Design of an All-Terrain Rover

New Jersey Governor's School of Engineering and Technology

TANISHQ AGGARWAL
tanishq.aggarwal.11@gmail.com

WILLIAM LI
superwilliamli@yahoo.com

ABHINAV RAGHUNATHAN
abhinavr2121@gmail.com

GARGI SADALGEKAR
g2sadal@gmail.com

ANITA ZIRNGIBL
anita.m.zirngibl@gmail.com

July 22nd, 2016

Abstract

Ever since the onset of robotics, humans have been finding increasingly clever ways to traverse harsher terrain. There have traditionally been two defined robotic locomotion methods: wheels and legs. This paper describes the design and construction process of a third, new type of All-Terrain Rover (ATR) that uses a hybrid design involving both wheels and legs. The ATR consists of four wheels, with six linear actuators extending radially in a hexagonal pattern from each wheel, as well as two heavy arm-like structures that dynamically alter the center of mass of the vehicle. This design gives the rover significant versatility, allowing it to adapt to both flat terrain (which it can traverse via wheels) and rugged terrain (through the motional flexibility provided by the leg-like actuators.) Using computer-aided design software, different iterations of the wheel were designed, laser cutted, and constructed from baltic birch wood. A prototype of the rover has been fully assembled and is ready for testing.

1 Introduction

While the two methods of traditional robotic locomotion—wheels and legs—are effective in some areas, each alone is insufficient in every situation a rover would experience. While wheeled robots are efficient for crossing large, flat landscapes, they struggle when their path contains obstacles. In particular, obstacles greater than the radii of the robot's wheels are nearly impossible to cross. Legged robots are far more flexible and are capable of crossing a variety of terrain, but also suffer in terms of their physical and software complexity and speed of locomotion. The ATR combines the ideas of both wheeled and legged robots in a "hybrid" system. This design is able to handle various obstacles and can help humans in search and rescue missions by driving through the wreckage of natural disasters without risking more lives. It could also be repurposed and used to tackle the uneven geography of Mars.

This study continues on the heels of many ongoing experimental projects in unconventional robotic locomotion. These projects include BigDog of Boston Dynamics [1], and HyQ conducted by Claudio Semini and the University of Genoa [2], both of which focus on optimizing four-legged robots. Our design has several advantages over both designs, and is structurally capable of performing unique tasks that legged/wheeled robots would theoretically not be able to complete.

2 Background

2.1 Shortcomings of Wheeled and Legged Robots

Wheeled robots are far simpler to both build and program than legged ones. Their maneuverability is flexible on a level surface, especially those with wheels that utilize differential steering and are able to reach high speeds fairly easily. However, the lack of variety in their locomotive ability hinders the robot in situations that require a higher degree of flexibility (i.e. large obstacles or gaps).

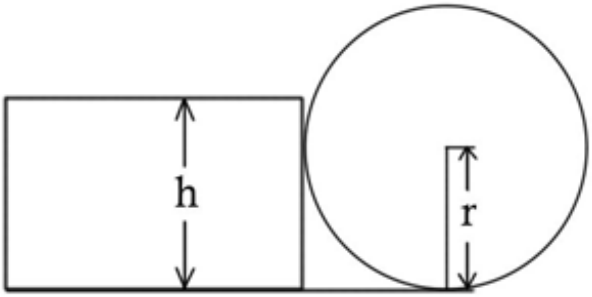


Figure 2.1: The height clearing capabilities are highly dependent on the radius of the wheel. [3]

As seen in the diagram above, a wheel is unable to clear a height greater than its radius and will only barely be able to clear one equal to its radius.

A wheel is also unable to cross a gap equal to or larger than its diameter. For these reasons, a wheeled-rover is severely crippled on rough terrain.

Legged robots have a distinct advantage over wheeled robots in that they are able to climb steps, cross large gaps, and walk on rough terrain by altering their geometry. However, their ability to handle such obstacles comes at the cost of simplicity. Legged robots often require complex designs and programming in order to yield the coordination needed for the limbs to function in unison.

In his essay, *Principles of Robotic Motion* [4],

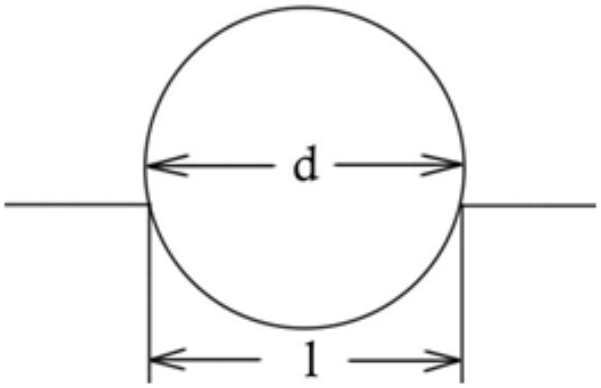


Figure 2.2: The gap clearing capabilities are highly dependent on the diameter of the wheel. [3]

Böttcher derives an equation for the total number of lift and release event combinations, N, of any legged robot based on the number of its legs, k. Lift and release event combinations refer to the combinations of raised and lowered legs where the only two positions possible for each leg are raised and lowered.

$$N = (2k-1)! \tag{1}$$

Based on Equation 1, a four legged robot has 5040 leg positions with just the lifting and releasing mechanisms. Additional joints provide even more flexibility which will require more servos and make programming correspondingly more cumbersome.

A legged robot, despite having wide degrees of freedom in its leg positions, is still limited in other capacities. For instance, the length which a legged robot is able to travel is always restricted by the length of its legs and how fast it can move them. This means its speed is fundamentally slower than that of a wheeled robot. Similarly, a robot can only climb over or onto obstacles shorter than the height of its fully bent leg. These limitations highlight the discrepancy between the high degree of complexity of a legged robot and its limited actual maneuvering capabilities.

Therefore, a design that enables a rover to al-

ter the radius of its wheels allows the rover to have several types of locomotive function, making the programming simple yet capable of great locomotive flexibility.

2.2 Altering Orientation

In addition to the increased maneuverability provided by the linear actuators, a robot that is able to alter its orientation drastically increases both its height and gap clearing capabilities. A robot that can flip over no longer uses only its wheels to overcome a height and instead uses its entire body. This increases both of the distances (height and gap) that the rover is able to clear immensely.

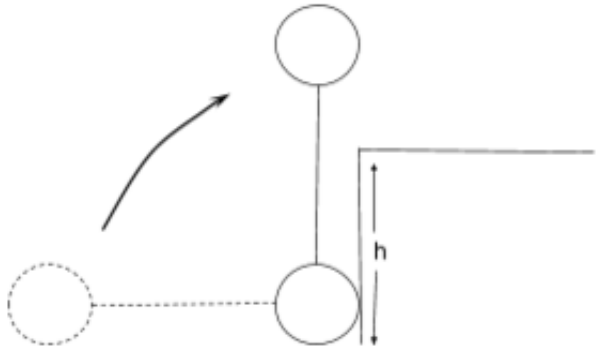


Figure 2.3: Height clearing capabilities of a rover that is able to alter its orientation.

In order to minimize the torque necessary for the motor to exert when flipping the rover, the center of mass of the rover must be shifted to one wheel axle, which becomes the axis of rotation. This torque (which balances the robot on one wheel axle, in preparation for the orientation alteration) is based on the equations that define torque:

$$\tau = \boldsymbol{F} \cdot \boldsymbol{d}$$

$$\boldsymbol{F}_r \cdot \boldsymbol{d}_r = F_a \cdot \boldsymbol{d}_a$$

Taking magnitudes, we have

$$M_r g d_r = M_a g d_a$$

$$M_r d_r = M_a d_a \tag{2}$$

Where M_r, M_a are the masses of the rover and arm, respectively, and d_r, d_a are the distances from the axis of rotation to the center of mass of the rover and arm, respectively.

3 Design Process

3.1 Wheel Design

The original concept for the ATR wheel was designed with six telescopic linear actuators that extend from its center. In order for the rover to traverse on actuator legs without the wheel contacting the ground, the actuator has to extend at least twice the length of the radius [3]. The available linear actuators simply cannot do this on their own, but the telescopic mechanism doubled the extended length of a linear actuator, and coupling the available actuators to the telescopic ones solved the problem.

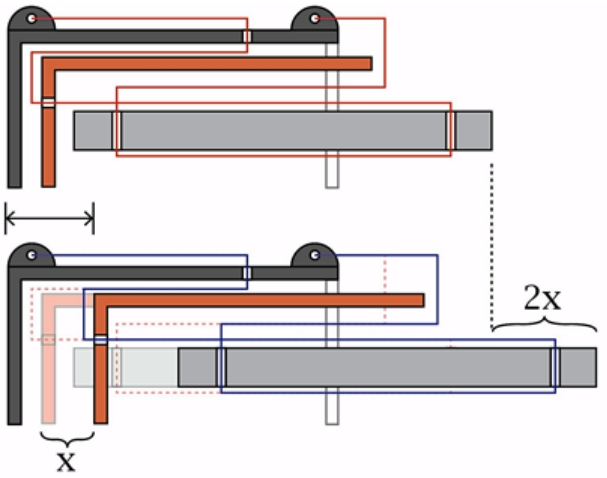


Figure 3.1: Telescopic linear actuator design [3].

The telescopic linear actuator is essentially a compound pulley system (see Figure 3.1) in reverse. It doubles the extension of the inner actuator via the appendages attached around it. As seen above, the telescopic mechanism requires a string wound around an inner and outer tube, both of which surround the actuator. The string

is fed through channels in each of the tubes; these channels act as the surface of pulleys in the compound pulley system.

The problem with this radial design was the sheer amount of material and time needed in order for the wheel to be constructed: each telescopic actuator requires (in simple terms) an inner tube, an outer tube, plates on either end of the tubes to connect them to each other as well as to the actuator, and string. While the string is not expensive, the process of tying it in a configuration taut enough to allow for the telescopic mechanism to work would be a hugely time-consuming task for 24 separate actuators.

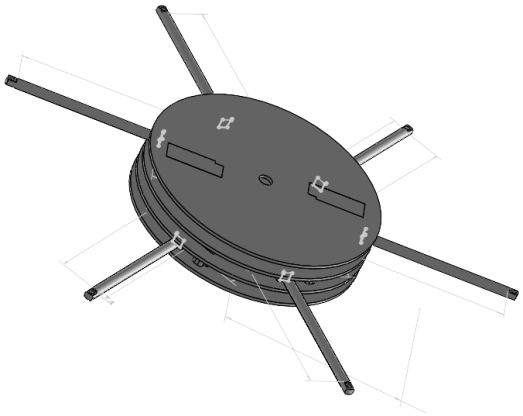


Figure 3.2: A CAD of the second wheel design depicting the three layers.

As a result, the design for the wheel was heavily modified. The second design for the wheel (Figure 3.2) involved layers of circular plates with a pair of non-telescopic linear actuators fitted between two layers, yielding a total of three actuator pairs between four circular plates. Within each layer, the two actuators would be placed in opposite directions, with their lengths spanning the diameter of the plate. The actuators were also arranged in six symmetrically spaced directions. A foot would be attached to the head of each actuator and fit between the plates. This design was temporarily rejected due to weaknesses in the foot found via FEA, mostly owing to width constraints posed on the foot by the small amount of space available between the plates.

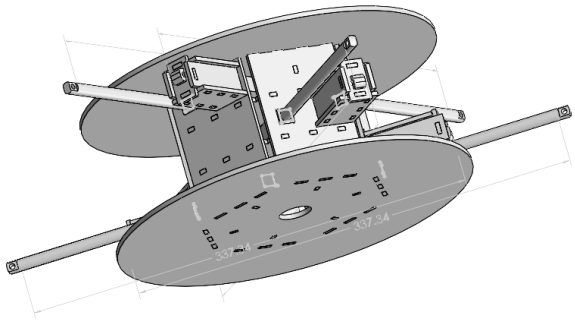


Figure 3.3: A CAD of the third wheel design.

The third iteration of the wheel (Figure 3.3) consisted of six planks joined in a hexagonal pattern and encased by two circular plates. Each pair of opposing planks had cutouts that fit around the shafts of a pair of actuators that faced opposite directions and spanned the diameter of the wheel. Attached to the outer face of each plank was a cage that surrounded the actuator's main body in order to prevent it from falling out. The foot, when attached to the head of the actuator, was intended to fit with the perimeter of the outer plates, so its thickness was no longer significantly limited and thus would be stronger. However, in order to fit both the foot and the cage within the perimeter of the plates, the diameter of the circular plates needed to be increased. Doing so not only violated materials constraints, but would cause the length that the actuators extended beyond the wheel to decrease to the point that the fully extended length of each actuator from the center could no longer be twice the radius of the wheel.

In the final version (Figure 3.4), the base of each actuator is attached to the edge of the wheel, and its body spans the entire diameter of the wheel. In this regard it is similar to the second iteration, except instead of the plates being circles, they are approximately concave polygons with several cutouts for support structures. The cutout with the greatest function is the cutout for the actuator body. By partially en-

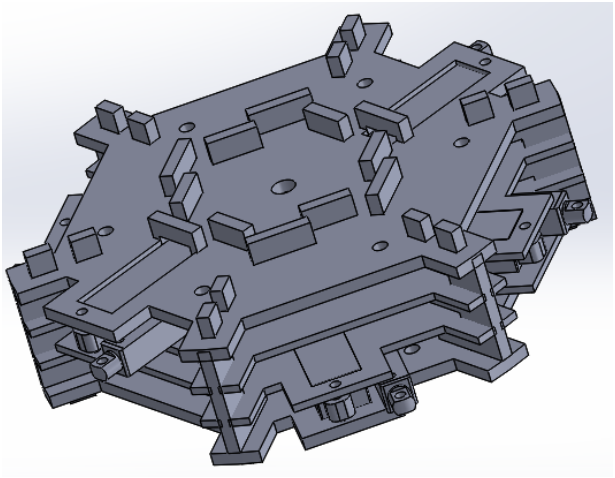


Figure 3.4: A CAD of the final wheel design.

casing part of the actuator body within the plate itself, the actuator shaft housing can rest flush against the surfaces of two adjacent plates, leading to significant structural integrity. Several more cutouts (patterned in concentric hexagons) are channels for rectangular pieces with holes designed to fit around the actuator body and shaft housing, thus anchoring them in place with respect to the lateral direction. Finally, several cutouts on the edges of the plates allow the connection of six E-brackets (Figure A.2) that holds all of the plates together.

3.2 Foot Design

The general design of the foot remained constant throughout the design as an isosceles trapezoid with an arc joined to the larger of the bases. The foot attaches to the actuator head and sits within the perimeter of the wheel when the actuator is retracted. When extended, the actuator pushes the foot outside the wheel perimeter where it makes contact with the ground at about a 30 degree angle. As the design for the wheel changed, the design for the foot had to be adapted. The original design included a solid foot in the aforementioned configuration that fit between the two face plates. In order to save material and fit into the constraints of newer wheel designs, the solid foot design was replaced with a frame design where two end plates shaped in the original configuration are connected by two side plates that were connected using a tab design. The actuator head and its connecting pieces are housed within the hollow center and top portions of the foot (see Figure 3.5).

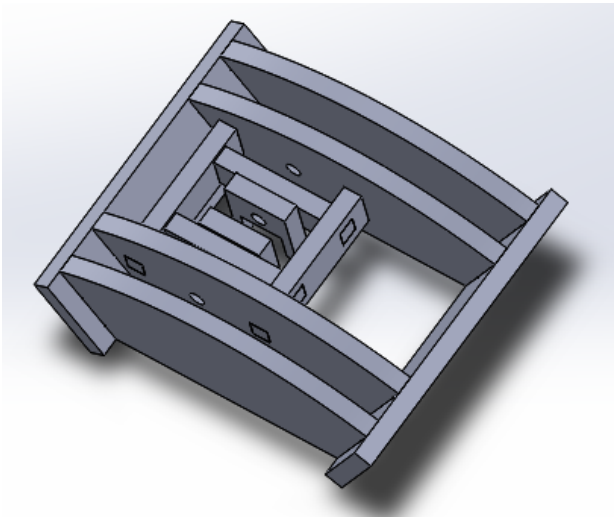


Figure 3.5: A CAD of the final foot design for the central actuators.

The assembly in the center of the foot is what the head of the actuator connects to and what provides structure to the rest of the frame.

3.3 Arm Design

The original design of the robot included a single arm which provided additional locomotive capabilities. The arm extends from the center of the rover and had 360 degrees of mobility, allowing it to function even when the rover had drastically altered its orientation. The arm design featured double joints, adding another degree of flexibility. The original arm design functions similar to a limb by adding support when needed and using its rigid structure to help the rover overcome large obstacles (see Figure A.1 for a demonstration of function).

The revised design for the arm not only alters its design, but is also a major part of its function.

In order for the rover to be able to drastically alter its orientation, a certain degree of torque must be applied in order to reduce stress on the arm motor. The arm's new purpose is to provide this relief. Although a variety of designs were considered, it was decided that the rover design should include two arms. The arms are both single jointed and are each comprised of a weight that can be moved by a pulley. The arms are attached to the two opposing edges of the chassis (in close proximity to the two wheel axles) and pass through slits that span the length of the chassis. When the ATR changes its orientation, the arm attached closest to the axis of rotation is rotated so that it extends outside the perimeter of the rover. The weight can then be moved to the end of the arm so that it creates the torque required to flip the rover.

3.4 Programming

3.4.1 Control Flow of Directional/Actuator Commands

Figure 3.6: Diagram representing the flow of information and commands: Python scripts and the GUI provide , which transfers commands to Arduino, which enacts those actions on the actuators and wheels.

The ATR is built with a series of programmatic components that interact with each other to allow the operator full control of the rover's motion. First, the hardware that directly controls the actuators and the wheels is an Arduino Uno board. The board receives commands from a remote computer that issues directions in Python, and transfers these commands to its subsequent components.

3.4.2 Controlling Actuator Extension Length

The ATR's functionality depends heavily on the user's ability to extend and retract the actuators to specific lengths. To facilitate user experience, the code created accepts a desired extension (in millimeters) and extends to that length. To process the extension amount into an analog output to the actuator, the program uses a function created by Arduino's map function to transpose the extension length to an analog value.

After multiple trials of testing the extension, a consistent amount of error was observed. The data collected was graphed (Figure 3.7) and a linear line of best fit with an R-squared value of 0.9994 was observed. Using this trend line to transform the input data before mapping, the actuator code yielded accurate extensions with a margin of error around 1 millimeter.

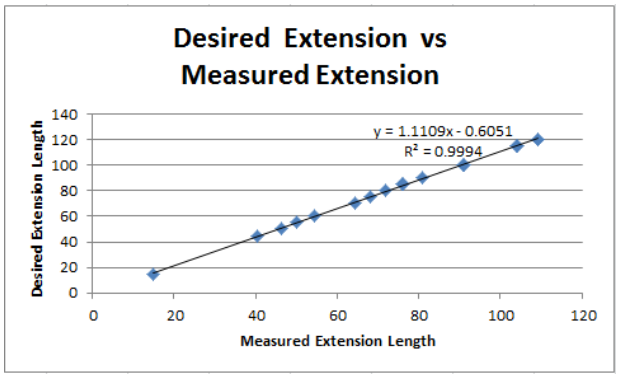


Figure 3.7: Actuator extension calibration results.

3.4.3 Python-Arduino Communication

Because the Arduino has limited memory and computing power but is useful for processing and relaying low-level commands, a more capable platform was necessary for programming the ATR. Python was chosen for its versatility and its ability to be easily integrated with Arduino and OpenCV. Python was setup to communicate with the Arduino board via Serial, using a code consisting of a sequence of characters and

numbers. The user input is processed through a Graphical User Interface and passed to a Python script. The Python script then passes information to the Arduino via a simple alphanumeric protocol, and the Arduino's low-level control methods execute the actuator and motor movements.

3.4.4 Rotational Position

Because the wheel does not have built-in motor encoders, it was necessary to devise an alternate method for determining the exact radial position (in degrees) from the origin. OpenCV was our first attempt at detecting the rotational position of the wheel without extra motors or gyroscopic sensors. OpenCV (Open Computer Vision) is a computer vision library that allows a script (written in Python) to get and process individual frames from a live video feed (via webcam positioned behind the wheel) and look for specific symbols to denote positions of the wheel. For example, the origin will have a specific shape that the script can identify and label as the origin. Within OpenCV, we attempted to use Contour Matching/Reduction to identify 12 unique shapes around the circumference of the wheel which signified different degree measures. However, the problem with OpenCV is the difficulty of its physical implementation, since it requires a webcam mounted behind the wheels, and also requires a computer program to allow the contour matching to take place.

After OpenCV was diagnosed to be overly arduous to program and implement, it was necessary to create an easy alternative for a rotational position detector. Stepper motors are the most efficient solution—electromagnets that are built within the stepper motor allow it to recognize its rotational position, and can be accessed via a Python script.

In order to maintain the best accuracy and eliminate the chance of any overlap between the symbols on the back of the wheel, it is important to test the size of the field of vision and to orient the camera in a way that allows it to focus on only one symbol and not its neighboring symbols. In this way, the camera was calibrated to detect only one image that is easily readable.

3.4.5 Graphical User Interface (GUI)

In addition to the script that controls the actuators and wheels, software to easily edit the speed of the wheel and the retraction and extension distance for the actuators is crucial. Continuing the use of Python within the software side, a Graphical User Interface (GUI) was developed (see Figure A.3). The GUI, written in Python (using the Flask web microframework [6]) and JavaScript (CreateJS [5]), provides an easy way to select an actuator and extend and retract it with a specified distance. On the GUI, the actuator is located by Wheel Number (1 through 4), Side (R or L) and Actuator Letter (A - F). Furthermore, the GUI contains the trigger for the Python-Arduino interaction (explained in "Python and Arduino Communication"). In addition, the GUI graphically shows the radial position of the wheels with regard to the origin with a degree measure from the stepper motor Python script (explained in "Rotational Position"). With this information, it becomes possible to correctly select the actuator that needs to be extended and retracted for the robot to continue its motion. In addition to the actuators themselves, the GUI also allows for the efficient control of the arms and the orientation system. The arms are controlled with a Arm Side (R or L) and a Arm Rotation Degree (where 0 degrees is fully folded and 90 degrees is fully extended). The orientation altering system can be controlled with a Weight Side (R or L), and a Position ("Bottom", "Middle", or "Top"). With all of this functionality built within the GUI and within the control flow, the ATR can efficiently and accurately be selectively automated.

3.5 Electrical Integration

One of the obstacles with building the ATR stems from the process of connecting wires from a stationary frame to rotating wheels. A direct, wired connection would result in the twisting of wires, so slip rings are needed for attachment. Slip rings have an inner cylinder and an outer shell that rotate around each other. By keeping the inner cylinder fixed to the wheel axle and allowing the outer shell rotate with the wheel, the wheel can spin (relative to the ATR) without having its wires becoming entangled.

4 Assembly Process

4.1 Wheel Assembly

Figure 4.1: Completed wheel assembly using parts assembled from laser cut baltic birch and corrugated plastic.

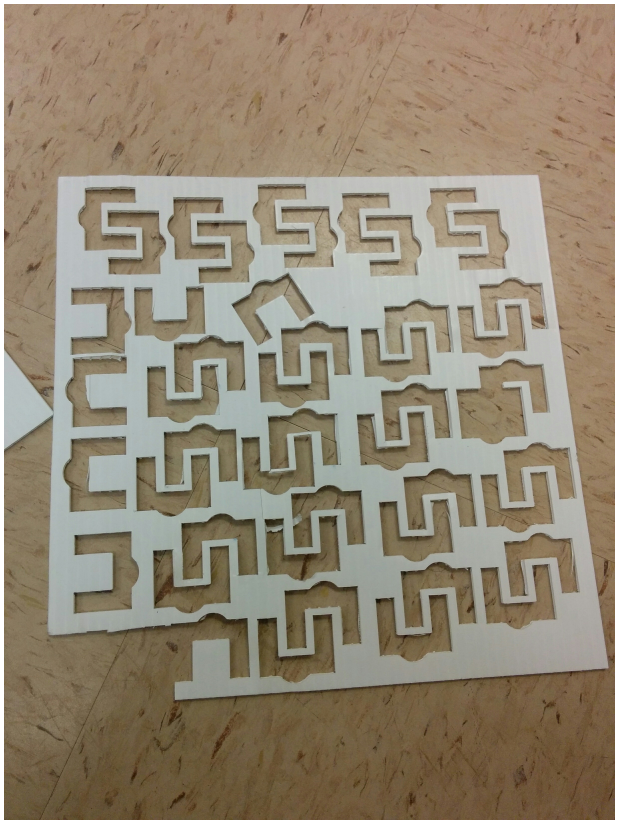


Figure 4.2: Laser cut corrugated plastic inserts for insertion between the actuator body mounting holes and the wheel plates.

The majority of the wheel structure is laser cut from baltic birch wood (Figure 4.1). The assembly of the laser cut wood parts is completed in a specific order so that all the parts to assume their proper positions within the structure. First, the inner support structures and inner base structures are inserted into the inner two plates. Then the actuators are inserted before the 'E' support structures are added to the edges of the plates. Spacers laser cut out of corrugated plastic (Figure 4.2) are inserted between the mounting hole and the plates at the back of each actuator, through which a bolt will eventually be inserted. Finally, the outer plates enclose either side of the wheel. When fully assembled, the tight fit of the parts create enough friction to hold the wheel together without any bolts, a phenomena called "press-fitting". How-

ever, bolts are inserted through the wheel as well as through the actuator body mounting holes for additional security.

4.2 Foot Assembly

The entire foot is also laser cut from baltic birch. The process in which the foot is assembled is done from the center of the foot outward in order to maximize efficiency. First, a spacer bolt connector and a spacer are placed on either side of the actuator head and a bolt is inserted through them. Then a spanners are attached to either side of the spacer bolt connectors via their tabs. The spanners are then sandwiched by two connecting plates, which are held together by side plates. Finally, outer plates are inserted between the two side plates and the side plates are super glued to the connecting and outer plates for additional security.

4.3 Chassis Assembly

The chassis was primarily constructed from VEX robotics kit components supplied by the IEEE club, with some parts sourced from non-functional 3D printers donated by WINLAB. The chassis consists of two channels on the sides fit together with long, thin structural elements, along with two similarly sized channels that house the arms. The arm channels are attached via a large gear to the rest of the chassis frame.

The two side channels house the drive train for each of the two wheels. To increase torque on the wheel drive, a gear system was used that made several interchanges between VEX and stepper motor-type gears (Figure 4.3). The axles are smooth stainless steel axles and are placed within ball bearings tightly encased between two wooden plates bolted to the metal VEX side channel.

The arm channels highly gear up a standard VEX motor to minimize the torque the mass of the arm exerts on the motor itself (Figure 4.4). The final gear interfaces with two connector pieces rigidly attached to the rest of the frame, so that as the gear rotates, the arm raises or lowers depending on the direction of rotation.

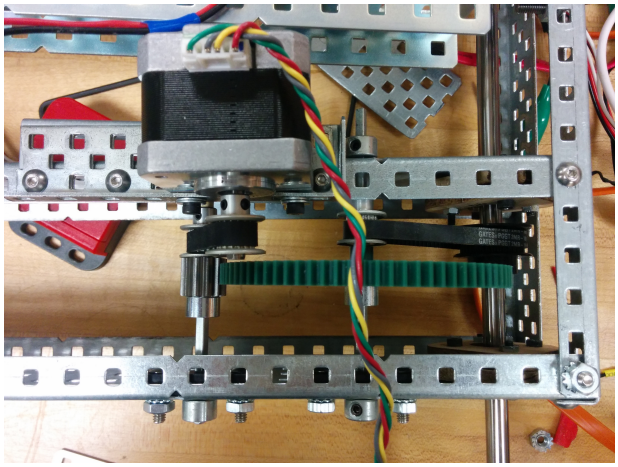


Figure 4.3: The drive train for each of the wheel motors; four such drive trains comprise the wheel locomotion system.

As of now, time constraints have prevented the construction of a pulley system on top of the arm in order to move a mass up and down the arm, but preliminary tests involving tying a mass once the arm is set to the desired position still produce the desired results.

5 Results

The rover has been fully constructed and is ready for rigorous testing. Several sub-assemblies within the rover have been tested for structural integrity and for functional accuracy, and these tests are listed below.

5.1 Wheel

The wheel demonstrated exceptional structural integrity even without its supporting bolts put in place. The frictional force between the plates was sufficient to keep the wheel together. This demonstrates the viability of the construction method of attachment via interlocking tabs.

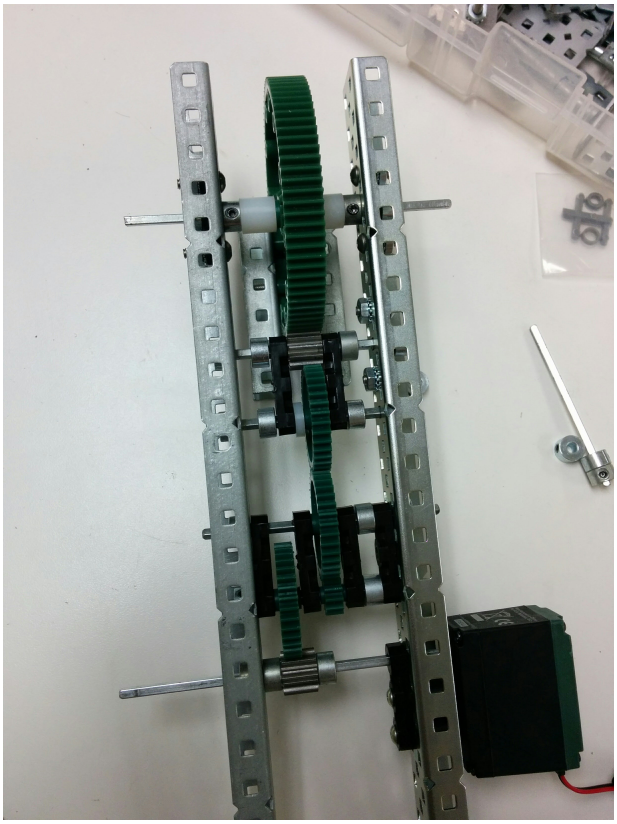


Figure 4.4: The gearbox in place on each arm to maximize torque from the VEX motor and actually turn each wheel.

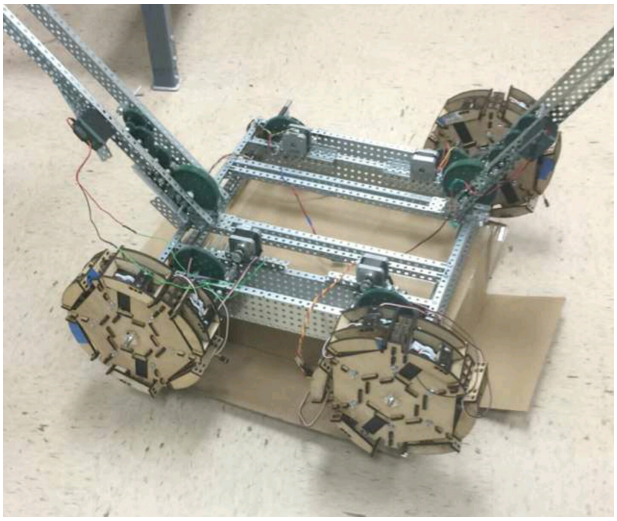


Figure 5.1: A picture of the assembled ATR, missing one wheel due to initial mounting issues.

In addition, the wheel's actuator mechanics have also been tested thoroughly. Using the Adafruit motor control board in conjunction with the Arduino code, each actuator can be controlled independently of another, so that several actuators can even be controlled simultaneously.

5.2 Foot

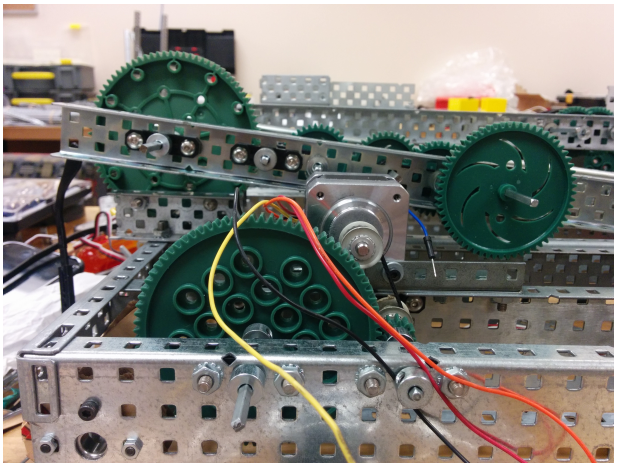
The foot as a whole is structurally sound, but due to the inherent rotational freedom in the actuator shaft, the foot slightly wobbles in place. Testing still needs to be done, but this wobble is likely not an obstacle to effective locomotion.

5.3 Chassis and Arms

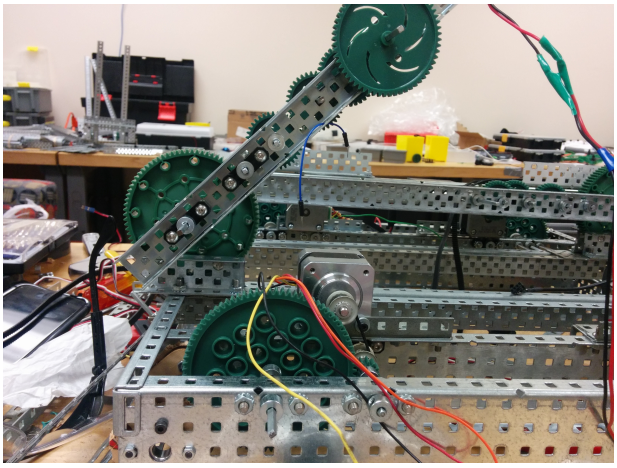
The chassis has been fully assembled to house the four drive trains and to provide an anchoring point for the arms (Figure 5.1). The two arms have a good amount of physical margin from the frame (see Figure 5.2), and are capable of 180 degree rotation.

6 Conclusion

Several separate design goals have been met for this project. The design for a wheel with linear actuators has been shown to be viably constructible using a press-fitted part design. The design of the wheel is effective because it resulted in a wheel that is structurally sound on all sides, while allowing the actuators to extend to the necessary length past the wheel. While the foot is wobbly due to rotational freedom in the actuator shaft, it is structurally sound. A viable chassis design has been constructed. Low-level control software has also been written for the control of the actuators, the stepper motors driving the wheels, and speed control for the arm motors.



(a) Arm Lowered



(b) Arm Raised

Figure 5.2: Images of one of the rover arms in a lowered/raised configuration.

6.1 Further Testing

The assembly of the rover has been completed, but due to time constraints the rover has not been tested. Below, some hypothetical tests and associated metrics for the rover's capabilities are listed.

6.1.1 Height Clearing Capabilities

The rover's height clearing capabilities will be tested by having the rover drive over heights starting at 110 mm, the radius of the wheel, and increasing in increments of 10 mm in order to gauge the maximum height that the rover can clear. A height would be declared successfully cleared when the rover was able to cross it with a success rate of 90% .

6.1.2 Gap Clearing Capabilities

The rover's gap clearing capabilities will be tested by programming the rover to cross a gaps of various widths. The widths will begin at 220 mm, the diameter of the wheel without extended linear actuators. The widths will be increased at increments of 20 mm until the rover is no longer able to successfully clear the gap. A gap will be declared successfully cleared when the rover is able to clear it with an 90% success rate.

6.1.3 Various Terrains

The rover is designed to be able to navigate a variety of terrains. The success of the rover on each terrain will be measured by the distance that the rover travels in 60 seconds. These terrains will include a flat surface such as a floor, a rough surface such as a gravel plane, and a soft surface such as mud or sand.

6.2 Further Steps

6.2.1 Assembly

It is unknown whether or not the wheels are capable of supporting the weight of the chassis at this point. It is inadvisable to try since the Arduino is probably not capable of providing the current to the actuators needed to keep the rover suspended at a certain altitude. The maximal stall current of the actuators is 650 mA according to its datasheet [7], and multiplying by 12 (for the three actuators in contact with the ground per wheel) results in a total current of 7.8A, which is ridiculously high for a microcontroller board. While this estimate is higher than current draw, a current value within that ballpark is probably likely due to the high weight of our metal chassis causing the actuator motors

to exert greater torque. In addition, the low lateral load rating of the actuators (20 N according to [7]) makes the weight of the chassis a more limiting factor mechanically rather than electronically. In order to be able to feasibly test the rover's maneuverability, the chassis probably needs to be reconstructed from a lighter material, i.e. wood in order to be feasibly lifted by the actuators, but in the meantime this chassis serves as a useful concept design.

The arms remain to be fully constructed. While their extension and retraction system has been developed, due to time constraints a pulley system was only constructed on one arm, and the design has not yet been tested to analyze the maximal weight the VEX motors (both the arm extension/retraction as well as the weight extension/retraction motors) can support.

Of course, nearly the entirety of the design is not feasible for field use. A chassis built out of wood is definitely not feasible for rugged rescue operations. In addition, while the puzzle-like design of the wheels simplified construction, was an interesting design study, and resulted in relatively sturdy construction, the wheels are still not rugged enough for use in field operations. In particular, the wobble on the feet due to the rotational freedom in the actuator shaft can induce unwanted stresses on the foot assembly. Later iterations of the wheel and feet need to be redesigned with tougher materials and better attachment methods.

6.2.2 Electronics and Software

From a software point of view, automation is a major step forward in the future. Even if the robot does not become fully autonomous, patterns of code should be developed for predictable scenarios, such as crossing a gap or climbing a cliff. To increase the ATR's capabilities, more sensors need to be connected since the ATR is effectively a system driven by dead reckoning as of now. For example, cameras and distance sensors will help the ATR detect obstacles in its path and predict the correct maneuver to perform. Optical encoders are also necessary for the arm motors. With more sensors, the robot can observe its environment and learn to match maneuvers to scenarios.

7 Acknowledgments

The authors gratefully acknowledge Michael Sanzari of Rutgers University, for without his technical experience and sincere dedication to this project, it would not be possible. The authors also thank the Rutgers University branch of the IEEE for allowing them to use their facilities and resources. The authors appreciate the work of the Rutgers Makerspace workers who helped laser cut the wheel parts.

The authors would also like to take this time to thank Rutgers University's Wireless Information Network Lab (WINLAB), which allowed the authors to use its equipment and facilities. The authors sincerely appreciate the generosity of Charles McGrew from the Rutgers Hackerspace, who loaned the linear actuators that made this entire concept possible. The authors would also like to thank Alexander Hobbs, who aided research as project mentor by providing transportation and guidance, and to the New Jersey Governor's School of Engineering and Technology (GSET), the program under which this research was conducted. The authors would like to acknowledge the Deans of the program, Ilene Rosen and Jean Patrick Antoine, for their support and review of the project through the program. Finally, the authors extend their appreciation to corporate sponsors of the GSET program: Lockheed Martin, South Jersey Industries, and printrbot.

8 References

References

[1] M. Raibert, K. Blankespoor, G. Nelson, et. al, “BigDog, the Rough-Terrain Quaduped Robot” , Boston Dynamics, Waltham, MA, 2008.

[2] C. Semini, “HyQ - Design and Development of a Hydraulically Actuated Quadruped” , Ph.D. dissertation, Dept. Advanced Robotics, Univ. Genoa, Italy, 2010.

[3] M. Sanzari, “Experimental Mars Rover Design: Telescopic Linear Actuators”, term paper, Rutgers’ School of Engineering, 2016, (unpublished)

[4] S. Böttcher, “Principles of robotic locomotion”, 6-7, WWW Document (`http://www2.cs.siu.edu/~hexmoor/classes/CS404-S09/RobotLocomotion.pdf`)

[5] ”EaselJS v0.8.2 API Documentation : EaselJS”, Createjs.com, 2016, WWW Document (`http://createjs.com/docs/easeljs/modules/EaselJS.html`)

[6] A. Ronacher, ”Flask Documentation (0.11)”, Flask.pocoo.org, 2016. WWW Document (`http://flask.pocoo.org/docs/0.11/`)

[7] *L16 RC Linear Servo.* L16-R. Rev C. Firgelli Technologies Incorporated. 2016.

A Figures

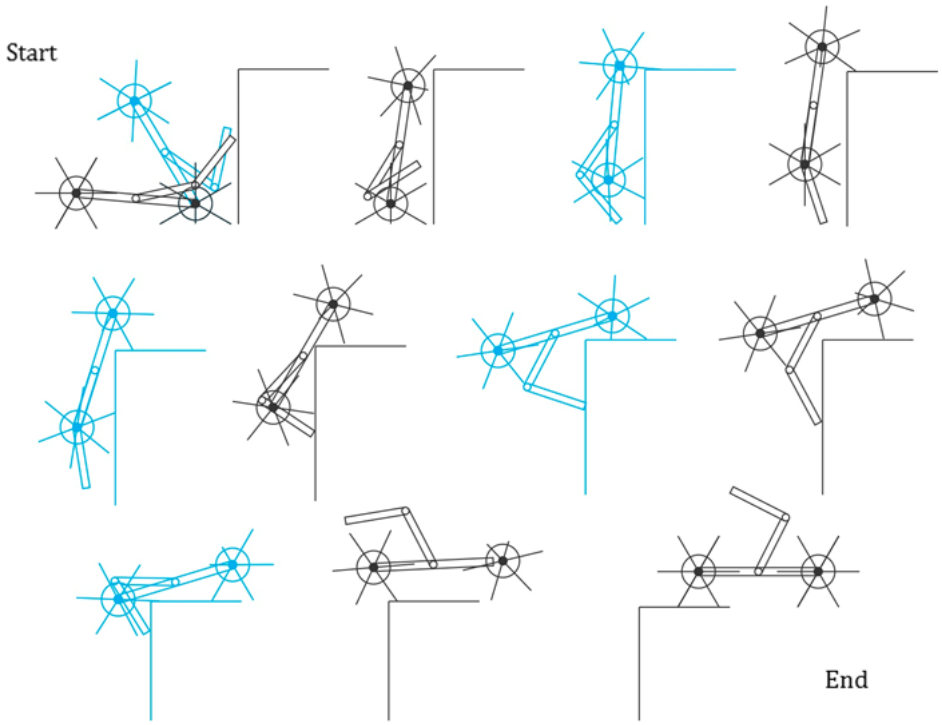


Figure A.1: A diagram depicting a potential use for the arm [3]

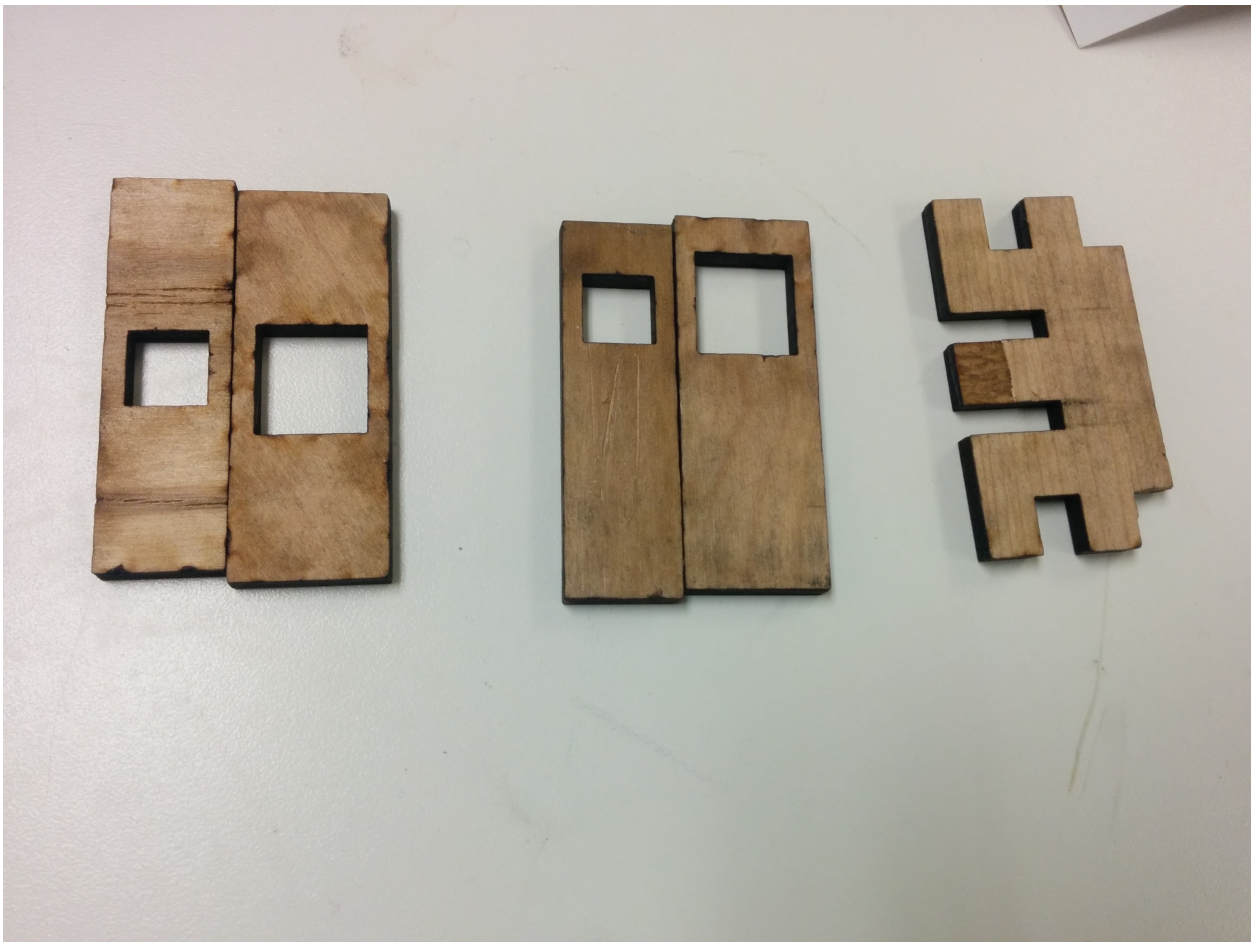


Figure A.2: Connector piece designs used in the final version of the wheel. From left to right: central actuator shaft housing support tab, central actuator body support tab, side actuator shaft housing support tab, side actuator body support tab, distal E bracket.

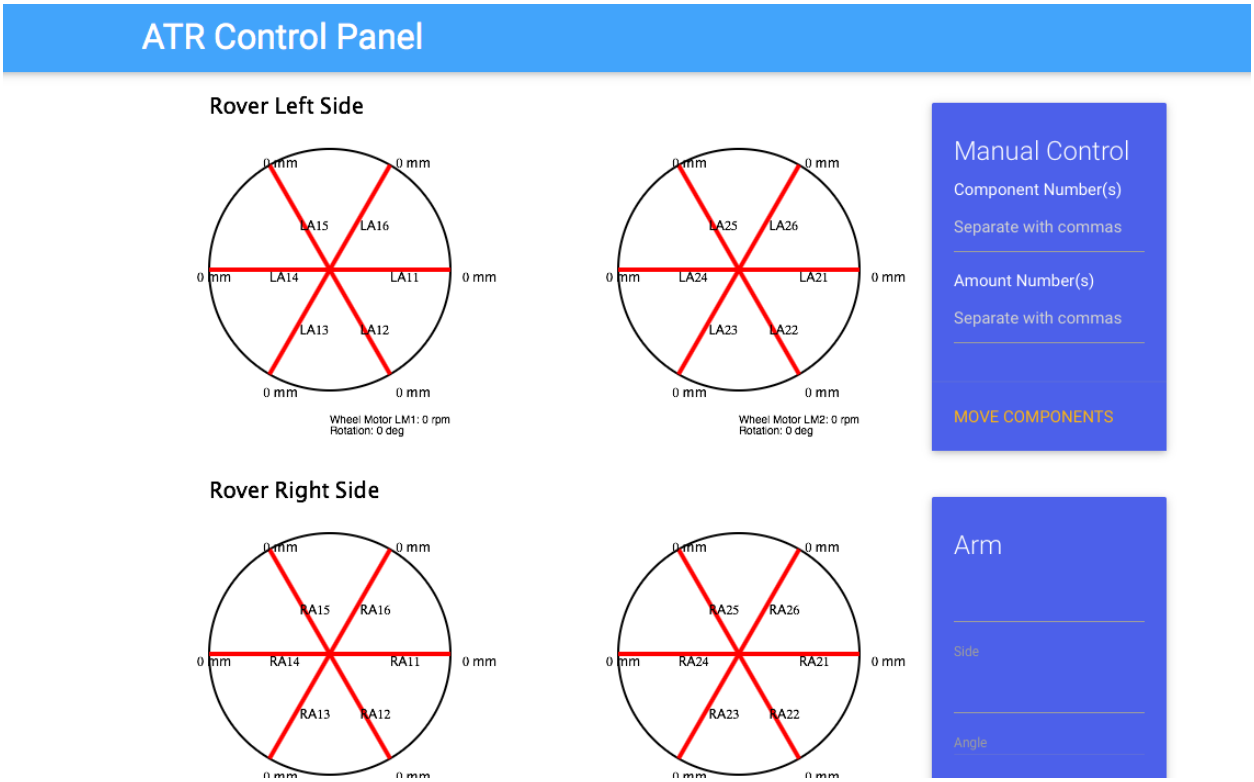


Figure A.3: A partial view of the graphical user interface.