind of relay was found that required about 1/4 the coil activation current and proved to be a more favorable option, as it was solid state. Unfortunately the TTL NAND gates were also insufficient to power these relays. When this was discovered, a non-inverting open collector buffer was used to quadruple the current sent to the coils and luckily drive the relays without a problem.

At this point, another problem was discovered with the melting of a 5VDC regulator. The current draw was much less than the rated capacity (around 300mA), but the constant draw caused the regulator to overheat. To solve this problem, a heat sink was added to a new regulator, which still heated up considerably in ambient air, but with a constant perturbation of the surrounding medium had no problem. To solve this, one of the power supplies (EFIKA or DC) was placed so that its fan blew over the heat sink. A half hour test was conducted to ensure there would be no problem later on.

Further testing was performed by first ensuring the correct inputs to the circuit were being applied. The two I/O pins of the U2C were attached to the available oscilloscope and Figure 21 and Figure 22 were recorded for the PC0 set to high and low respectively.

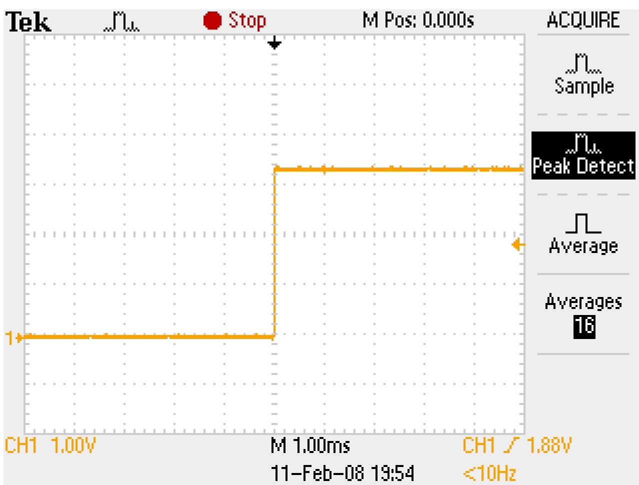


Figure 21. PC0 Set High on the U2C-12

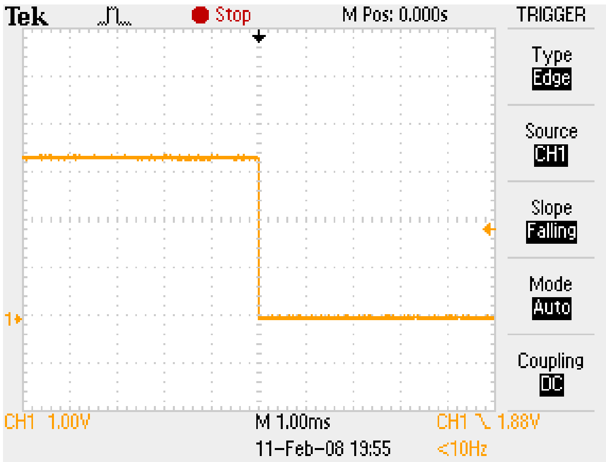


Figure 22. PC0 Set Low on the U2C-12

Knowing that the inputs were valid, the outputs of the two motor pins were tested when the inputs were set to the predetermined logical combinations 00, 01, 10, and 11. With both inputs low, the outputs should all be low, as shown in Figure 23.

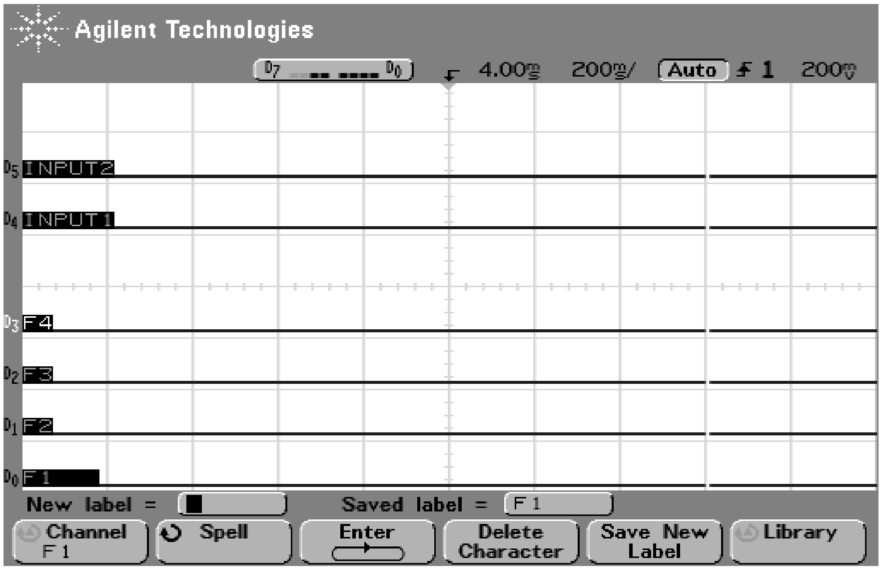


Figure 23. Logic Circuit Response to Two Low Inputs

With both inputs high, the outputs should all be low, as shown in Figure 24.

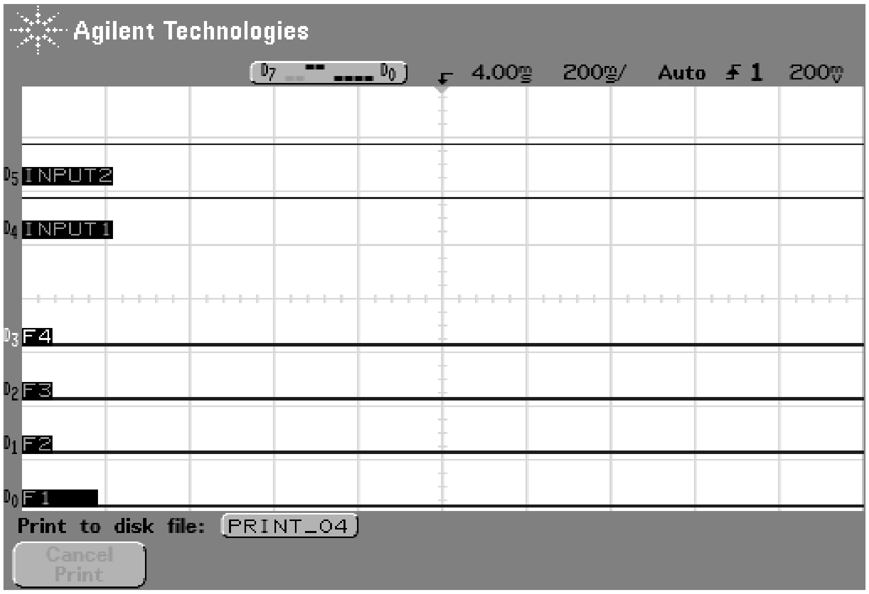


Figure 24. Logic Circuit Response to Two High Inputs

Now if input A is high (input 1) in the figures, and input B is low (input 2), then two of the function outputs should be high, shown in Figure 25.

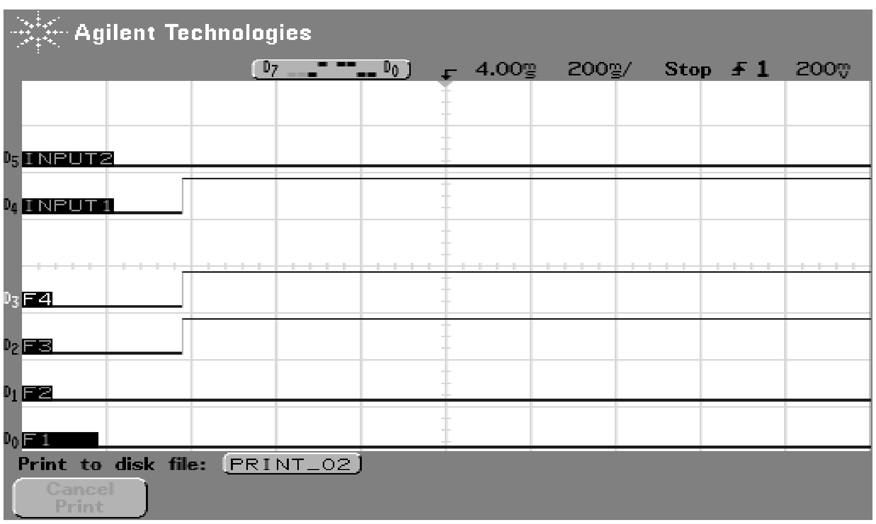


Figure 25. Logic Response to High Input 1 and Low Input 2

If the other input is put high, the other two function outputs, in this case F1 and F2, should go high, while F4 and F3 should drop low. This is shown in Figure 26.

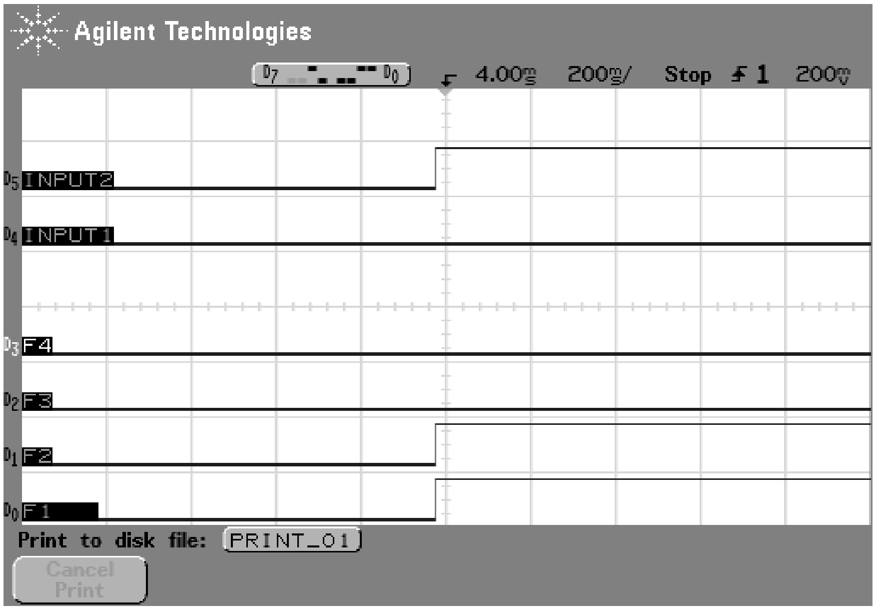


Figure 26. Logic Response to High Input 2 and Low Input 1

At this stage of testing the I/O was set on a timer, because the control system was open loop, and the ADC was tested independently and integrated later (to close the control loop). When this test was completed successfully, the motor was ready to be tested. The motor itself gave the correct response on the first trial, and pending the successful testing of the ADC, was ready to become a closed loop controller.

To test the ADC, a circuit was constructed based on the schematic seen in Figure 27.

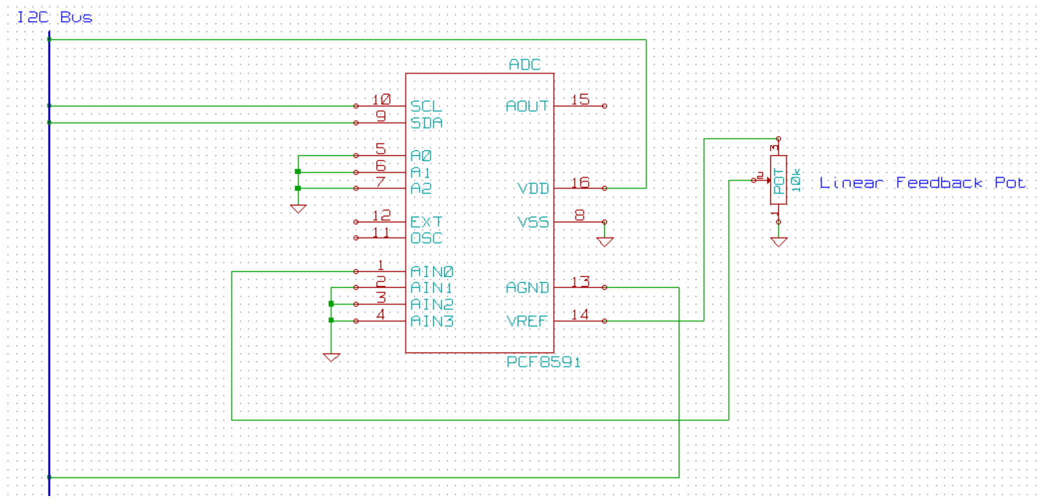


Figure 27. Analog to Digital Converter Schematic

In the circuit shown in **Error! Reference source not found.**, the potentiometer is used to adjust the voltage input to the ADC, while the recorded value is monitored on the computer screen.

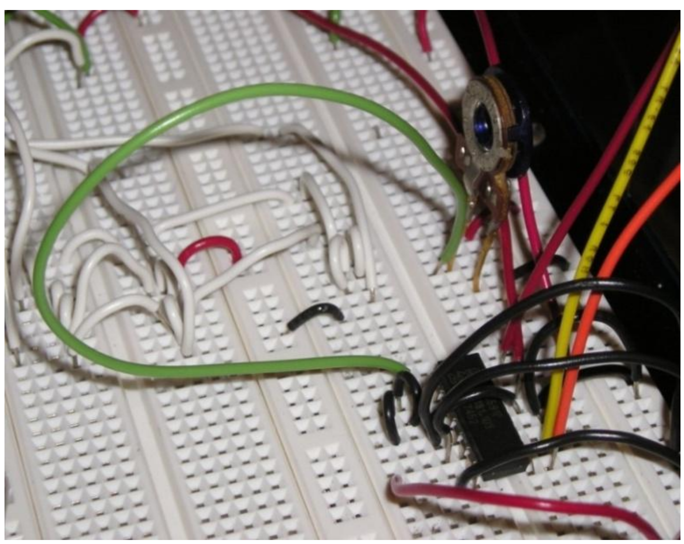


Figure 28. Potentiometer for Input Voltage

The values ranged nearly the full possible byte and corresponded linearly with the recorded voltage. With this testing done, the ADC was attached to the protoboard as shown in Figure 29.

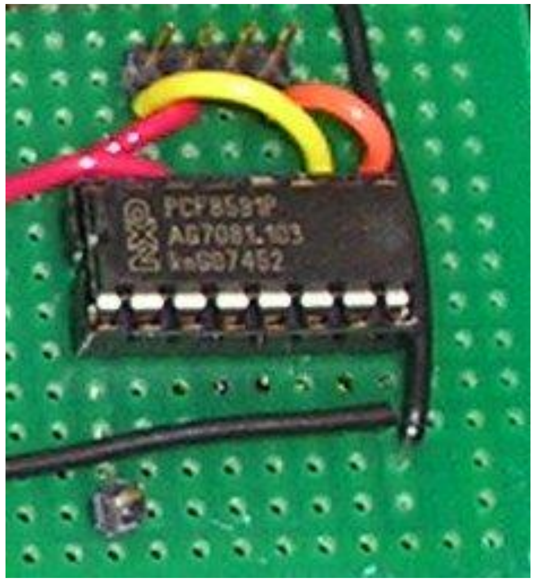


Figure 29. ADC Attached to Protoboard

The ADC was tested independently to ensure correct values before the two programs were integrated to make a smart motor controller. Once the ADC again proved to be valid, the program was integrated into a single C++ program. Empirical measurements were taken of the full open and closed voltages of the wiper on the actuator and limits were set to create simple functions that opened and closed the actuators. After successful testing using these limits, the motor controller proved to be robust, reliable, and easy to use, just as it was originally designed to be.

6.5 Clamp Actuator Controller

The drivers originally designed for the clamp actuator drivers were adequate in all ways but feedback. The resulting complexity of the previously designed multiple input analog to digital (ADC) converters made it a nearly impossible task to complete on a single printed circuit board (PCB). This was the majority of the changes applied to the design throughout the prototyping cycle. In controller terms, the resulting controller was accurate, robust, and simple, yet allowed for a large amount of flexibility in programming without being overly cumbersome.

6.5.1 Final Design

The two horizontal linear actuators, which controlled the clamping mechanism, needed a different control methodology other than ON/OFF control, as an ON/OFF controller would drive the actuators beyond the rated torque when they clamped onto the tree. All of the linear actuators came with a simple analog position feedback, which was used to judge the current position of each actuator utilizing a quad-ADC which communicates via I2C (AD7924). A constant torque controller was taken from the application notes of an 18200 H-bridge datasheet to derive a controller as seen in Figure 30.

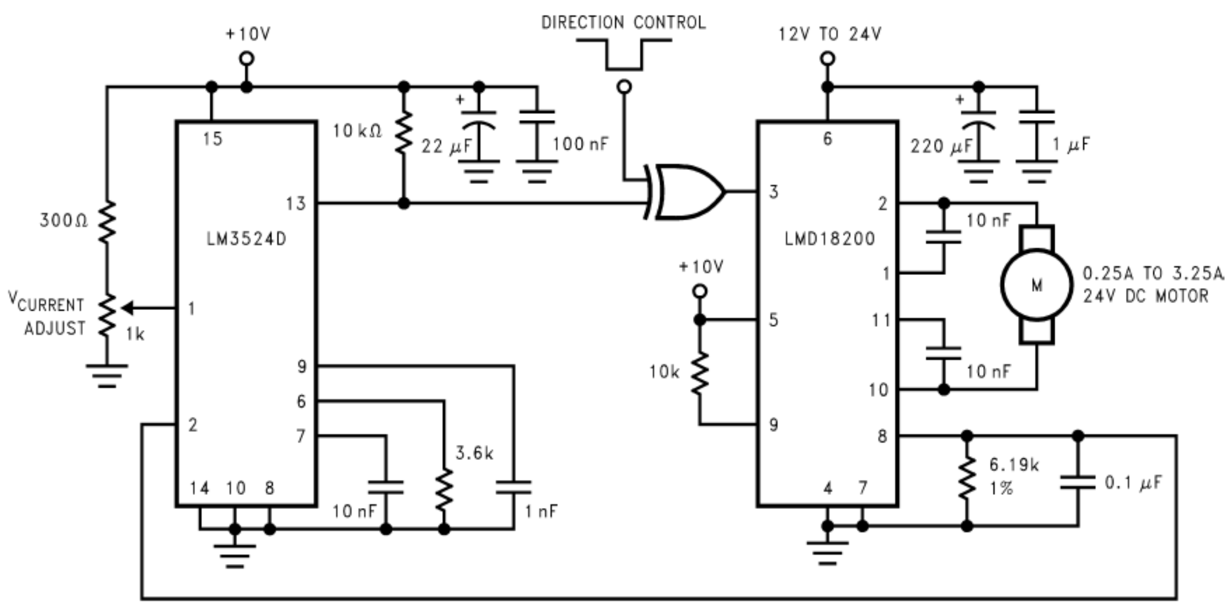


Figure 30. Constant Torque Controller Utilizing the LMD 18200 (LMD18200)

This controller used a voltage comparator in the LM3524D to compare the current sense output of the LMD18200. The current sense output was pin 8 on the LMD18200, which was input to pin 2 on the LM3524D and compared to pin 1. Pin 1 was connected to a variable resistor, which effectively set the desired torque. The variable resistor in the controller was replaced by a digitally controlled potentiometer, which could be controlled by a variety of methods including SPI, I2C, and parallel. This resulted in a controller that controlled torque based on the position feedback and action of the clamp with a relationship similar to that of the one in the application notes for the LMD18200 shown in Figure 31.

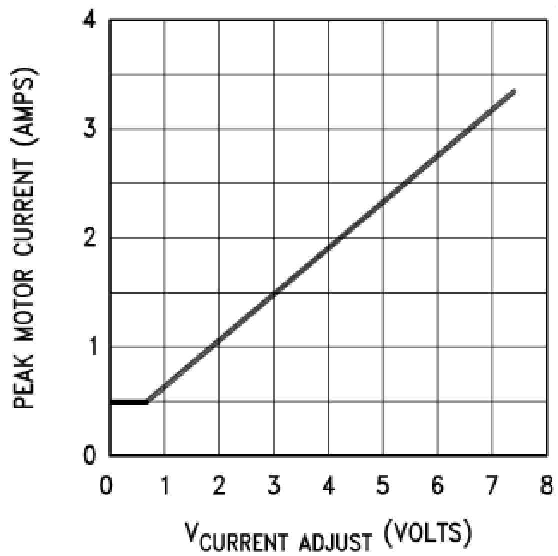


Figure 31. Expected Current Response for Horizontal Actuator Controller Output (LMD18200)

In using two distinct I2C slaves for each motor driver, programming became even simpler, as each motor driver called a specific ADC. Due to this, the modularity of the programming was even more effective, as the same writes to the command register were maintained and the same read cycle was repeated as each was connected to the first input. This allowed both driver functions of the clamps to vary only the pin number (which drove direction), and the two I2C slave addresses. This modularity allowed for easier debugging and quick replication for the complimentary clamp. As seen in Appendix B.2, the circuit was fairly complex, but was extremely robust and allowed for both easy and flexible programming. Although the control system was reliable and was only used for position in the program, it could have been used further to dynamically determine velocity and acceleration to further refine the control system. The final controller can be seen in Figure 32.

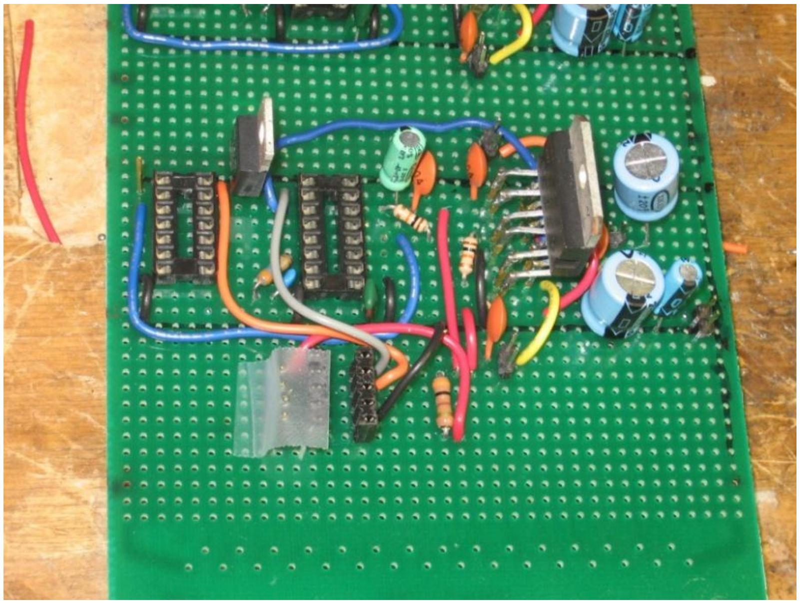


Figure 32. Final Soldered Clamp Motor Driver

6.5.2 Testing

Originally the design implemented was to use a single ADC for all of the actuators as it had four inputs, and a differential to the reference voltage applied to the actuator feedback pots could be used for slightly better accuracy. However, the datasheet revisions of the IC revealed that a large amount of external circuitry was needed to guarantee that the interference of each signal on the others would be minimal. An example from the PCF8591 datasheet in Figure 33 shows the extent of additional work required to use multiple inputs.

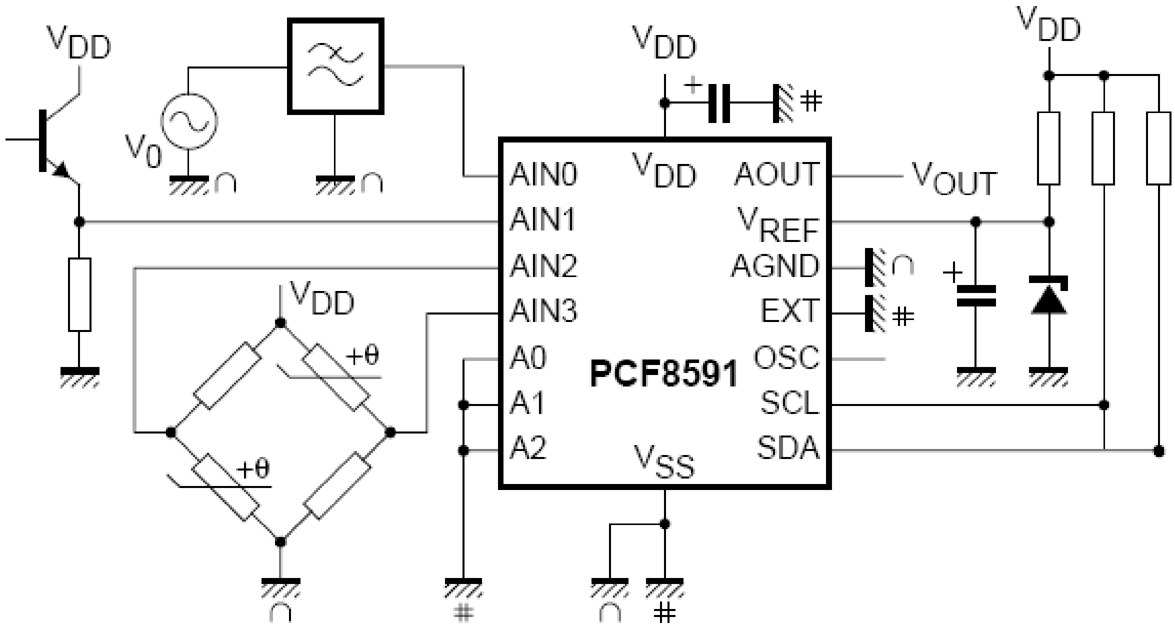


Figure 33. Multiple Inputs with the PCF8591F

This design would have required a long period of prototyping on a protoboard before being able to solder to a printed circuit board, and apply nearly the same additional cost that purchasing additional ADCs implied. Due to this, the team opted to have several ADCs instead of the large labor cost that would have been required to ensure signal accuracy. With this solution there was slightly less accuracy and more I2C slaves, however, the accuracy surpassed that which was needed and gave a much simpler, less labor-intensive, and more cost-effective solution.

Similarly to that of the vertical linear actuator, this driver also required signal conditioning for the input signal. This was solved in exactly the same fashion; a buffer was added to the I/O prior to entering the digital XOR that controlled the direction input to the 18200. The digital potentiometer also required a small amount of additional work in that it needed to be soldered onto a DIP style PCB before being added as it was MSSOP packaging.

During testing, a few problems were found: a mystery pin that was attached to nowhere, and a point which needed to be attached to +12VDC. After a few quick fixes, the motor driver performed flawlessly.

6.6 Rotational Motor Control

The two motors controlling the treads, which propelled the robot circumferentially around the tree, were constrained to strict uncertainties, as positioning on the tree was the most meticulous attribute of the overall system. Without this capability the robot would have been unable to descend the tree based on memory, and the uncertainty would have caused problems as the system's position would have drifted. Also, the torque required to spin the robot around the tree was provided by a traditional DC motor rather than an actuator; therefore, speed control needed to be implemented. For this application, a handmade controller would have been too complex and too intensive, and it would require a lengthy design period. Given these dilemmas, a more advanced manufactured controller was preferable to designing one from the ground up. The selected controller for the application was the MD23 by Devantech Ltd. Based on the simple I2C interface, the ability to control velocity curves, and the coupling with the EMG30 DC motor to produce an extremely accurate feedback system; the MD23 was the preferred choice and was used to control the circumferential rotation motion.

6.6.1 Testing

The motors worked as expected; the motors spun up and down accordingly. No real deviation was required or desired, so testing did not involve anything more than clockwise, counterclockwise, and stop. The extremely high gear ratio (100:1) on top of the (30:1) just made the output axle very slow. Since the mechanical portion of the rotation system did not work, further testing of the motor controllers was not possible.

6.7 *Branch Detection System*

The branch detection system utilized was a sonar array, as it allowed for maximum detectability, in a relatively small and low-cost package. The sonar array consisted of six I2C controlled sonar devices that allowed for binary segmental divisions of the tree trunk to be observed in two-dimensions. Incoming dimensions allowed for the possibility of branch tracking, and the critical regions near the robot could be looked at and logic applied in a simplified fashion.

6.7.1 Final Design

The sonar array offered a challenge of both conformity to the other systems and simplicity, as the array had to be large enough to measure around the circumference of a tree with a 13" diameter. Given the precedent of an I2C interface for the motor controller and the digital potentiometers, the preference for the communication of the sonar array was also I2C. The same company that produces the MD23 also produces a large selection of sonar devices for robotics applications. The SRF02 was chosen from the available sonar devices due to self tuning resonance, range, settable output (English/metric unit choices), and I2C interface. The advertised angle of sonar dispersion is 55°. Despite this information, an experiment provided dispersion closer to 45° for objects of thin cylindrical geometries close to that of small tree branches. The smallest object accurately sensed by the available SRF02 was less than 1/3" wide. The test also showed that the SRF02 could accurately sense these small cylindrical objects 20cm – 45cm away with the new measured half-angle. Based on this experiment, the number of needed sonar devices was approximated to 6 for full surface mapping.

The threshold was set at 20cm creating a blind region, which had to be tracked by the computer for object avoidance. Similarly, the maximum distance the sonar array could record was 45cm. The devices were setup so the threshold was the distance at which the ranges began to overlap. This created a variety of analytical solutions to the sonar responses, which fully satisfied the desired 2-dimension ranged limb tracking criterion.

The sonar array, as all aspects of the electronic design, was subject to top down design, and bottom up testing. Once the sonar devices were purchased and wires attached, each individual device was used to ensure both functionality and accuracy. Each sonar behaved as predicted, with a ~45 degree signal dispersion angle for large objects, and ~35 degree dispersion angle for small objects. Each sonar device was then individually named for distinction on the I2C bus, which is provided by the Diolan U2C-12, and tested on a digital logic analyzer for functionality (shown in Figure 34).

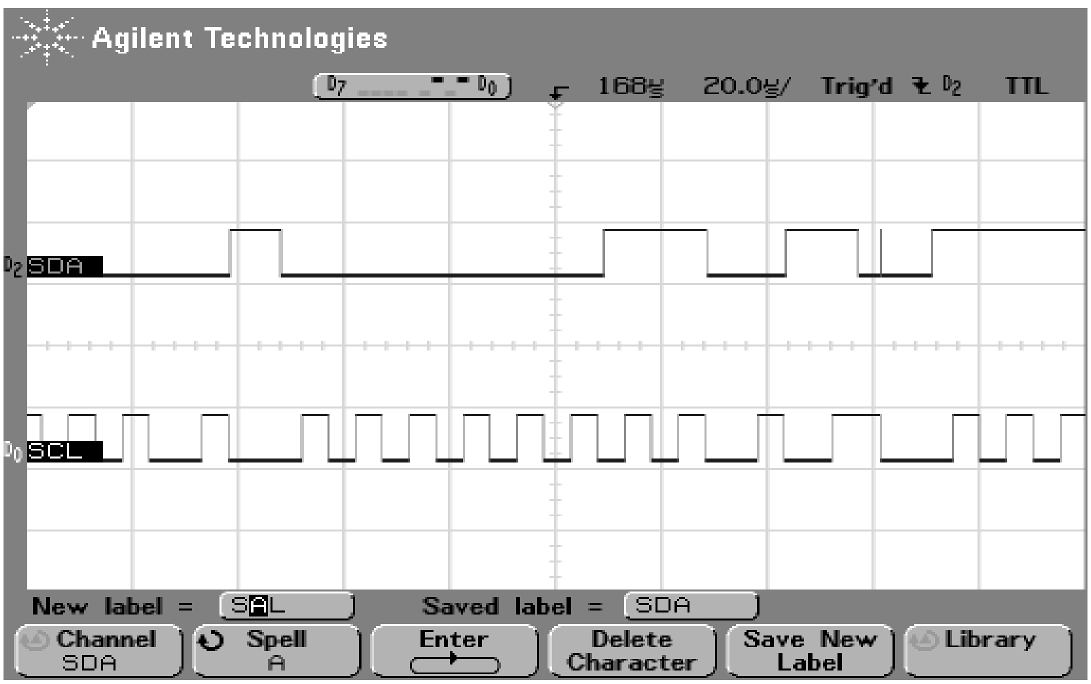


Figure 34. I2C Bus Initial Test On Digital Logic Analyzer

The labeled inputs to the DLA in Figure 34 are the SDA and SCL lines, which are the serial data, and the serial clock lines. The SCL goes low for a beat to signal the beginning of data transfer, which it does twice in the figure. With this in mind it is easy to see that one complete byte is transferred in between the low beats of the SCL line.

Knowing that the U2C's I2C bus worked, a physical bus to allow the attachment of all the slave devices to the master was constructed; this is shown in Figure 35.

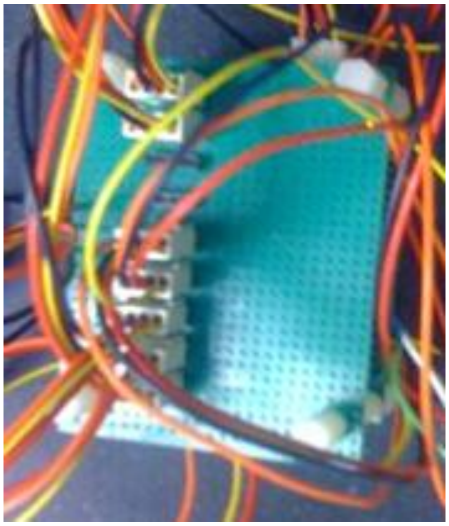


Figure 35. I2C Bus Expander to Accommodate I/O Needs

At this point, the challenge of programming the sonar array began; but first, a mock-up of correct proportions to the top of the robot was devised, shown in Figure 36.

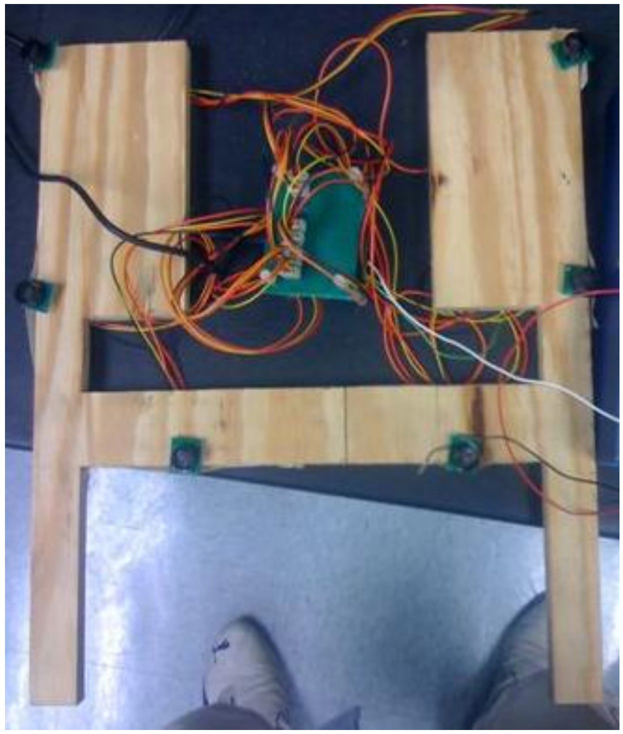


Figure 36. Wooden Mockup of the Top of the Robot

The wooden mockup allowed for the programming of the sonar devices to be done with accuracy, even without the physical presence of the robot. Initial programming involved simply taking the simple available program from the Diolan website and adapting it to ping all of the sonar devices. The order in which the sonar devices were pinged was very important, as incident readings from an adjacent sonar would cause bad data and, inevitably, a non-functioning portion of the robot. This was a problem encountered when first constructing the array, as more distant objects returned to the adjacent sonar as it pinged and caused incomprehensible data. This problem was avoided by both a minimum time delay and proper sequencing. Pairs of sonar opposite each other on the tree were pinged at each time, which allowed time for the signals to return before the neighboring sonars were pinged.

The sonar resolution of the tree trunk was originally a large concern, but simplicity was taken in these aspects. The circumference of the tree trunk was divided into six regions for each of the six sonar devices; this means that 60 degree sections of the tree were covered by each sonar.

The final design was namely in coding, as it had to be integrated with the overall logic of the system, however a schematic of it and the actual attached sonars can be seen in Figure 37.

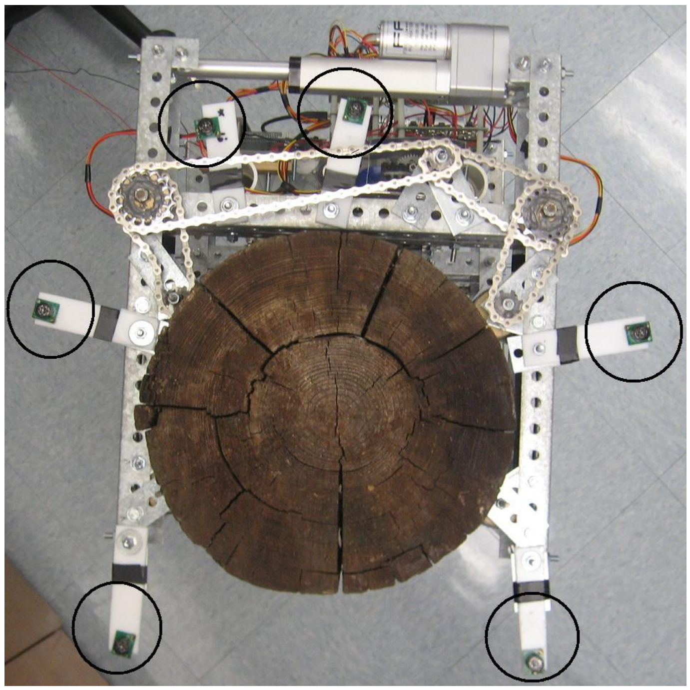


Figure 37. Attached Sonars.

This provided a relatively cheap, effective, accurate, and extremely simple branch to the robot.

6.7.2 Module Testing

The sonar program was tested by passing small cylindrical objects in front of the sonar at varying heights and positions around the mock up. The output in Appendix C shows the results of a test involving a 1” copper pipe that spiraled around the center of the sonar array to simulate how a branch might pass over while the robot was turning, while Appendix D contains the results of a test involving the same small copper pipe placed over a sonar at varying heights to show correct data categorization of the program. Each sonar device was put through these paces to guarantee that the programming was correct. The sonar device test output in Appendices C and D were the result of one continuous test, and are not the results of testing on different occasions. The output in Figure 38 shows no objects within 45cm of the array, and was the standard format for which the sonar program operated.

```
lumberjackL/~Desktop/Desktop/i2c_bridge-0.1.0/u2c/i2c# ./i2c

|Ranges    | Sonars 1 thru 6
13 - 21cm : 000000
21 - 30cm : 000000
30 - 38cm : 000000
38 - 45cm : 000000
```

Figure 38. Standard Sonar Output

Along with this output, the sonar data was stored in a file denoted as “map.txt” in the same directory and served as the principle navigation tool for the descent of the climb. The sonar output shown previously was stored within this file, and formed a binary two-dimensional map of the tree. An excerpt of this file is shown in Figure 39.

```
000000
000000
000001
000000
010000
000000
000000
000000
000000
```

Figure 39. Map.txt for Descent Navigation

To simplify the navigation, map.txt only retained the data from the 13-21 cm and 21-30 cm ranges masked over each other, (ex. 13 – 21 cm = 000100 and 21-30 cm = 000001, such that the mask = 000101). This was done so that the robot could turn right or left and extend its full body length without needing to read another line from map.txt. To complement this, a file containing orientation data was added to supplement the map.txt data to ensure correct maneuvering despite orientation changes made along the trip.

The sonar array could be elaborated on to produce better resolution for more complex problem solving in branch navigation, but the resolution used in the design was ample to provide a robot with the sensory required to navigate around one and two branch scenarios.

The revisions in the design were predictable and easy to attain. The tree and robot were slightly larger than originally expected, so the sonars were placed slightly further away from the tree trunk. This fortunately did not affect the programming, as the distance seen by the sonar was equivalent. The only things not seen be the sonar were bumps on the trunk, which caused the robot to no longer be level with the regions it was observing. Due to the large region under which the branches were divided, this was of no consequence to the programming itself.

One basic change was the change of the outgoing file; instead of columns, where each column is assigned to a sonar device, the file printed in-line one reading after the other. For instance, the first three rows from Figure 39 would appear as “000000000000000001.” This was so that the file buffer could keep track after each re-reading was taken while coming down the tree.

Due to the large demand on the I2C bus, which was originally constructed prior to the prototyping of a large number of the system components, it had to be expanded to meet I/O demands.

6.8 System Logic and User Interface

The user interface was a simple keyboard and LCD, which was attached to the outer-midsection of the robot and provided the user an easy access and simple way to communicate to the robot the vertical distance to climb. This went through a few revisions that changed which

LCD that was going to be used, as originally it was going to be a serial (RS232) device by Crystal Fontz. This was replaced by a much smaller and simpler non-RS232 LCD.

The system logic was a modular system which used predetermined functions from the other modules, with very few additional modules for itself. For instance, to climb vertically it used nearly plain English: open top clamp, extend vertical actuator, close top clamp, open bottom clamp, close vertical actuator, close bottom clamp. An excerpt of this code can be seen in Figure 40.

```
int climb_up()
{
        clamp_vert_open;
        vert_extend;
        clamp_top_close;
        clamp_btm_open;
        vert_close;
        clamp_btm_close;
        return 0;
}
```

Figure 40. System Logic Code

As can be seen, the bulk of the work was done by many, small prefabricated functions that were specific to each actuator, which was the beauty of the overall design. The flow charts for these functions can be found in Appendix E.

6.8.1 Final Design

Getting a good understanding of all of the system components was key for creation of the small unique functions that made up the overall majority of this system. For instance, the analog to digital converters that used the feedback from the actuators to complete the control loop required knowledge of the IC that took the reading, knowledge of the U2C, programming, and empirical data from the actuators themselves. By setting up the test circuit, it was found that the ADCs had a linear trend with an R-squared value of ~0.98, and that the result would return from the U2C in one byte hex format. By measuring the wiper voltage of the actuator, it was found that when it was closed it was at ~75mV, and when it was open fully it was at about ~4.25 mV. Given that the ADC would return a value from 0 to 256 (in decimal), a few simple if-else statements gave a function that would set the pins of the GPIO to close the actuator, and likewise with opening the actuator. The majority of the function to close the actuator is seen in Figure 41.

```
for(dummy =1; dummy !=0;)
{
vert_position = vert_ADC(myargc, myargv); /*reads the ADC
if(vert_position == 0 || vert_position == 1 ) /*0 is not closed or extended, 1 is extended
        vert_set_close();  /*set the GPIO to apply reverse polarity to actuator
if(vert_position == 2)
{       vert_set_stop();
        dummy = 0;}  /*set the GPIO to apply no polarity
}
```

Figure 41. Vertical Actuator Close Function

As is seen, the vertical actuator close function was extremely simple, and it was simple because it used other small simple functions to do the work. The vert_ADC function called in the code above actually read and wrote via the I2C bus and sent its findings to another function (convert_ADC), which actually categorized the position of the actuator before returning to vert_close. The ADC information was critical in the vert_ADC function, and the empirical evidence was critical in the convert_ADC function for correct categorization of the actuator position. With just a little programming knowledge it fell into place rather easily.

Functions like the one mentioned before were used throughout the code. Specifically, the clamp actuator drivers, which bore exactly the same driver as the vertical actuator, with the addition of setting a digital potentiometer and using only one I/O pin of the GPIO. Programming this way made it fast and simple, so that the tough problems, such as writing a driver for a motor, could be broken down into small, simple, and even lazy solutions.

6.8.2 Module Testing

After completing the logic design, this was applied to each actuator and motor controller in turn to verify that it functioned correctly. Once this was done, the system logic was further tested in the final testing.

7 Final Testing

After the individual components were modularly tested to ensure that they were working correctly, the robot was tested against the project's working criteria to evaluate the relative success of the project. Initial project testing was done on a 5' section of telephone pole to demonstrate the robot's ability to climb without slipping and to detect obstacles in its path. Ideally the diameter would have been constant, but it actually varied up to $\pm$ 1" from the irregular circle of the cross section. This initial test was repeated several times as a precaution to ensure that the clamping and extension systems worked as they were designed to and that the robot would not fall or slip from a greater height during the full scale testing. Obstacles were placed in the robot's path to ensure that the detection system was indeed functioning correctly.

Unfortunately, the failure of the rotation system prevented the robot from being fully tested as originally planned.

8 Conclusions and Recommendations

At the completion of the design process, the final design was analyzed to determine how well it met the original project goals and working criteria, and recommendations for further work were made.

8.1 Conclusions

Since the goal of the project was based on the working criteria initially described, the success of the project as a whole can be evaluated based on a relative comparison to the completion of each criterion. As is seen in Table 2, the design was given a percentage of each weight, with a maximum of the full percentage allotted for each criterion, to aid in the evaluation.

Table 2. Evaluation of Final Design Based on Achievement of Working Criteria

Criterion	Weight (%)	Score (%)
Maneuverability around branches	30	15
Fairly simple design	20	18
Climb fixed diameter trunk	15	15
Reversible climbing	15	15
Extra weight	15	10
Vertical speed	5	5
	Total	**78**

The only criterion that the robot fell short of was the maneuverability, which was also weighted the heaviest. This category can be thought of to be composed of three equally weighted sub-criterions: branch detection, rotating, and having a body that allows for more complex branch geometries. The final sonar array was in working order and capable of detecting branches adequately, so this was awarded the full 10%. The ability to rotate, on the other hand, was not in working order at the time of the project completion and was given a 0 of 10% rating. While the final design would have allowed the body to navigate through some branch arrangements if the rotational system had worked, the robot's maneuverability would have been increased if the unextended height was decreased and the arms were able to swing back farther. Ideally, the robot would have a limited vertical height, but this had to be extended during construction for more room to mount the electronics. The original design had the extension actuator mounted to the outer sides of both clamps, but did not provide enough room between the clamps. An ideal design would also have allowed the arms to swing back entirely behind the extension system. The robot's body was awarded half of the 10% for its vertical height and arm swing back for a total of 15 of 30% for the maneuverability criterion.

At the beginning of the design process, the group strongly emphasized the necessity to have a simple design for a higher chance of project success. This philosophy drove many construction revisions throughout the course of the project and led to an overall simple design. The only reason a full 20% weight was not awarded was because several components such as gears, gearboxes, and motor controllers had to be constructed. With additional funding to purchase these components prefabricated, the design could be slightly simplified.

Throughout the course of testing it was seen that the robot performed well in the areas of reversibility and the ability to climb a fixed diameter, so these categories were given the full 15% each. The robot can easily replicate its climbing actions to transverse up as well as down and it can map the branch positions necessary to climb back down without two sets of sonar. The telephone poles used to model the tree were not exactly a constant diameter or perfectly circular, which allowed for testing on slightly variable diameters between 12-13 inches. The robot worked well and had no problem 13" diameter sections for which it was designed, but began to slip on the undersized 12" diameter sections because the clamping actuators reached their maximum extension before the necessary force to tighten the treads was achieved. This problem was expected, but could be fixed with minor body changes, so it was not factored into the scoring.

The robot's ability to carry additional weight was satisfactory but limited by the maximum strength of the extension actuator. The 20-lb extension actuator was purchased instead of a stronger actuator, which increased speed at the sacrifice of strength, but the system was still able to accommodate an additional payload. When attached to the tree, each clamp was able to support 50 extra pounds without slipping. However, since each clamp weighed about 12 lbs, it was not possible to increase the payload to the goal of 15 lbs, since this would require the 20-lb extension actuator to support a total load of 27 lbs during extension and contraction. For this reason, the full score was not awarded. In hindsight though, 15 lbs would not be necessary to add a lightweight attachment, and the problem could easily be corrected by replacing the extension actuator with a stronger one. Although the overall speed of the robot was not stressed, the robot moved fairly quickly for the simplistic design, leading to a full score for this criterion.

This evaluation of the project based on the working criteria ultimately led to a final score of 78 out of 100%. The group considers this to be a successful score because with the exception of the failure of the rotational system, the robot accomplished all of the design objectives.

8.2 Recommendations

Based on the results and conclusions, the group has a several recommendations for further work on this project. First, there should be a more even distribution of concentrations working on the project. The tasks were pretty evenly split between mechanical and electrical/computer requirements, but the one electrically-inclined engineer ended up having a heavier workload, since the mechanical tasks could be split between four people. An extra electrical engineer would have allowed the workload to be distributed much more evenly.

As for the rotation system, the group concluded that the system of bike chains and gears was not the best possible solution, and further brainstorming, prototyping, and testing would definitely be required if further work were continued on the project. Access to a larger budget would obviously improve the quality of design, as this would allow for custom parts. It would also improve the simplicity of the design since prefabricated controllers could be used. Other revisions and variations on the final design, such as replacing the 20-lb extension actuator with a 100-lb actuator, would also help to improve the overall performance and robustness of the robot.

A EES Modeling Code

Tree Climber Mechanical Systems

Sensitivity Variables

$d_{a1} = 6$ [in] *Distance from shoulder joint to actuactor*

$F_{act} = 20$ [lbf] *Force of actuator*

$n_{g2} = 21$ *Number of teeth on powered gear*

$r_{roller} = 1.5$ [in] *Constant radius of roller*

System 1: Arms

$d_{shoulder} = 14$ *Distance between shoulder joints*

d_{a2}

Distance from shoulder joint to first roller

d_{a3}

Distance from shoulder joint to second roller

$d_{act} = 9.4$ [in] *Constant length of actuator collapsed*

$d_{act,v} = 4$ [in] *Stroke length*

$\frac{d_{a2} + d_{a3,comp}}{2} = \frac{dia_{tree}}{2}$ *Want rollers to bisect center of tree*

$dia_{tree} = 12$ [in] *Diameter of tree*

$d_{a3,comp} = d_{a2} + L_{comp} + 2 \cdot L_{comp,2}$ *Distance from shoulder to roller 2 during clamp*

$x_{a3} = d_{a3} - d_{a3,comp}$ *Distance roller 2 moves during clamp*

$L_{arm} = d_{a3} + r_{roller}$ *Min arm length to contain rollers*

$T_{act} = T_{a2} + T_{a3}$ *Torque of actuator*

$T_{act} = F_{normal} \cdot d_{a1}$

$T_{a2} = d_{a2} \cdot F_{a2}$ *Torque at 1st roller*

$T_{a3} = d_{a3} \cdot F_{a3}$ *Torque at 2nd roller*

$F_{climb} = 2 \cdot F_{a2} + 2 \cdot F_{a3}$ *Force required to stay on tree*

$F_{climb} = weight \cdot \frac{n}{\mu}$

$weight = 20$ [lbf]

$n = 1.25$ *Safety factor*

$\mu = 0.7$ Coefficient of friction between tree and tread

$T_{act} = t_{a2.5}$

$t_{a2.5} = \left[\frac{d_{a2} + d_{a3}}{2}\right] \cdot f_{a2.5}$ Torque applied for point load

$f_{applied} = 2 \cdot f_{a2.5}$ Actual force applied for given act/angle/dim

$t_{clamp} = \frac{d_{act,v}}{speed}$ Total clamping time

$speed = 2$ [in/s] Extension speed

$F_{normal} = \cos[\theta] \cdot F_{act}$

$\theta = 0$ [deg]

System 2: Gears

$n_{g1} = 1$ Number of teeth on driven gear

$T_{g1} = 0.1085$ [lbf*ft] Constant motor torque

$T_{g2} = \frac{n_{g2}}{n_{g1}} \cdot T_{g1} \cdot \left| 12 \cdot \frac{lbf*in}{lbf*ft} \right|$ Torque generated on powered gear

$rpm_{g1} = 170$ [rev/min] Constant motor rpm

$rpm_{g2} = \frac{n_{g1}}{n_{g2}} \cdot rpm_{g1}$ rpm generated on powered gear

$T_{fric} = \mu \cdot f_{applied} \cdot r_{roller}$ Rotational friction caused by clamping

Angle of Deflection from tree

$\sin[\theta_{compliment}] = \frac{d_{shoulder} - d_{act}}{2 \cdot d_{a1}}$

$\theta_{deflection} = 90$ [deg] $- \theta_{compliment}$ Maximum angle arms deflect from tree

Actuator Check

$half_{actuator} = 3 / 4 \cdot 3.25$ Maximum distance actuator can move toward tree

$act_{actual} = \sin[\theta_{compliment}] \cdot d_{a1} - half_{actuator}$ Actual distance actuator moves toward tree

Other Dimensions

$d_{gap} = 0.125$ [in] Gap between wheels and c-clamp side wall

$z = r_{roller} - 0.5 \cdot dia_{spring}$ The horizontal distance beyond tree diameter

$d_{shoulder2} = 2 \cdot r_{roller} - z + d_{gap} + dia_{tree}$ Actual shoulder pin distance

System 3: Treads

Assume no slip between roller and tread

$d_{roller} = d_{a3} - d_{a2}$ *Distance between rollers*

$c_{tread} = 21$ [in] *Constant circumference of tread*

$c_{tread} = 2 \cdot d_{roller} + r_{roller} \cdot 2 \cdot \pi$

$w_{tread} = 3$ [in] *Width of tread*

$A_{tread} = d_{roller} \cdot w_{tread}$ *Contact surface area*

$n_{revolutions} = 1$ [rev] *Constant revolutions covered by wheel over 1 circumference*

$$roller_{circum} = \frac{\pi \cdot r_{roller} \cdot 2}{n_{revolutions}}$$ *Constant circumference of roller*

$$rotatation_{speed} = roller_{circum} \cdot rpm_{g2} \cdot \left| 0.016666667 \cdot \frac{in/s}{in/min} \right|$$ *Rotational speed of entire robot*

$F_{tread,spring} = k_{tread} \cdot x_{tread}$ *Force of tread spring*

$F_{tread,spring} = F_{a3}$ *Max force, want less than this so will compress*

k_{tread}

Tread spring constant

$x_{tread} = d_{roller} - [L_{comp} + 2 \cdot L_{comp,2}]$ *Compression of tread spring*

$2 \cdot r_{roller} = dia_{spring} + 2 \cdot d_{comp}$

$dia_{spring} = 0.5$ [in] *Diameter of spring*

$$L_{arc} = \pi \cdot dia_{tree} \cdot \frac{\theta_{comp}}{360 \text{ [deg]}}$$ *Parabolic contact length*

$c_{tread} = L_{arc} + L_{comp} + 2 \cdot L_{comp,2} + 2 \cdot L_{wraparound}$

$$\sin[0.5 \cdot \theta_{comp}] = \frac{0.5 \cdot L_{comp}}{\frac{dia_{tree}}{2}}$$ *Angle of arc when compressed*

$$L_{wraparound} = 2 \cdot \pi \cdot r_{roller} - 2 \cdot \pi \cdot r_{roller} \cdot \left[\frac{180 \text{ [deg]} - \theta_{force}}{360 \text{ [deg]}} \right]$$ *Length of tread on roller when clamped*

$$\cos[\theta_{force}] = \frac{0.5 \cdot L_{comp}}{\frac{dia_{tree}}{2}}$$ *Angle of applied force*

$$\cos[\theta_{force}] = \frac{L_{comp,2}}{r_{roller}}$$

EES Code Solution/ Final Dimensions

SOLUTION

Unit Settings: [kJ]/[C]/[kPa]/[kg]/[degrees]

act_{actual} = -0.1375 [in]
c_{tread} = 21 [in]
dia_{tree} = 12 [in]
◎ d_{a2} = 2.682 [in]
$d_{a3,comp}$ = 9.318 [in]
$d_{act,v}$ = 4 [in]
d_{gap} = 0.125 [in]
$d_{shoulder}$ = 14 [in]
F_{a2} = 8.057 [lbf]
F_{a3} = 9.8 [lbf]
◎ $f_{applied}$ = 37.73 [lbf]
F_{normal} = 20 [lbf]
$half_{actuator}$ = 2.438 [in]
L_{arc} = 5.926 [in]
L_{comp} = 5.688 [in]
$L_{wraparound}$ = 4.219 [in]
n = 1.25 [-]
⇨ n_{g2} = 25 [-]
$roller_{circum}$ = 6.283 [in/rev]
rpm_{g1} = 170 [rev/min]
⇨ r_{roller} = 1 [in]
θ = 0 [deg]
$\theta_{compliment}$ = 22.54 [deg]
θ_{force} = 61.7 [deg]
$t_{a2.5}$ = 120 [lbf*in]
T_{act} = 120 [lbf*in]
◎ T_{fric} = 26.41 [lbf*in]
◎ T_{g2} = 32.55 [lbf*in]
w_{tread} = 3 [in]
x_{tread} = 0.7219 [in]

A_{tread} = 22.08 [in^2]
dia_{spring} = 0.5 [in]
⇨ d_{a1} = 6 [in]
◎ d_{a3} = 10.04 [in]
d_{act} = 9.4 [in]
d_{comp} = 0.75 [in]
d_{roller} = 7.358 [in]
◎ $d_{shoulder2}$ = 13.38 [in]
$f_{a2.5}$ = 18.87 [lbf]
⇨ F_{act} = 20 [lbf]
◎ F_{climb} = 35.71 [lbf]
$F_{tread,spring}$ = 9.8 [lbf]
k_{tread} = 13.58 [lbf/in]
◎ L_{arm} = 11.04 [in]
$L_{comp,2}$ = 0.474 [in]
μ = 0.7 [-]
n_{g1} = 1 [-]
$n_{revolutions}$ = 1 [rev]
$rotatation_{speed}$ = 0.7121 [in/s]
rpm_{g2} = 6.8 [rev/min]
speed = 2 [in/s]
θ_{comp} = 56.59 [deg]
$\theta_{deflection}$ = 67.46 [deg]
T_{a2} = 21.61 [lbf*in]
T_{a3} = 98.39 [lbf*in]
t_{clamp} = 2 [s]
T_{g1} = 0.1085 [lbf*ft]
⇨ weight = 20 [lbf]
◎ x_{a3} = 0.7219 [in]
z = 0.75 [in]

No unit problems were detected.

⇨ denotes important input
◎ denotes important output

<u>Criteria for success:</u>

Clamping System

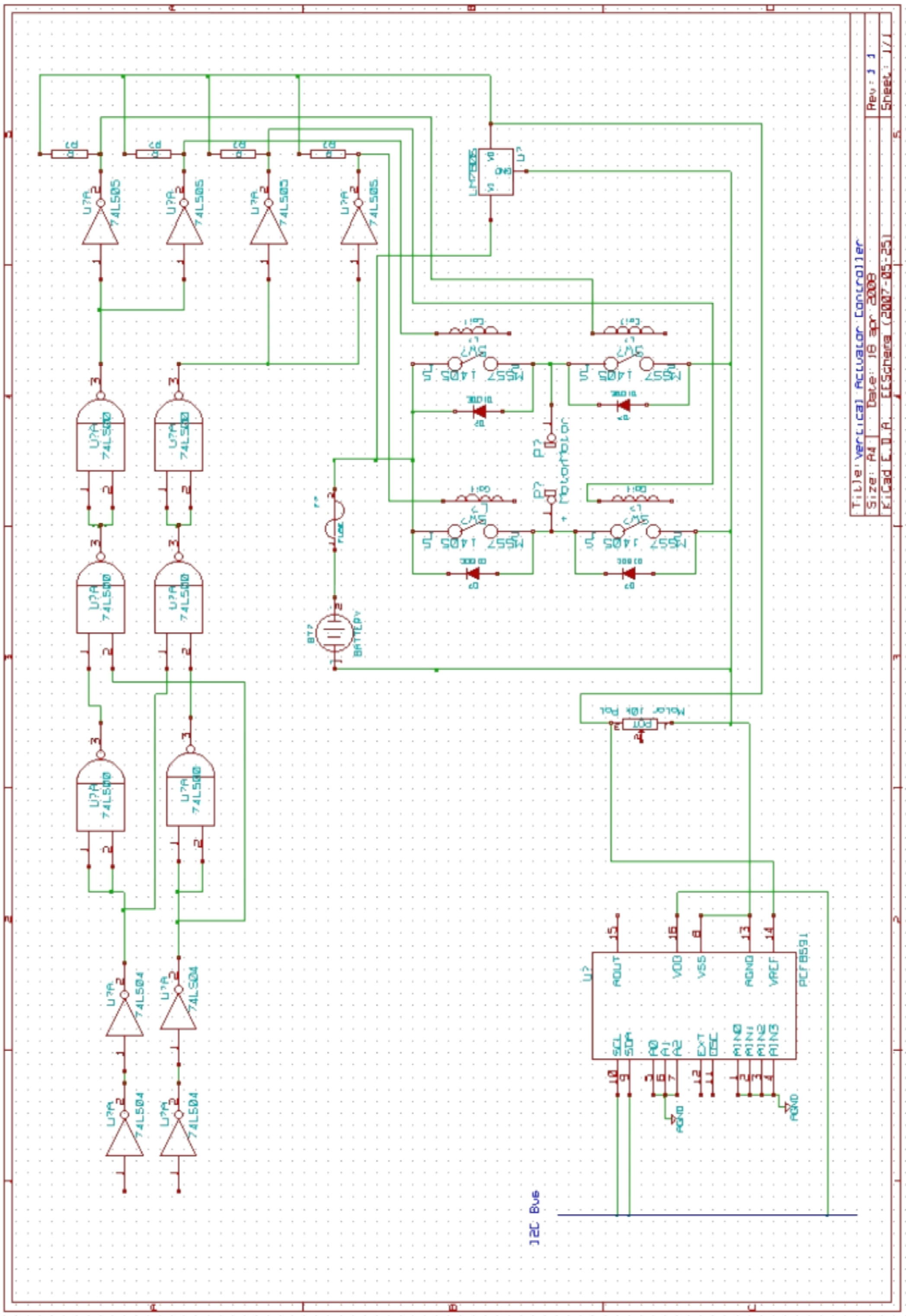
if $F_{applied} > F_{climb}$

Rotation System

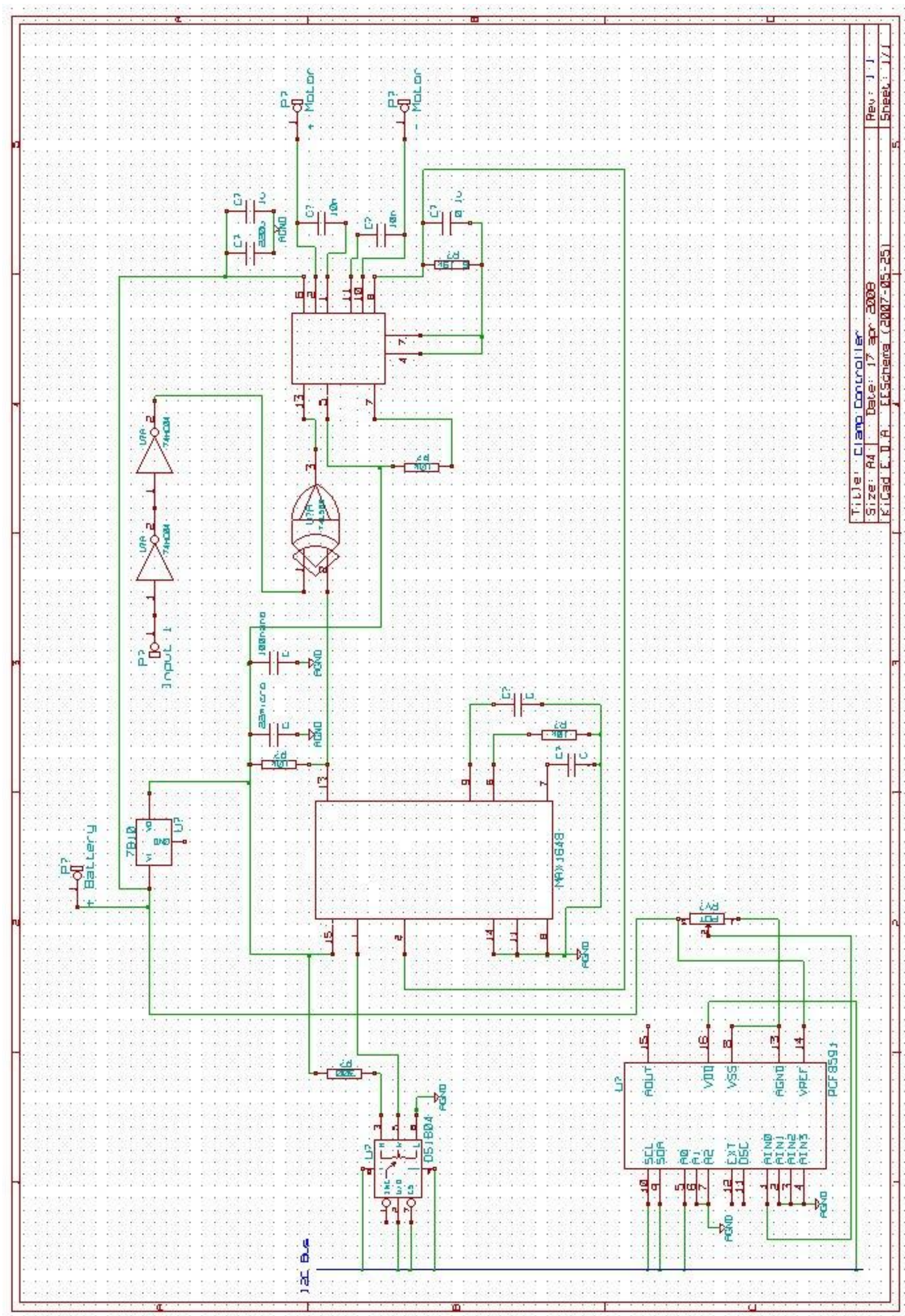
if $T_{g2} > T_{fric}$

B Electrical/Electronic Schematics

B.1 Vertical Linear Actuator Controller Schematic



B. 2 Horizontal Linear Actuator Controller Schematic



C Sonar Program Output

lumberjackL/~Desktop/Desktop/i2c_bridge-0.1.0/u2c/i2c# ./i2c

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 000000

30 - 38cm : 000000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 000001

30 - 38cm : 000000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 000010

30 - 38cm : 000000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 000100

30 - 38cm : 000000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 001000

30 - 38cm : 000000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 010000

30 - 38cm : 000000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 100000

30 - 38cm : 000000

38 - 45cm : 000000

D Sonar Program Output 2

|Ranges | Sonars 1 thru 6

13 - 21cm : 100000

21 - 30cm : 000000

30 - 38cm : 000000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 100000

30 - 38cm : 000000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 000000

30 - 38cm : 100000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 000000

30 - 38cm : 000000

38 - 45cm : 100000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 000000

30 - 38cm : 100000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 000000

21 - 30cm : 100000

30 - 38cm : 000000

38 - 45cm : 000000

|Ranges | Sonars 1 thru 6

13 - 21cm : 100000

21 - 30cm : 000000

30 - 38cm : 000000

38 - 45cm : 000000

E Flow Charts

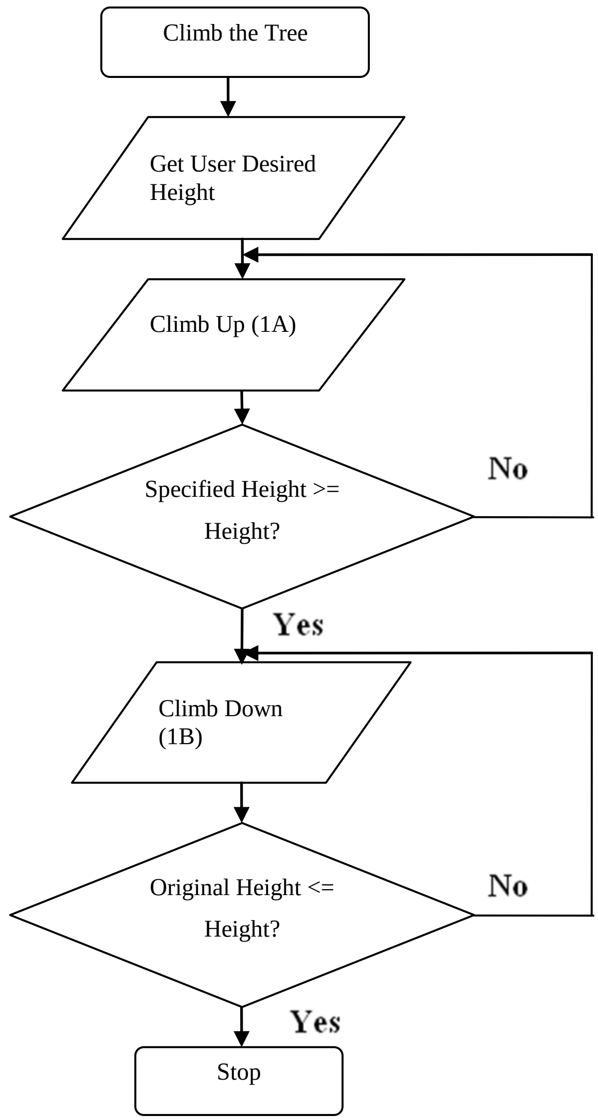


Figure E-1. Main Flow

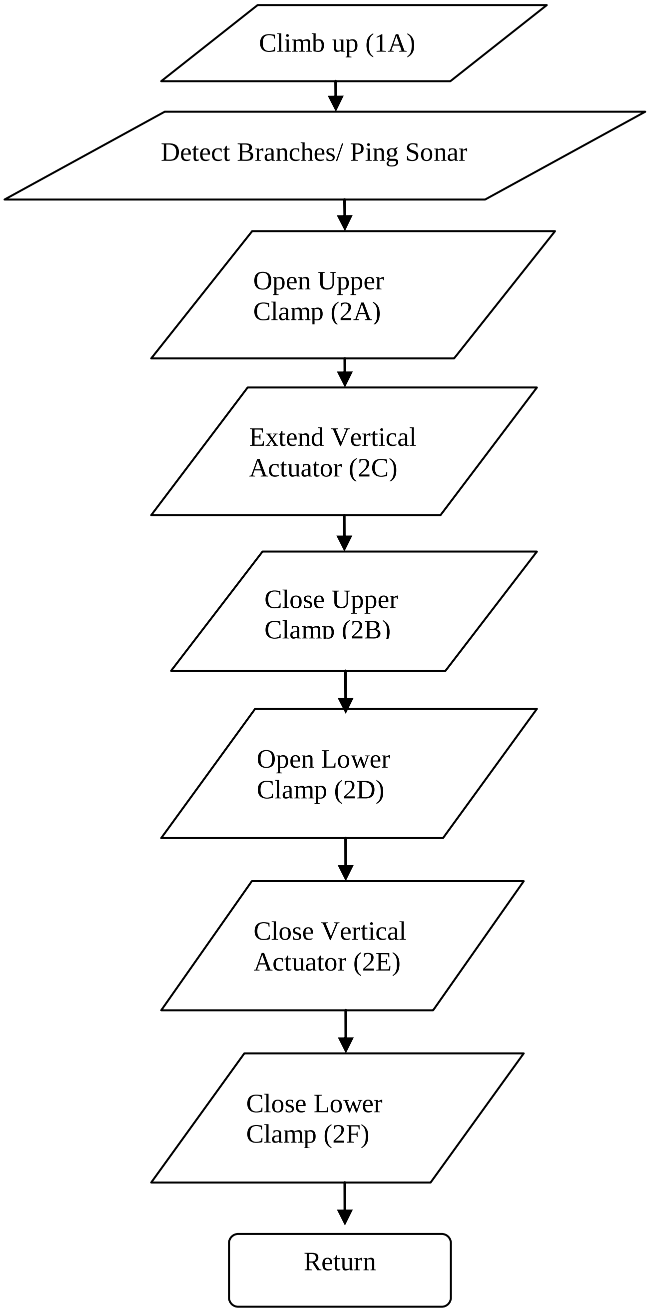


Figure E-2. Single Extension Climb Up Flow

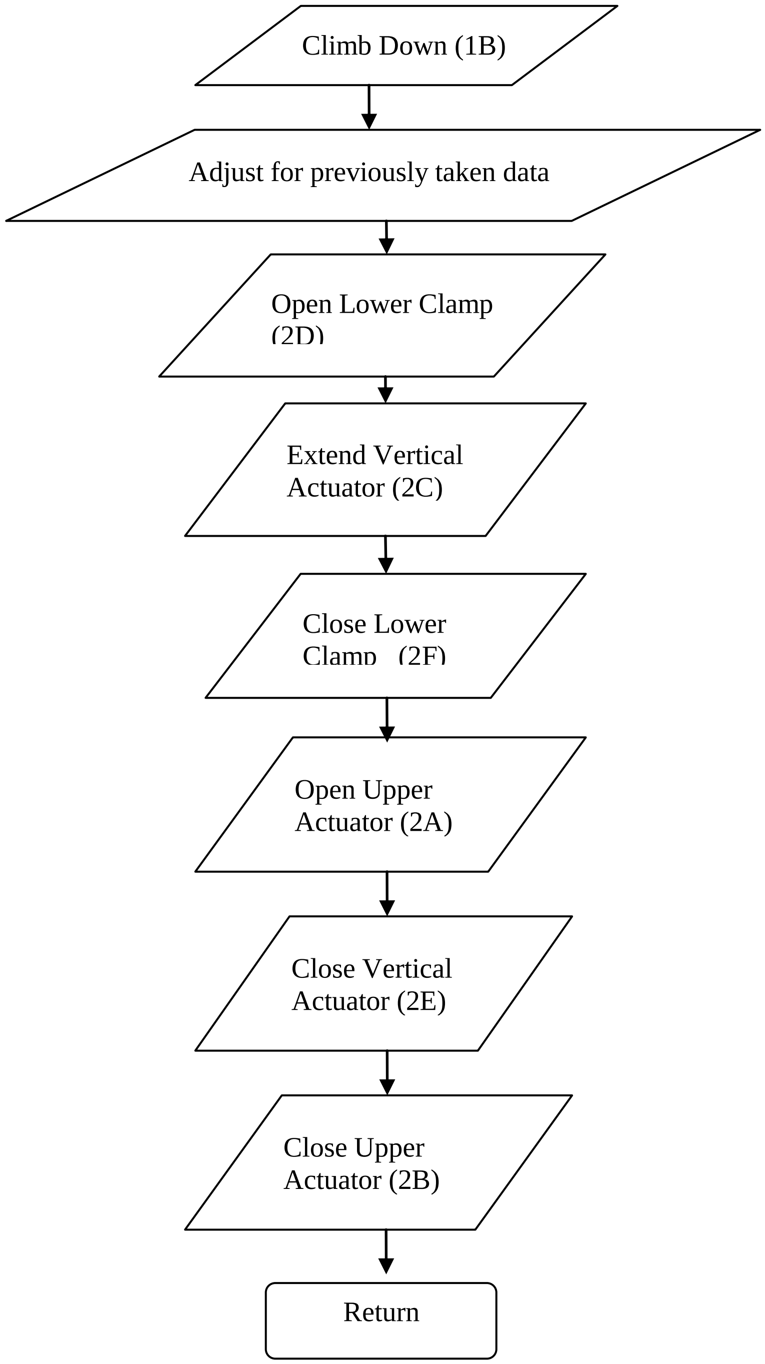


Figure E-3. Single Extension Climb Down Flow

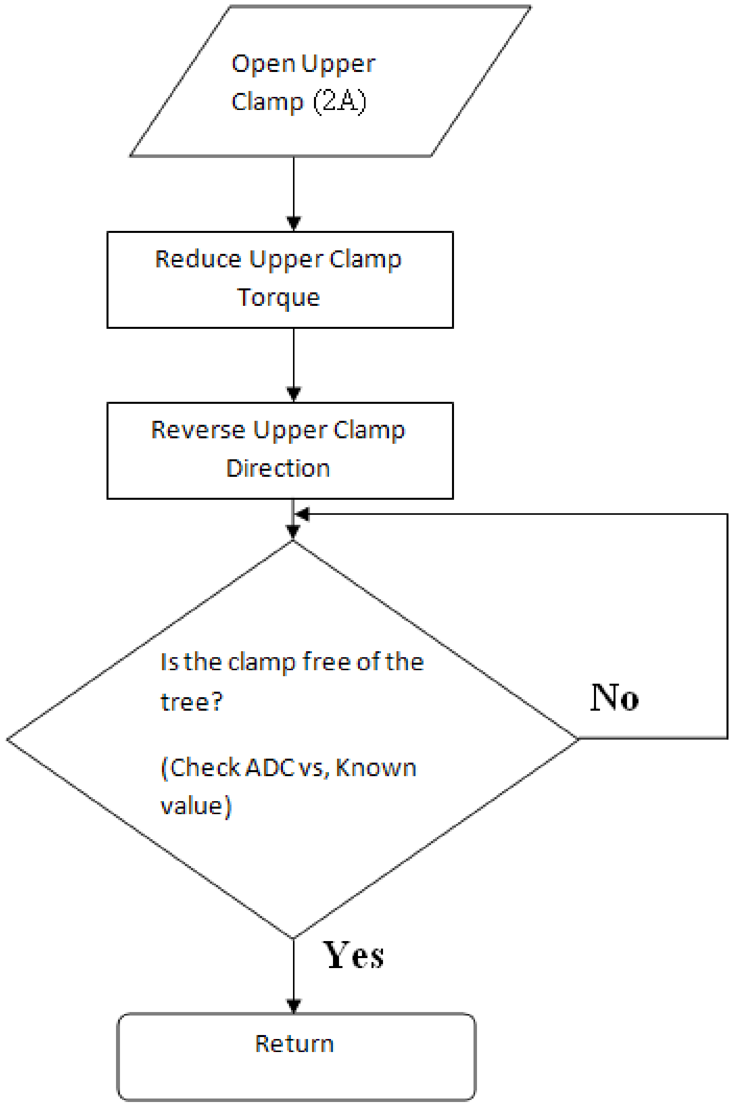


Figure E-4. Upper Clamp Open Driver

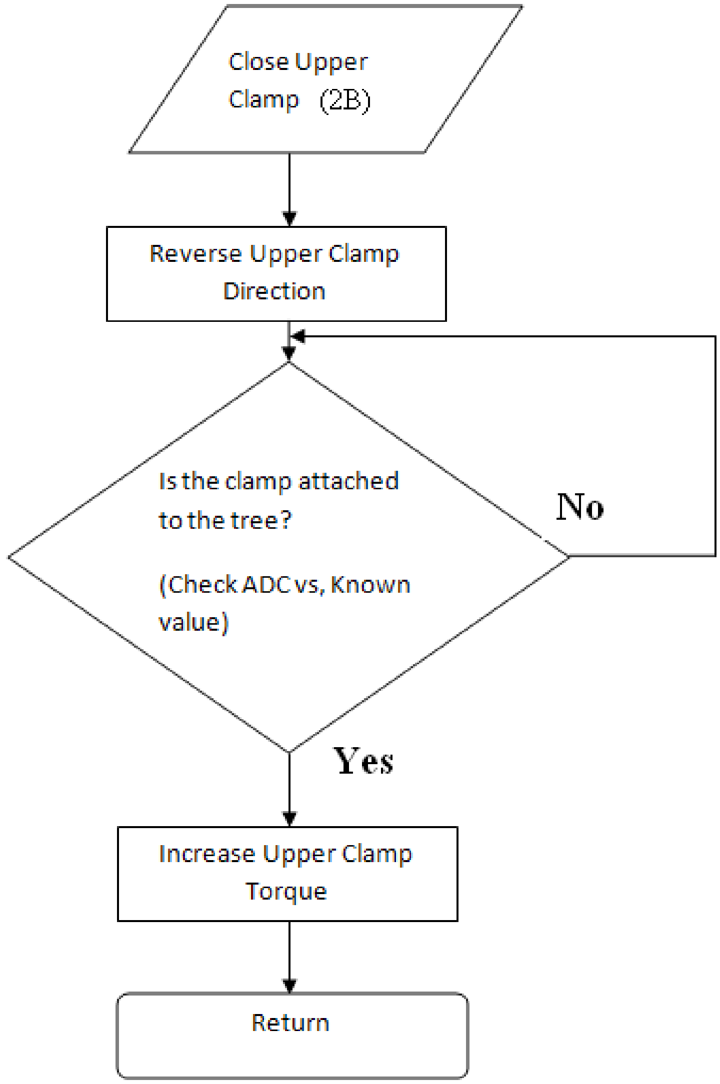


Figure E-5. Upper Clamp Close Driver

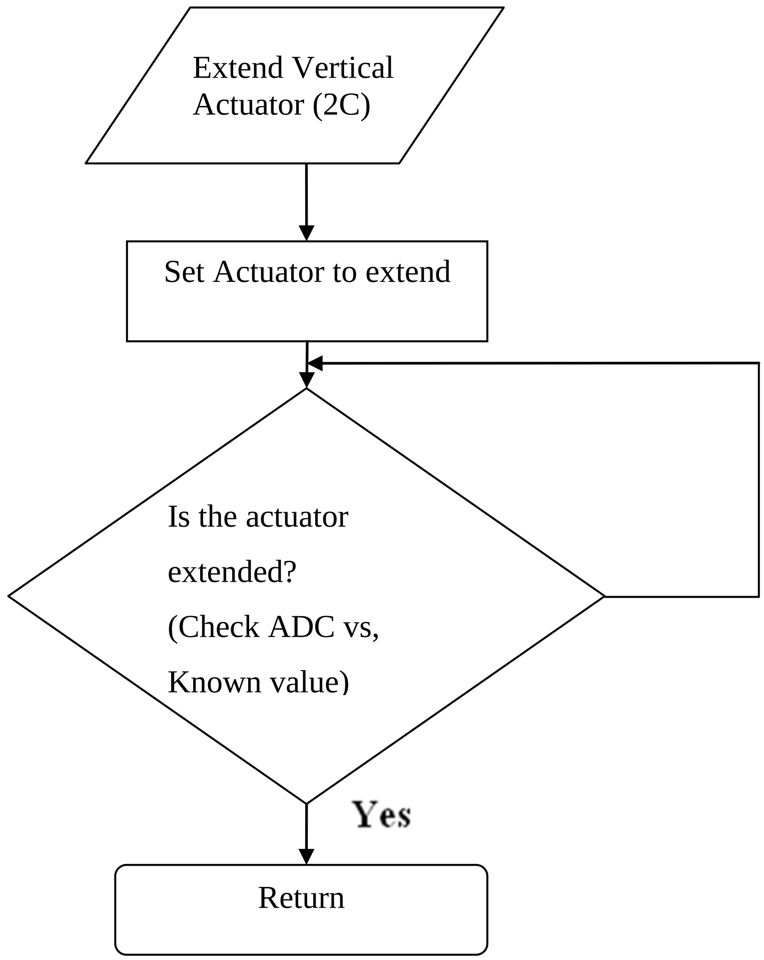


Figure E-6. Vertical Actuator Extension Driver

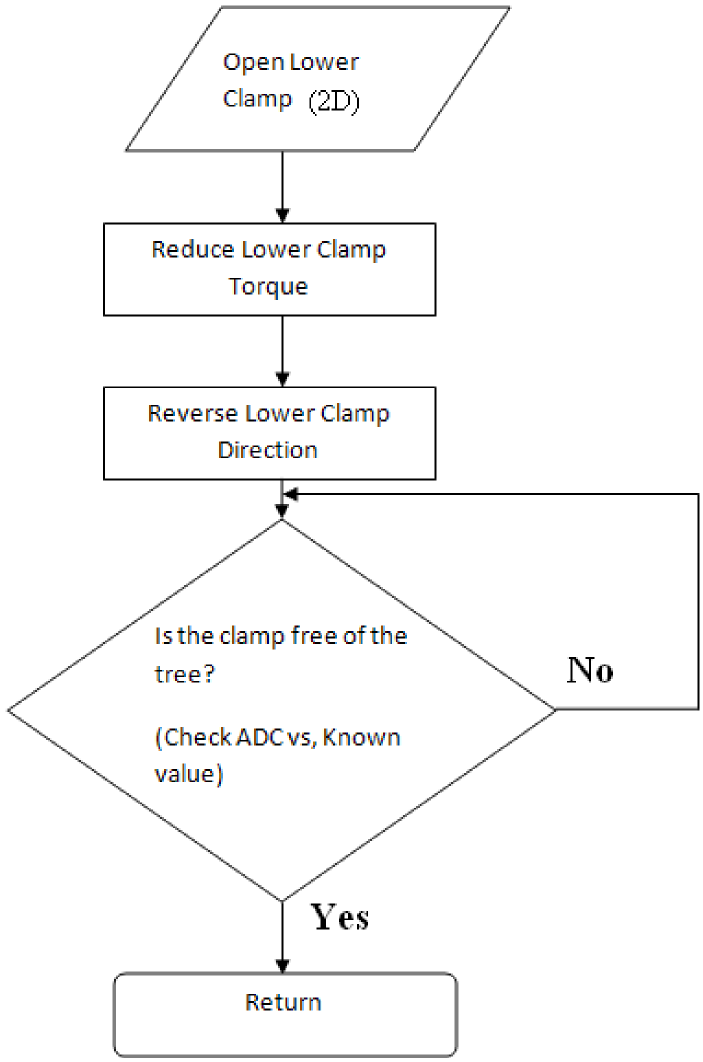


Figure E-7. Lower Clamp Open Driver

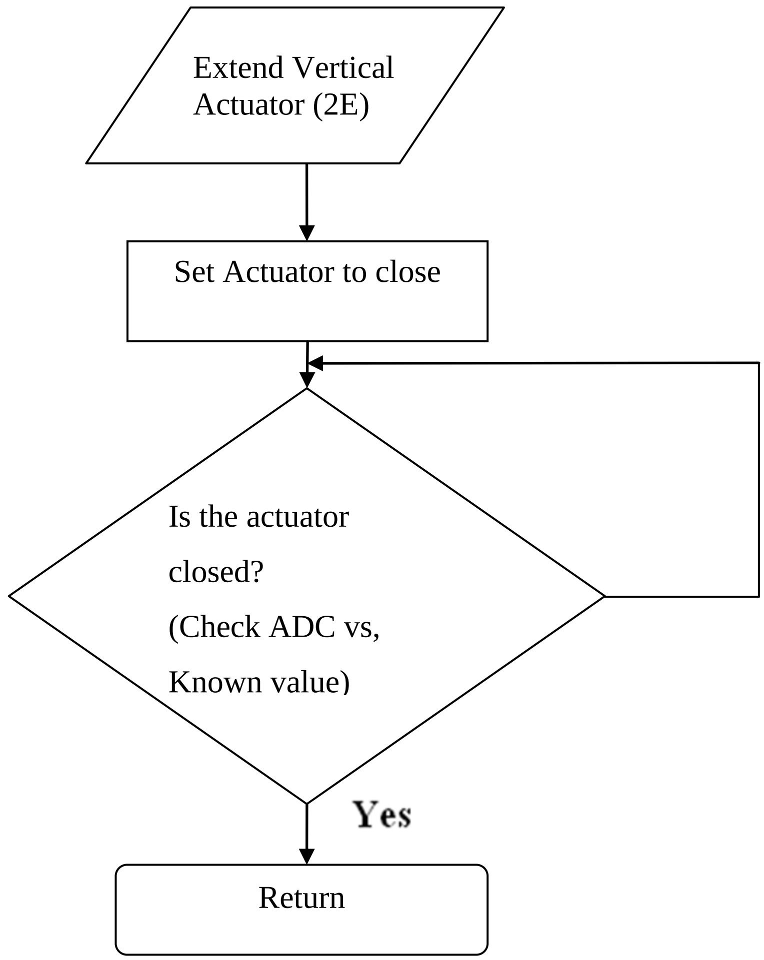


Figure E-8. Vertical Actuator Close Driver

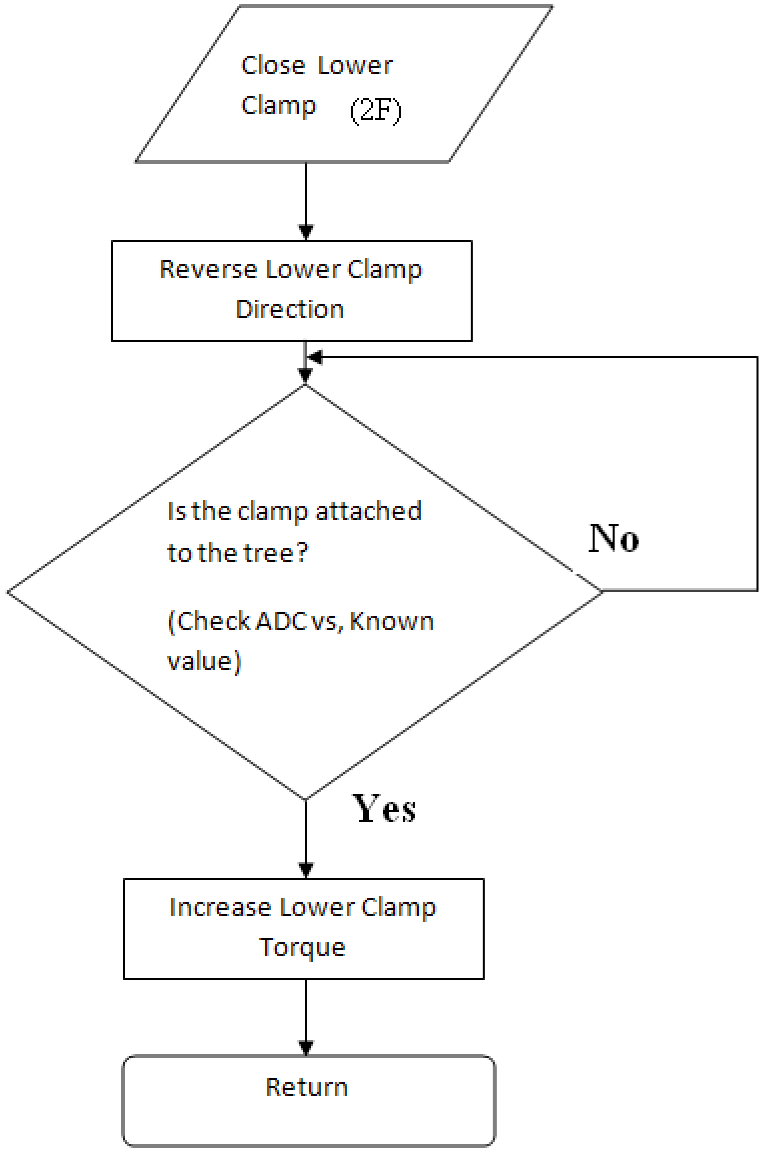


Figure E-9. Lower Clamp Close Driver

F Mechanical Construction

Bill of Materials:

Support and Body

Part	Quantity
Extension Actuator: Firgelli Automation [FA-PO-100-12-12"]	1
Clamping Actuator: Firgelli Automation [FA-PO-100-12-4"]	2
PVC Pipe • 12 1/8" Long • 2" outer Diameter • 3/16" Thick	2
PVC Pipe • 12" Long • 1 5/8" Outer Diameter • 3/16" Thick	2
Angle Bracket w/ 3/8" Holes • Rounded ends • 15 ¾" Long • 1 ¼" thickness on both sides of bracket	4
Angle Bracket w/ 3/8" Holes • 1" Long • 1 ¼" thickness on both sides of bracket	12
Angle Bracket w/ 3/8" Holes • 3" Long • 1 ¼" thickness on both sides of bracket	6
Threaded Bar for Hex Nuts • 5" Long • ¼"-20 thread diameter	6
Metal Brad Screws	16
Gear Mounting Sheet Plate • 4 ½" x 5 ¼ " • Scrap Metal Sheet	2
Gearbox Cover Plate	2

• Scrap Metal • 9” Long • 3” Wide	
Hex Bolts • 1” Long • ¼”-20 thread	18
Hex Nuts • ¼”-20 thread	38
Washers • ¼” Inner Diameter	30

Arms

Part	**Quantity**
Flat Metal Strip with 3/8” Holes • 2 ¼” Long • 1 3/8” Wide	24
Flat Metal Strip with 3/8” Holes • 3” Long • 1 3/8” Wide	20
Threaded Bar • ¼”-20 thread • 6 ¾” long	8
Threaded Bar • ¼”-20 thread • 5” long	4
Threaded Bar • 7” Long • 3/8”-16 thread	4
Wood Wheels • Scrap wood • ¾” thick • 2” diameter	24
36 Grit Sandpaper	4
Angle Bracket w/ 3/8” Holes • 20” long • 1 ¼” thickness on both sides of	8

bracket	
Angle Bracket w/ 3/8” Holes • One flap has triangular cut to make the lengths ¾” and 2” respectively • The other side is 2” long all the way • 1 ¼” thickness on both sides	8
Support Nut • 1” Long • ¼”-20 inner thread	8
Hex Nut • 1/4“-20 inner thread	104
Hex Bolt • 1/4”-20 thread • 1” Long	64
Washer • ¼” inner diameter	80
Hex Nut • 3/8”-16 inner thread	
Hex Lock Nut • 3/8”-16 Inner thread	8
Hex Lock Nut • ¼”-20 inner thread	16

G Final Budget

Departmental Costs	Donations	Overall Total Cost
$941.72	$629.45	$1,571.17

Product	Quantity	Unit Cost	Extended Cost
Linear Actuators			
Firgelli Automation [FA-PO-100-12-4"]	2	$121.86	$243.72
Firgelli Automation [FA-PO-100-12-12"]	1	$122.43	$122.43
USB Adapter			
U2C-12 USB-12C/SPI/GPIO Adapter	1	$88.75	$88.75
Sonar Sensors			
R287-SRf02 Devantech SRf02 Sensor	6	$26.50	$166.95
Motor Controller			
MD23	1	$187.50	$197.25
Motors & Mounts			
EMG30 DC motor & mount	2	$22.85	$45.70
Single Board Computer			
EFIKA	1	$239.99	$239.99

Flash Drive			
pqi i221 4GB Flash Drive	1	$14.99	$14.99
USB Hub			
Belkin F5U045 Hub	1	$6.99	$6.99
Relays			
MSS7 1A05	4	$3.00	$12.00
Integrated Circuits			
SN74LS03N	2	$0.54	$1.08
SN74LS00N	2	$0.57	$1.14
SN74LS04N	2	$0.51	$1.02
SN54LS00N	2	$0.56	$1.12
LM3524	2	$1.98	$3.96
LMD18200 H-Bridge	2	$8.00	$16.00
PCF8591F	3	$3.74	$11.22
LM7805	2	$1.29	$2.58
AD5161	2	$2.94	$5.88
Miscellaneous Electronics			
PCB	3	$8.95	$26.85
Heat Shrink	8	$1.05	$8.40
Slip on Jumpers (Power Bus)	1	$7.37	$7.37
3 Pos Term Block (Power Bus)	2	$1.85	$3.70
DPDT Toggle Switch	1	$1.05	$1.05
Fine Line Solder (12 ft)	2	$2.35	$4.70
Push Button Switch	1	$1.35	$1.35

Output Pins	2	$7.94	$15.88
#10 Ring Terminals	1	$1.95	$1.95
Power Supply			
Dynamite 10A 12VDC	1	$54.99	54.99 (without shipping)
Telephone Pole Tree	1	$30.00	$30.00
Body L-Bars w/Holes Threaded Rods (1/4 & 3/8") Bed Rollers	1	$106.98	$106.98
Nuts and Bolts 1/4" hex nuts 1/4" hex bolts 3/8" hex nut 1/4" lock nuts 3/8" lock nuts	1	$30.47	$30.47
Bike Chains	4	$15.00	$60.00
Bike Gears	12	$2.50 (estimate)	$30.00
Gear Box Materials	1	$30.00 (estimate)	$30.00
Final Printing			
B&W	124	$0.04	$4.96
Color	46	$0.49	$22.54
Binding/Covers	2	$1.10	$2.20

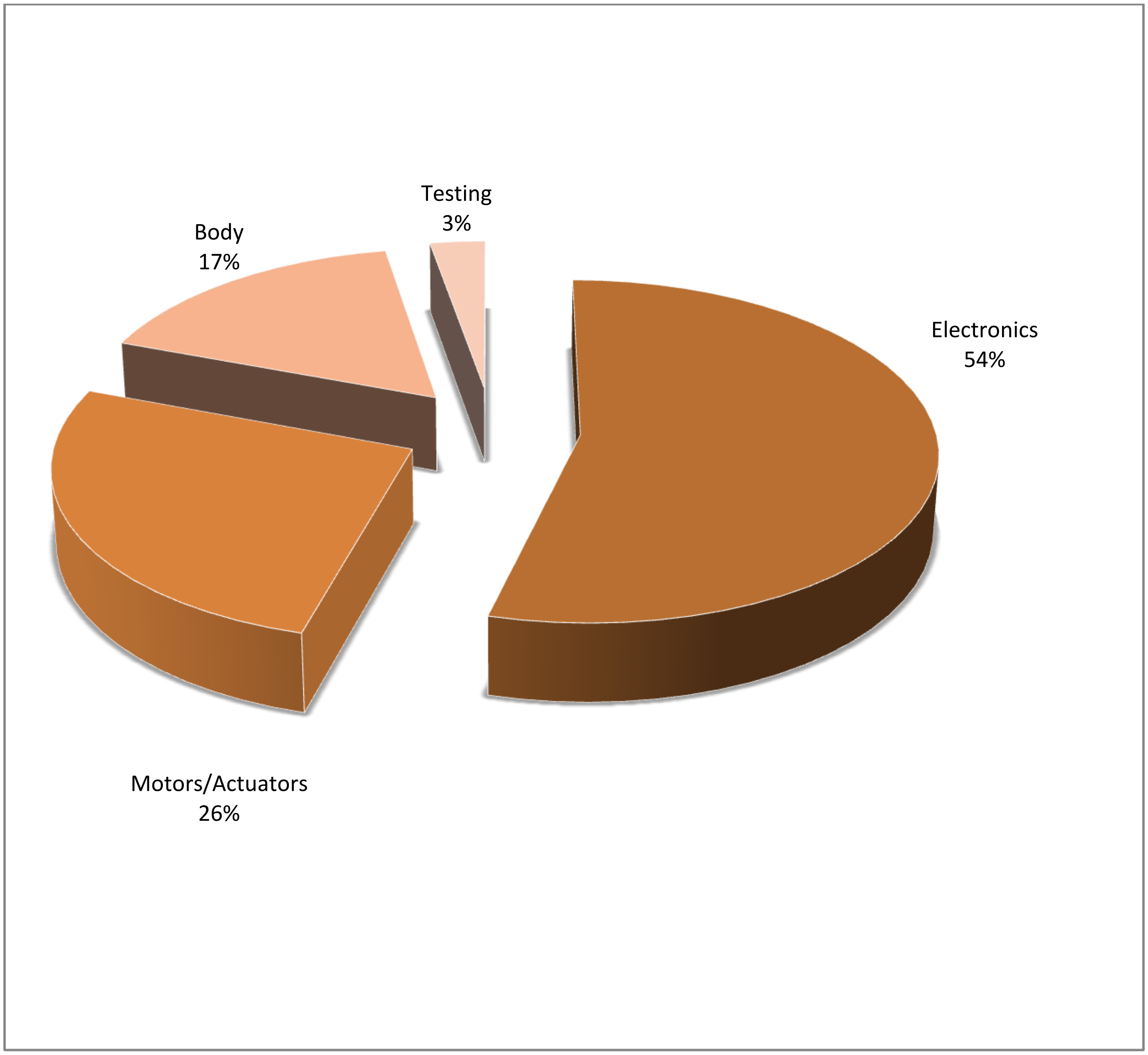


Figure G-1. Budget Breakdown

H Design Constraints

Increasing the safety of loggers motivated the construction of the robot, and thus precedes an accomplishment achieved by the robot. Currently, the logging industry presents one of the most dangerous jobs in the modern world, second to king crab fishing. Assuming the robot exhibits sufficient potential to undergo mass construction and implementation, and that future modifications incorporate a pruning saw, the death rates could decrease significantly. Currently, loggers are expected to scale some structurally unsound trees to successfully harvest and/or remove the trees. Upon scaling the tree, the logger must haul all the necessary equipment to prune branches prone to collateral damage when large chunks plummet to the ground. Climbing to these heights also increases in difficulty during inclement weather conditions such as wind or rain. Once trained to supervise the robot, workers could wait safely on ground level while the robot ascends the tree and prunes detrimental branches. To prevent logger injury from the falling branches, a proximity sensor would need to be implemented to postpone pruning until the supervisor or employee clears the area.

Most components of the robot are environmentally sustainable and recyclable. The body of the robot consists of primarily sheet metals that can be reused and recycle with minimal losses. However, complex circuitry and components such as the actuators are not easy to recycle. Similarly, various saw components might not exhibit recyclable traits. The construction of the robot might require the use of dangerous chemicals when synthesizing computer components, some of which might not be environmentally friendly. Lastly, depending on the composition, a battery probably does not exhibit recyclable traits.

Economically, the robot should work faster than the typical logger, thus maximizing profits earned by the logging industry. Also, since workers are not allowed to work in inclement weather by law, they could still proceed with logging with the robot. The robot does not need to stop for the weather, and the robot could also climb trees pending natural collapse. Finally, little maintenance should be required due to the simplicity mandated by the group from the beginning. Simple parts should be readily available, and implementation of replacement parts should be simple to attach.

Ease of manufacturability was another high priority of the group. Once the parts were chosen, the group brainstormed until a certain level of simplicity in assembly was reached. When comparing alternatives, the number of moving parts was limited, to reduce the chance of failure with the final product. Simpler design and fewer moving parts increased the reliability of the systems, thus making the robot fairly sustainable. Simpler parts also reduced the amount of programming necessary to make the robot maneuver around complex systems. For example, electrical feedback actuators were chosen deliver necessary output to the onboard computer. Also, the extension actuator was a different version of the same clamping actuators, thus reducing the number of unique code segments and circuits.