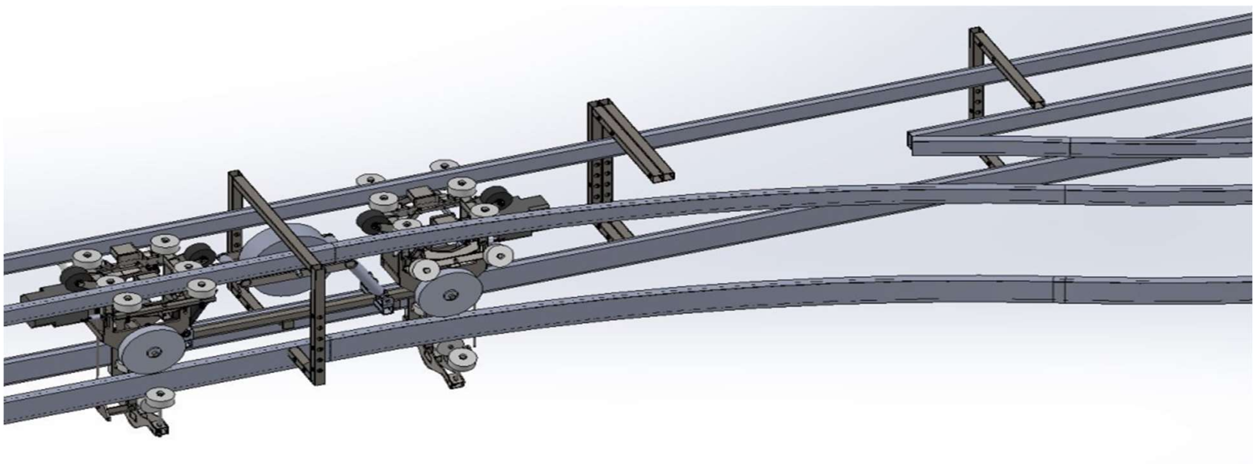
Spartan Superway
A Solar Power Automated Rapid Transport Accelerated Network

Half-Scale Bogie Design and Controls



May 19, 2019

Oscar Balvaneda
Karandeep Binner
Hector Gomez
Matthew Leal
Karen Tan

Advisor: Burford Furman

SJSU SAN JOSÉ STATE UNIVERSITY | SPARTAN SUPERWAY

Abstract

The Spartan Superway is a new concept for conventional transportation systems that will make use of the airspace while minimizing ground level interference. The 2018-2019 half scale bogie design and controls team continued this research by furthering the designs and mechatronics on previous contributions to the half scale prototype. The goal for the school year was to complete a fully working prototype of the half scale system. The previously implemented wayside power system was incomplete, so the team put it on hold for future teams to tackle. Over the winter the Superway moved to a different location, so the existing guideway needed to be modified.

Both the mechanical design and the mechatronics systems of the bogie were overhauled and substantial improvements were made to both systems. On the mechanical design side, metal chains and sprockets for the steering systems were implemented and new shafts were manufactured for additional bogie strength and stability. The team also secured the lower steering arm and prevented in from slipping by adding a D-shaft. Fixtures were designed, 3D printed, and attached to hold the hall effect sensors, ultrasonic sensors, and limit switches. On the mechatronics side, the team put a reliable powertrain and steering system into effect which is extremely necessary for a bogie to function autonomously. The team also made careful use of sensors by incorporating hall effect sensors for velocity detection, ultrasonic sensors for proximity detection and limit switches to prevent oversteering. The team also made sure that all the wiring is well-labeled and all the research data is well documented for the future teams to easily understand and carry the project forward. A master code was compiled that controlled the steering, powertrain, sensor integration, and automation of the bogies. There are manual and autonomous control modes for troubleshooting and testing functionality, respectively. A control box was designed to be suspended from the bogies to reduce clutter, but it was not assembled into the final design.

Overall this year's team's efforts will act as a guide for the future half scale teams. All the modifications and additions have proved to make the bogie more stable, reliable, autonomous and has improved its functioning by leaps and bounds. Future teams still have ample contributions to make. On the mechatronics side, future teams could contribute to establish a wireless communication network between the master Arduinos and slave Arduinos and working on the LCD screen code. In addition, on the mechanical design side, future teams must find a way to use wayside power or find a portable power source to give the bogie more freedom to move and showcase the proof of concept even more boldly. In addition, the track switching mechanism needs smaller modifications such as replacement of tire to allow for better grip on the tracks. Also, future half scale teams can expand their forte of design specifications by expanding the track and incorporating the 17° incline and decline sections of the track as well. In future, teams should also

test the bogie and controls with actual weight suspended from it especially on inclines to test the maximum capabilities of the system.

Acknowledgments

On behalf on the 2018-2019 half-scale team, we would like to thank Dr. Burford Furman, Eric Hagstrom, and Ron Swenson for their efforts in helping all the senior project students with properly presenting our research and proposing solutions to problems that arose throughout the project. We give thanks to colleagues and working professionals for helping us on the final touches of the project. Our fellow Spartan Superway teams assisted in the relocation and setup of the new half-scale track. Dominic Mutimer and Fernando Casas deliberately dedicated time in soldering and assembling the bogie. Their efforts are massively appreciated, as is the help from professionals Logan Vahlstrom and Jose Ruelas for assistance on machining and welding, respectfully. Their pictures are commemorated in the Appendix. We would also like to thank Peter Joshua (Solarius president) and Solarius for sponsoring our team by providing the funds and the training necessary for the quality completion of the half-scale prototype. We could not have accomplished our goals without them.

SOLARIUS™

SPARTAN SUPERWAY

Table of Contents

SPARTAN SUPERWAY

Table of Figures

SPARTAN SUPERWAY

Table of Tables

SPARTAN SUPERWAY

Executive Summary

Introduction and Objectives

Dr. Furman and Engineer Ron Swenson founded the Spartan Superway, which is a suspended transportation system that will deliver pedestrians from one location to another without interference and delays. The objective for the 2018-2019 half-scale bogie design and controls team was to improve the design of the mechanical and mechatronic components such that the bogie would be able to traverse the entire track both manually and autonomously. The track is 18 feet in length and 7 ft tall. The half-scale group was divided into two major sub-teams: one on the mechanical design and the other on mechatronic controls.

Procedure and Results

This project is a continuation from the 2016-2017 semesters, so our team first needed to analyze the status of the project and decide what needed to be done. The mechanical design and mechatronics both could be improved. The steering components were upgraded from plastic gears and the rubber belts held together by zip ties were swapped with chains and sprockets, maintaining a gear ratio of 2.8. Various shafts were changed to carbon steel and machined to create a tighter fit for the powertrain and steering systems. This resulted in yield strengths of 60,000 psi for the steering shafts and 75,000 psi for the drive shaft. Gearbox covers, housings for the limit switches, ultrasonic sensors, and hall effect sensors were designed and 3D printed to protect the gearbox from debris and help with mounting the sensors onto the bogie.

An Arduino microcontroller was used for the mechatronics because of its simplicity and versatility. The code was created in phases. First, individual codes were written to test the functionality of the drive motors, stepper motors, and LCD shield. These codes were then integrated into one. Next, codes were written to test the functionality of the hall effect sensors, ultrasonic sensors, and limit switches. Once tested, all of the codes were integrated into one master code. A control box was then assembled to contain all of the necessary controls systems such that they could be suspended from the bogie with minimal clutter. A fan was implemented to keep the components from overheating and burning out. Excess features such as USB ports for updating the Arduino code, manual and autonomous mode switches, and emergency switch were placed on the outside of the box for ease of access.

Conclusion and Recommendations

Overall, the 2018-2019 half scale bogie and controls team was successful. The bogie functioned according to the specifications. It was able to move forward, backward, and the steering arms were all functioning properly with the code. The main issues came from the motor drivers overheating, presumably because they were drawing too much current. The mechanical design changes improved the stability of the bogie. It is recommended that future half scale teams focus on improving the LCD code, implementing wireless communication between the two bogies, and adding a wireless power source. Manufacturing a better fitting drive shaft for the second bogie will eliminate stresses and using the separate portion of the track slope to test reliability of the wheels can be done.

Introduction

The goal of the Spartan Superway is to create an autonomously controlled, elevated transportation system that is run completely on solar power. The concept picture is shown below in ***Figure 1***. An overhead guideway design includes many benefits such as congestion relief and decreased transportation costs. This new and innovative concept of public transportation was founded by Dr. Burford Furman and Engineer Ron Swenson. First introduced in 2012, the International Institute of Sustainable Transportation (INIST) was created for the research and creation of this elevated transit network (Inist, 2017). Since then, students of San Jose State University's College of Engineering and supporters worldwide have been developing the Superway and creating small-scale, half-scale, and full-scale prototypes for experimentation on different design approaches. The focus is on sustainability and social wellbeing.

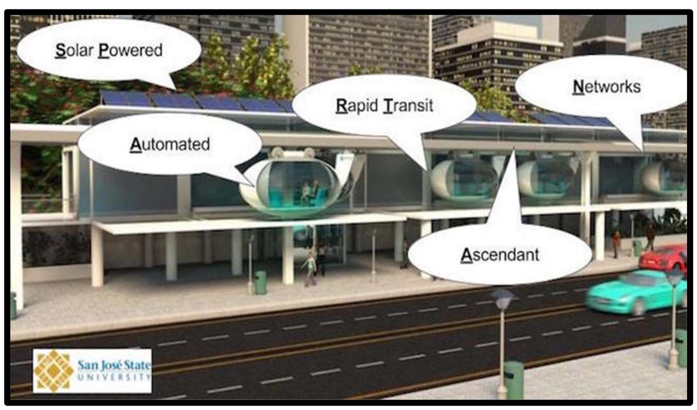


Figure 1: *Spartan Superway Concept Drawing.*

With the growing population in major cities, more traffic leads to collisions and accidents. San Francisco, a small 46.87 square mile city has one of the largest population density in the United States (U.S. Census Bureau, n.d). The population size in this major city has grown 9.8% from 2010 to 2017 and is currently at 884,363 people. An increase in population correlates with an increase in vehicle related accidents. As recorded by the City of San Francisco's Transportation Services, about 30 casualties and 200 serious injuries occur every year. Similarly, the national rate of 2.27 million collision or vehicle related deaths in 2010 has increase to 2.78 million in the 2017 recordings (NHTSA, n.d). Taking advantage of public transportation will likely decrease issues involved with congestion as well as vehicle pollution.

Cars running on fossil fuels produce many harmful emissions such as carbon dioxide, carbon monoxide, methane, hydrocarbons, and pollutants (UMTRI, 2008) that can affect our health and the health of our ecosystem. Many public transit systems also run on petroleum-based fuel and non-renewable energy sources. Today, there is a greater need and awareness for clean energy use. Around the world, people are developing new technology needed for the acquisition and utilization of more reliable forms of energy. The Superway will use photovoltaic solar panels to capture energy and distribute it throughout sections of the track. The power from the solar array will be enough to operate the entire network and produce zero emissions to the environment. Use of public transit will reduce the quantity of cars on the streets and reduce the amount of emissions from vehicles. Although there are many benefits in the use of public transportation, it is not widely popular in America.

Current public transportation is not as popular as it can be compared to that of other countries. The national percentage for those who drive in America is 76.04% (U.S. Census, 2011) while the percent that take public transportation is 5% compared to 35% and 64% in London, respectively (Bendix, 2015). America is a vast country with many spread out areas and public transportation is seen to work best in urban, dense environments. Transits in cities make for a more economic development and can benefit more people than those in the spaced out suburban and rural areas. San Francisco's system is one of the more widely used networks in the bay area and about 25% of the city population utilize the buses (SFMTA). Still, limitations prove why private vehicles are more preferred. Most importantly, it is the inconvenience and low maintenance of the current systems that turn people away from forms of public transportation. The new and modern Superway system will increase the popularity of transit use. Having the ATN (automated transit network) elevated will decrease the traveling time to get from one location to another and allow for more street use to the public.

Current buses and trains have minimal number of stations and in many instances, people have to find other modes of transportation to get from a station to their destination a fair distance away. This is where an elevated guideway can provide a solution and eliminate the need for cars. The Spartan Superway is designed to minimize travel times by providing no interference between retrieval to destination. The pod car carrying vehicles will carry up to six or more people depending on the needs of the location. This capability of transporting individuals and small groups of people at a quick speed makes the Superway a personal rapid transit (PRT) system. The carriage will also be designed for easy access by the elderly and disabled and an application software gives the user easy access to their ride. With a simple click, the closest pod car is on the way to a nearby station for pick up. With the autonomous feature, the podcars do not need to be manually operated, and will be available at any time of the day. The elevated concept utilizes the air space about 15 ft off the ground. This helps with the quick travel time and still allows for traffic on the streets to pass undisturbed. An elevated concept that is able to improve transit usage and lessen the amount of vehicles on the streets will also make room for recreational spaces and buildings that are taken over by cars. The Spartan Superway will not only appeal to the public, but to urban developers as well.

Installing new transit systems are costly. The Bart subway extension that will run from Pittsburgh to Antioch in the bay area has an estimated cost of $525 million per mile (Bart, n.d). The 4.5 mile Boston Green Line extension will cost an estimate of $489 million per mile (Dunca, 2017). The infrastructure for an elevated system is simply lighter and will cost less material and labor to install. JPods, a similar elevated ATN concept that has estimated their system to cost $10 million per mile (JPods, n.d). Although the concept is new and not yet approved by city governments, the economics for an elevated systems is a lot cheaper than that of any other system. That should be able to entice the production of Spartan Superway and similar ATNs in new urban developments.

With a more modern system, public transportation is likely to increase in popularity. The greater use of public transits will decrease issues involved with pollution and congestion and benefit society as a whole. San Jose State University has been involved with the Spartan Superway for the past seven years and has made a large contribution in testing elements of the system. Aside from the mechanical engineering department, help has been provided by other departments around the school. The College of Business and Urban Planning got involved early in 2012 when the project first started. Students from the Industrial Design explored with the aesthetics of the pods and stations in 2013 and the first full scale pod and guideway was made. In 2014, students from other countries worked on the project and made much contributions to the suspensions, controls, and switching mechanism on the prototypes. In 2015, performance of the bogies on various degrees of slope were tested and the location of potential stations were researched. In 2016, the half-scale model was made to run autonomously, the user interface app was developed for the 1/12 small scale prototype. In 2017, wayside power was explored and integrated into the half-scale model. This year, the half-scale bogie and controls team was able to fix automation flaws in the previous designs, write code for the sensor integration, and build an organized control box that can be easily utilized for future work.

This paper continues to describe the half-scale team's role in the end goal of the Superway project, the team's focus for this 2018-2019 school year, the design requirements for the system, exploration in similar developments of ATNs, the design and testing completed, budget of the project, and finally, suggestions for future endeavors.

Background and Context for the Work of your Sub-team

Introduced in 2012, Spartan Superway has won accolades since the day of its inception. The first Spartan Superway team (2012-2013) created an award-winning 1/12th-scale model that provided the blueprint for all subsequent 1/12th-scale teams. The following academic year, students created a full scale track, bogie, and pod-car that is still used for demonstrations. The 2014-2015 team made significant progress with the construction of a long full-scale track, improvements to the bogie, and the addition of the Beamways switching concept (and its associated controls system).

The half scale team of the Spartan Superway first came into being in the academic 2016- 2017 year with an objective of demonstrating a proof concept for the full scale model, but with less resources and faster prototyping capabilities. During this year, students were able to make leaps and bounds of progress as the half scale team. With eleven students working on various diverse half scale teams such as active suspension, wayside power, braking and propulsion, bogie steering and failsafes, and mechatronics and integration, this team laid a solid foundation for future half scale teams. Teams during the summer of 2017 and 2017-2018 academic year were able to make some progress on the wayside power and bogie suspension, respectively. But since there were only two to three students working on this team, a lot of major work was left on the mechatronics side. The mechanical design aspect of the model also required necessary modifications for the bogie to function properly and desired by the user.

Problems faced by the half scale team

The result of the previous half-scale team was disorganized and undocumented, leaving much to be questioned. All the issues can be divided into broadly two categories: mechatronics issues and mechanical design issues.

1. *Mechatronics issues:* The main issues our team faced on the mechatronics side are as follows:

    a. *Jumbled wiring and solder overflow:* The 2016-2017 team did a great job putting together the mechatronics of the bogie, but over the years, the solder and the wiring started weakening, leading to bad connections and short circuits. ***Figure 2*** shows the control assembly from the 2016-2017 half scale team and ***Figure 3*** shows the state in which we received it. The mechatronics team needed to find a solution to this problem for reliable functionality of the control assembly.

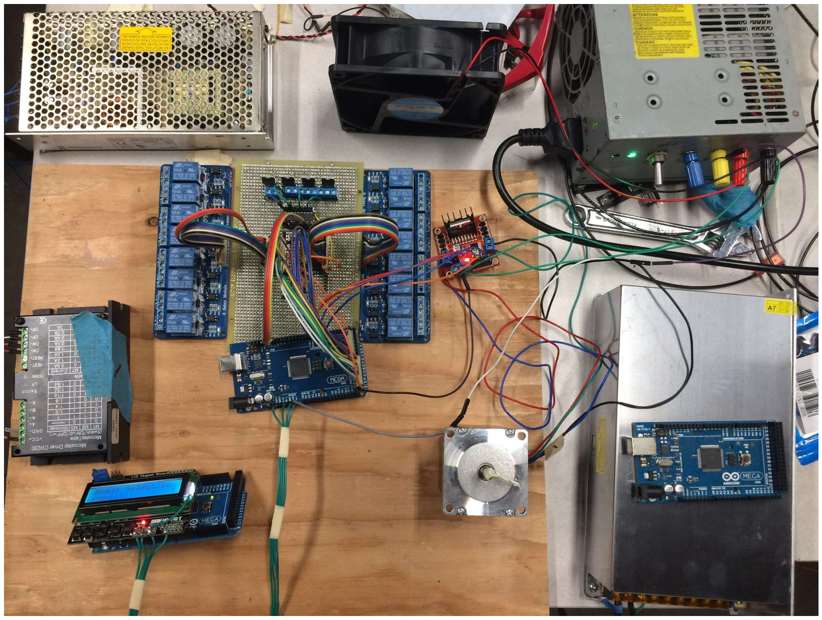


Figure 2: *2016-2017 half scale controls assembly*

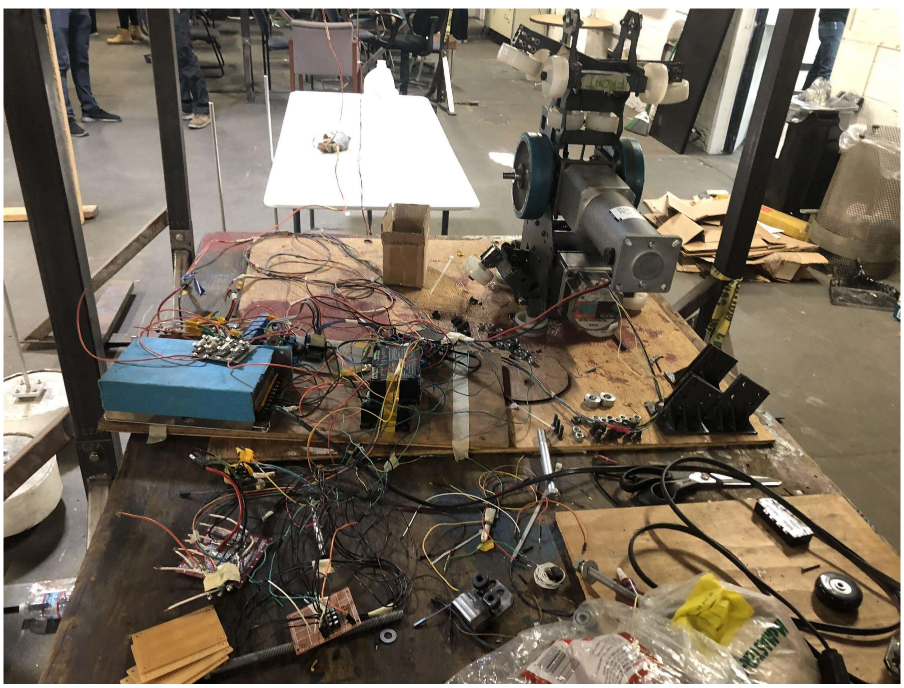


Figure 3*: 2016-2017 half scale controls assembly*

b. *Powertrain:* The powertrain section of the mechatronics design include the circuitry related to the brushed DC motor which drive the bogie forward, the power source related to it and the motor driver component. The previous year's team's design building their own motor driver by soldering components. The motor driver built in the past did not prove quite as reliable. Thus, this year's team tried to solve that issue by buying a motor driver off the shelf.

c. *Steering:* The steering and track switching mechanism also faced certain issues. Past year's reports especially 2016-2017 half scale report noted some success in steering the mechanism but this year's team planned to improve on their work and make it fail proof. The steering mechanism is essential for the bogie to traverse a complete loop around the track. Thus, it is an essential part that needs work when our ultimate goal is to create an autonomous bogie that can travel a full loop.
d. *Sensors:* Sensor integration with the bogie hardware is equally as important as the bogie itself. Without sensor's it would be impossible for the bogie to function in an autonomous fashion, which is the ultimate goal of the Spartan Superway project. Previous year's teams had integrated hall effect sensors in their bogie which seemed to work perfectly to gather the velocity data of the bogie. This year's team felt the need to inculcate more sensors such as ultrasonic sensors on the front and back of the bogie and limit switches underneath the steering arms. Ultrasonic sensors are supposed to act like the eyes of the bogie, detecting the distance of the bogie to the other objects in the path of the bogie and preventing it from falling of the track. On the other hand, limit switches are supposed to stop the steering arms from over-steering and prevent damage to the stepper motor.

2. *Mechanical Design issues:*
    a. *Lower steering arm:* While the updated design did improve the strength of the shaft, it caused the lower steering arm to slip. Slipping of this kind should be prevented to make the steering system more stable and reliable. Our team came up with the ingenious idea of machining a flat D shaft to counter any slipping.
    b. *Steering gear and chain:* The steering system involved the use of a plastic pulley and zip-tied plastic chain. Since, the stepper motor has high torque, a sudden jerk of the stepper motor could have easily damaged or broken this chain. It was extremely important to place tension on the chain and use a sturdy metal gear and chain system to eliminate possibilities of failure.
    c. *Drilled sensor fixtures:* Previous year's teams put limit switches and hall effect sensors on the bogie but never secured them effectively. As a result, over the years the sensors fell off the bogie and were hard to put together due to the complexity of the wiring. This year's team wanted to counter such issues by 3D- printing the housings for various sensors and drilling them onto the bogie as a permanent

addition to the bogie design.

d. *Custom-made Gear-box:* The half scale team also wanted to rule out the possibility of debris getting stuck into the gears. Therefore, a custom designed 3D-printed gearbox, which has storage for future components, was fabricated.

Description of the Sub-team and Objectives

The objective for the 2018-2019 half-scale bogie and controls team is to improve the mechanical design and mechatronics components on the bogie while keeping the original specifications implemented by previous teams in the past. Improving these aspect will guarantee for the bogie to be able to transverse the entire track. Some of the mechanical design components were changed and improved upon that will ensure the bogie will completing a loop around the track. Originally, our goal was for the bogie to transverse the entire track within five minutes. However, things did not go according to plan and the original Spartan Superway shop on St. John St. was lost. This resulted disassembling the track into pieces. Luckily, there was a portion of the track that could demonstrate the half-scale bogie capabilities without further machining. On this shortened track, the bogie is able to travel back and forth, switch tracks, and handle the 600lb estimated additional weight of a pod car with 6 passengers. The new section of the track has dimensions of being 18 feet long and 7 feet tall (***Figure 4)***. On the mechatronics, several sensors were utilized with an Arduino master code to guide the bogie and make appropriate steering motions, allowing for a smooth travel in both directions along the track.

Figure 4: *The location of the half-scale track and current size of the track.*

The half-scale bogie and controls team is a valuable example of what a mechanical engineer is expected to know in the aspect of both mechanical design and mechatronics. In the mechanical design, the half-scale consist of having the mainframe, steering arms, gears, and a chain. The mechatronics system consists of various sensors and motors controlled by a microcontroller, Arduino and independent motor drivers. The sensors include hall effects, limit switches, and ultrasonic. Furthermore, it uses a DC motor for the drivetrain and a stepper motor to control the steering mechanism of the bogie. All of these controls systems were put into a controls box shown in ***Figure 5***. Integrating all these parts and components ultimately makes the half-scale bogie and controls for 2018-2019.

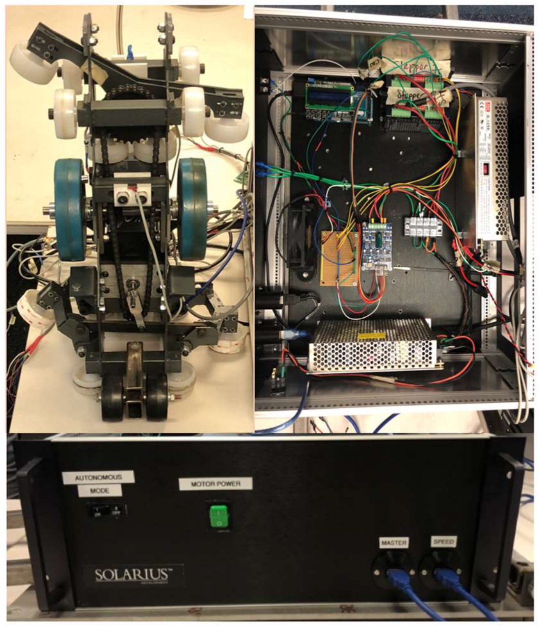


Figure 5: *2018-2019 half-scale bogie and control box.*

Design Requirements and Specifications for the Sub-team's Work Products

The 2018-2019 half-scale bogie team continued to improve on the mechanical design and mechatronics aspects of the bogie initiated and handed down to the current team by the previous year's half-scale teams. Initially, the 2018-2019 half scale bogie team intended to achieve certain design specifications which changed over time due to several factor. One major factor that encouraged these modifications in design specifications was the change of venue of the Spartan Superway Design Center (SSDC). Due to different space constraints at the new location, the team could not incorporate some key design specifications in the track such as a 17° slope incline and decline section and also reduce the length of the track. In addition, the team lost valuable time in moving and resizing the track for it to fit in the new Spartan Superway Design Center.

In the beginning of the semester, the team spent time on modifying the design specifications to have concrete but not immensely complex half scale bogie model by the end of the semester.

Several design and mechatronics systems have been integrated with the bogie to achieve all the current design specifications. The bogie has traverse the whole length of the track autonomously and manually at a safe speed of 6 inches/s which is half a foot per second. At this speed the current wheels, with a diameter of 3 inches, would rotate at 17.63 rpm which is another design requirement the team kept in perspective. Being the half scale model, the half scale team decided that it the weight carrying capacity of the bogie should meet half of what full scale model aims for. Thus, the bogie should also be able to support and carry 600 pound weight placed at it base underneath the bogie. In addition, since the length of the track was reduced to one-third of the original track, the half scale team set a goal to achieve the above mentioned tasks in less 2 minutes. The team came up with this number by performing calculations shown in ***Figure 60 in Appendix D*** and incorporating a factor of "one-third". The team's design also incorporates reducing the weight of the bogie and a redesigned midshaft to reduce torsion and ensure a better and safer drive.

The half scale team also laid out major design requirements in the beginning of the semester. To achieve the fully autonomous and manual track traversal it is really important for the bogie to have a fully reliable mechatronics system for both the powertrain and steering controls. The bogie's track switching mechanism was also in dire need of replacement as it was held in place by zip ties. Replacing the plastic pulley and chain system with a metal gear and chain system which could withstand the torque provided by the stepper motor would be the ideal way to improve the steering system. Another design requirement which was essential to the success of the project was having a centralized gearbox with ability to store components and protecting them from rust and debris. Last but not least, better design of shafts was also necessary for the smooth functioning of the bogie. With collective agreement on the modified design specifications and design requirements, the half-scale team for the current year proceeded to achieve these monumental tasks over the course of the semester.

State of the Art/Literature Review

The concept of an elevated ATN is not entirely new. Companies such as SkyTran, JPods, Futran and much more are also developing this concept similar to that of the Spartan Superway (University of Washington, 2018). SkyTran is one example of a recently developed suspended ATN with an objective "to solve the transport crisis by creating a transport option that is high-speed, scalable, and low-cost" (SkyTran, n.d.). They are current partnered with NASA and are working together to test the project. On their website, it shows that SkyTran plans to place a support poles every 50m and reach cruising speeds of 100mph. ***Figure 6*** shows a full scale prototype of their guideway and pod. The difference in the SkyTran suspension and propulsion system is that they are reliant on a magnetic levitation instead of a mechanized bogie one like the Superway. JPods started in 1998 and built a 20 ft working prototype that was displayed to the public around the United States and Canada, in 2006 (JPods, n.d.). JPods projects that their system will diminish the amount of energy needed to power a transit system by 90%. Their product has made major progress throughout the years and in 2018, the company signed a contract with the City of Shaxin in China to build the "world's first solar-powered mobility network" (JPods: Shaxin [...], n.d). An early prototype of their pod is featured in ***Figure*** *7*. Below. Futran is another leading competitor in suspended ATNs and are the first suspended guideway system that have commercialized their technology (Milotek, n.d.). Since the testing done in South Africa, the company has been commissioned twice to resume testing in different locations (Futran, 2018). The Futran system aims to carry both passengers and goods from one location to another. As shown in ***Figure 8***, this systems will help with mining and good transportation for areas outside of the cities and in other types of environments. Like the Spartan Superway, all these other networks utilize solar collectors into their design concept and we are inspired by the progress of similar projects around the world. These were the main examples that informed our work and design concepts.

Figure 6: *SkyTran Testing Facility in Mexico.*

Figure *7:* *JPods prototype.*

Figure 8: *2015 Futran System in South Africa.*

Description of Our Design

Mechanical design

The prime design of the 2018-2019 half-scale bogie focused on maintaining most of the original design specifications since the bogie already has great features and there was no need to recreate a new design for the bogie. Therefore, only improvements were made on what was causing problems or what needed the most attention. One of the bogie refinements consisted of a new selection of carbon steel metal gears and a roller chain to fix the existing unsustainable design the bogie had with the belt and zip ties. This is one of the first changes that were necessary on the bogie since the current design of bogie consisted of having a rubber belt and plastic pulleys. The belt was being held by a zip tie as shown in ***Figure 9***. It is assumed the previous team put a zip tie on the belt to ensure tension and for the belt to grab on the plastic pulleys. The problem with design and setup is with over time with the rapid response from the stepper motor and the steering mechanism, the plastic will get worn out and fail. Also, the zip ties will eventually give up due to the bogie constantly doing turns and switches on the track. Zip ties tend to be used for a quick fix, not for significant features and reliability.

Figure 9: *The old setup with a zip tie.*

Implementing the new gears and chain made dramatic improvement to ensure when the bogie needs to make a turn or switch on the track, it will be able to handle all the torque being giving of from the stepper motor. Therefore, this will reduce the chances of the steering mechanism to fail when the bogie is operational. Furthermore, there is a significant amount of advantages of having a roller chain compared to a belt. Roller chains are stronger, last longer, and can handle more torque. Also, having the roller chain on the steering mechanism will ensure on not having slippage on the system. A picture shown in ***Figure 10*** displays how the roller chain and carbon steel gears look on the bogie once they were mounted on. In regards to the gear ratio, the previous team designed specifications chose the gear ratio to effectively output the required torque while maintaining a reasonable turning and switching speed on the track. Therefore, the new designed metal gears gear ratio was chosen to fit the closest to the existing gear ratio. The new gear ratio is 2.78 compared to the old gear ratio being 2.8. The calculation for the new gear ratio can be seen in the ***Appendix D.***

SPARTAN SUPERWAY

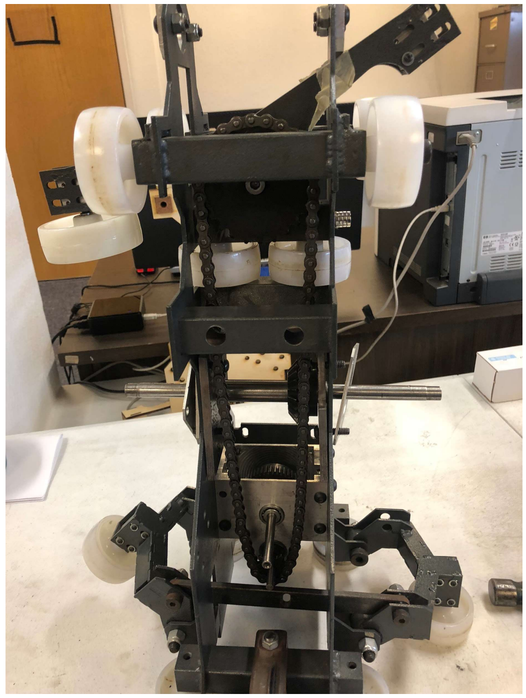


Figure 10: *The new gears and roller chain mounted on the bogie.*

Next, since the new carbon steel gear for the bottom steering arm assembly had a bigger diameter then the old shaft that was connected to the stepper motor, it needed to be changed. The new selected shaft has a diameter of 3/8 inches and being 4.2 inches long. Also, choosing this diameter for the shaft it had to accommodate the current metal gear inside the gearbox which there was no intention of changing. ***Figure 11***, shows the configuration of the metal gear inside the gearbox and the new carbon steel gear. The old shaft connected to the stepper motor had a step diameter from 3/8 to 3/16 inches which can be seen in ***Figure 12***, with the metal gear that goes inside the gearbox. The reason why this shaft needed to be changed was due to the smaller section diameter having a greater chance of bending or eventually breaking from the all the torque the stepper motor delivered to the shaft. Furthermore, the bottom shaft that controls the lower control arm was replaced with a large diameter shaft to accommodate the new carbon steel gears. ***Figure 13***, shows a picture of the new manufactured shafts for the new selected gears.

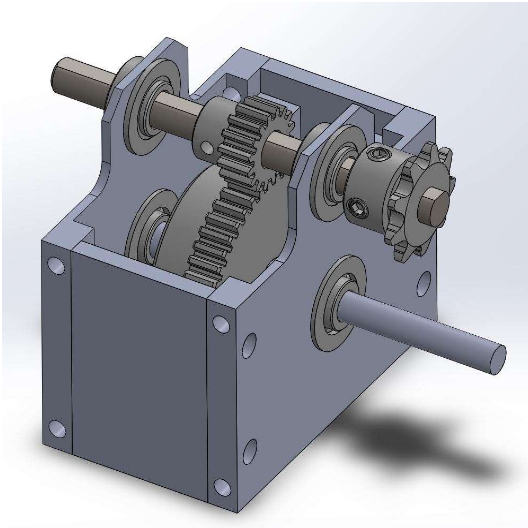


Figure 11: *A CAD picture of the gearbox with the gears configuration.*

Figure 12: *The old upper shaft with existing gear and bearings.*

Figure 13: *The new shafts manufactured.*

Overall, the mechanical design for the 2018-2019 half-scale bogie consists of a stepper motor that controls the steering and a DC brushed motor to power the drivetrain of the bogie. There is a total of four gears, three shafts and a 28-inch long chain. A gearbox is implemented to enclosed two gears in place with bearing to drive the other two gears that have a chain. These two gears which have a chain are located at the bottom and top of the steering arms of the bogie. The setup of the stepper motor can be seen in the CAD model in ***Figure 14***.

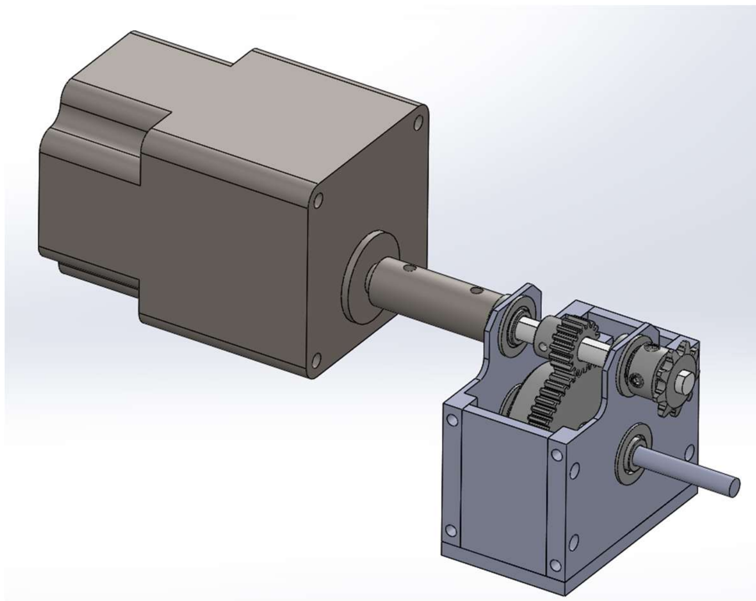


Figure 14: *SolidWorks assembly of the gears with the stepper motor shaft.*

The stepper motor has a holding torque of 106 lb-in and a speed of 50 RPM, which is efficient enough for the purpose of our bogie to make turns and switches on the track. The datasheet can be seen in the Appendix. These specifications of the stepper motor will provide enough torque and speed to the bogie's steering mechanism to accurately be able to make turns and switch between rails within an adequate time. The stepper motor controls two steering arms, an upper and lower control arm as shown in ***Figure 18***. These two controls arms are operated by gears and a chain mechanism that connected them together and makes the upper and lower control arms synchronized. The stepper motor is connected to a coupler that connects to a 3/8 in diameter shaft that holds two gears in place. One gear is placed in the middle of the shaft that connects to a bigger gear which leads to a shaft that is connected to linkage and drives the upper control arm. The output gear has 48 teeth and the input gear has 16 teeth which result in having a gear ratio of 3. Performing a few calculations with this gear ratio, the speed of upper control arms moves at 16.6 RPM and having a torque of 318 lb-in. All the calculations can be seen in ***Appendix D***. In the CAD shown in ***Figure 15*** shows the two gears that inside the gearbox.

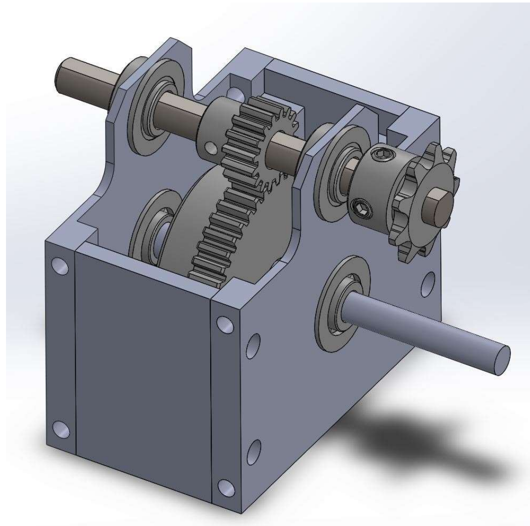


Figure 15: *The two gears inside the gearbox.*

The third gear is located outside of the gearbox and at the end of the shaft as shown in ***Figure 16***. This gear has a chain that will connect to the four gear which is located the bottom as shown in ***Figure 17*** The fourth gear functionality is to steer the lower control arm. When the stepper motor is given a demand to turn left or right all these gears are synchronized together to operate the steering arms at the same time. The gear at the top is the input and has 9 teeth and the bottom gear is the output which results in having a 2.78 gear ratio. Using this gear ratio and applying the same equations as for the previous gears, it gives the lower control arm a speed of 18 RPM with a torque of 294.44 lb-in. ***Figure 17*** displays the whole CAD model of the bogie with the two gears attached to the chain. Also, this CAD image is able to represent how the lower and upper control arms are synchronize.

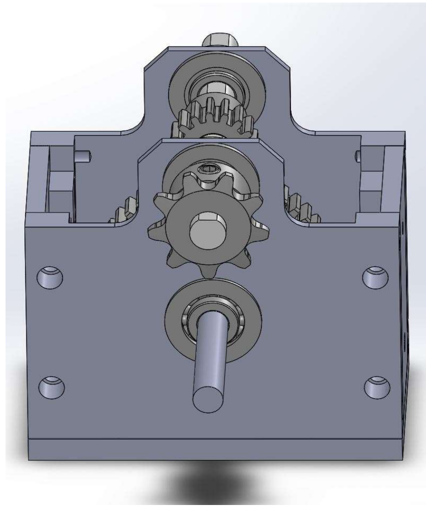


Figure 16: *A CAD picture displaying the location of the third gear.*

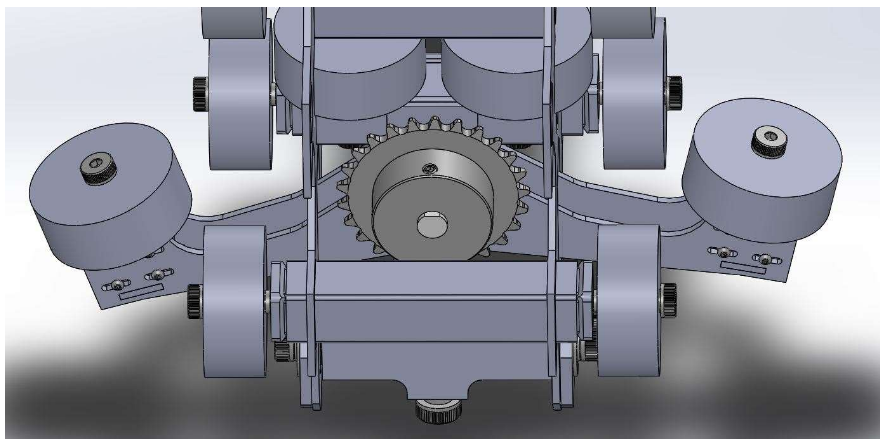


Figure 17: *A CAD picture displaying the location of the fourth gear.*

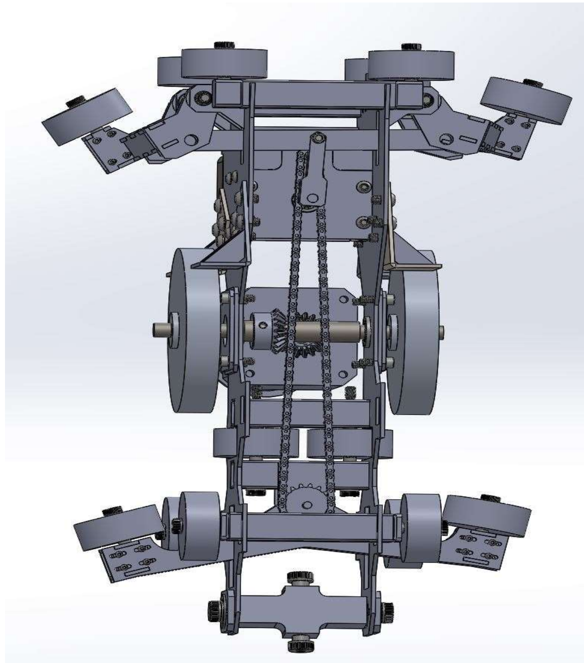


Figure 18: *The whole assembly of the bogie which shows the bottom and upper steering arms.*

For the DC brushed motor, it is a 24V motor and having a gear ratio 160:1 which produce speed of 2500 RPM and torque output of 677.5 lb-in. The datasheet can be seen in ***Appendix C***. There are two miter gears and a 10-inch long shaft which has two 6.5 inch rubber wheels attached to it. Using ***Equation 1***, we were able to derive what speed to expect when the bogie is traveling along the track. The speed was calculated to be 6 in/s and the calculations can be seen in the Appendix. Therefore, when the hall effect detects the magnet going around the wheel we expect to see a speed around 5.31 in/s on the serial monitor or the LCD screen. Furthermore, both miter gears have 20 teeth which result in having a 1:1 gear ratio. This will result in the output power to be delivered equally throughout the shaft. The setup of the powertrain can be seen in the CAD model in ***Figure 19***.

$$\text{Velocity} = 17.6\,\frac{rev}{min}\cdot\frac{1\,min}{60\,s}\cdot\frac{2\pi\,rad}{1\,rev}\cdot 3.25\,in \qquad (1)$$

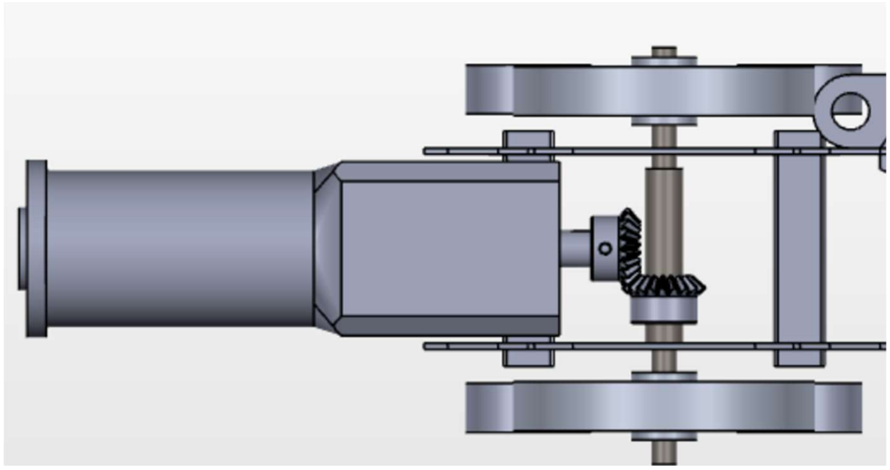


Figure 19: *A CAD picture displaying the setup of the powertrain.*

Lastly, several brackets and holders were designed to hold all the sensors that were going to be integrated onto the bogie to ensure the bogie can properly transverse around the track with no issue. All the sensors brackets and holders were 3D printed and there was no need to manufacture these parts, especially since they are just used as place holders for the sensors. The limit switch holders are bolted on under the left and right side of the upper control arm. The hall effect sensor holder is bolted onto the frame near the wheel where a magnet is going to be placed. The ultrasonic bracket is going to slide onto front and rear bar frame of the bogie. ***Figure 20***, shows all the 3D printed holders and brackets with the all sensors which will go on the bogie.
Figure 21, represents all the locations of where all the sensors are going to be placed in.

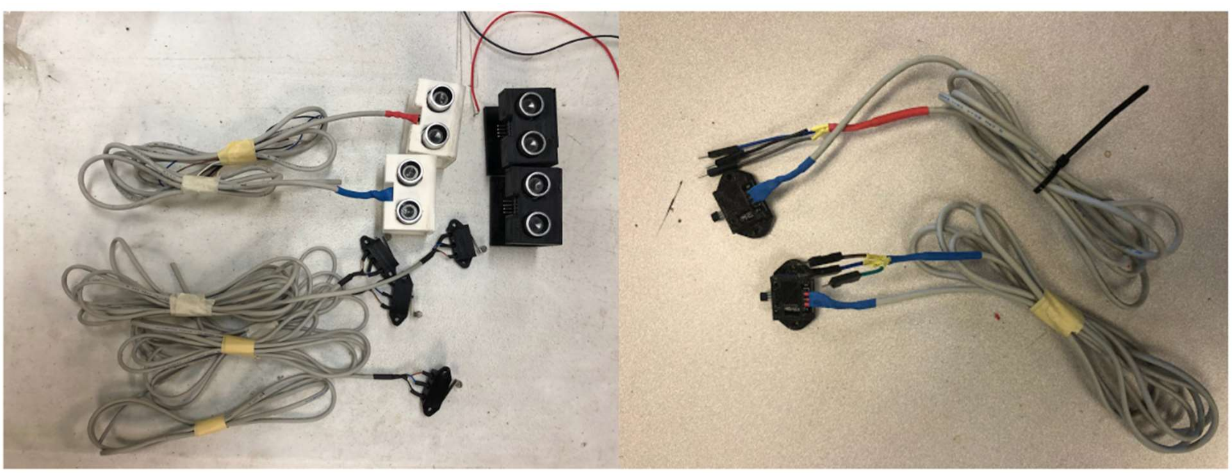


Figure 20: *A picture of all the sensors solder to wires and placed in the 3D holders.*

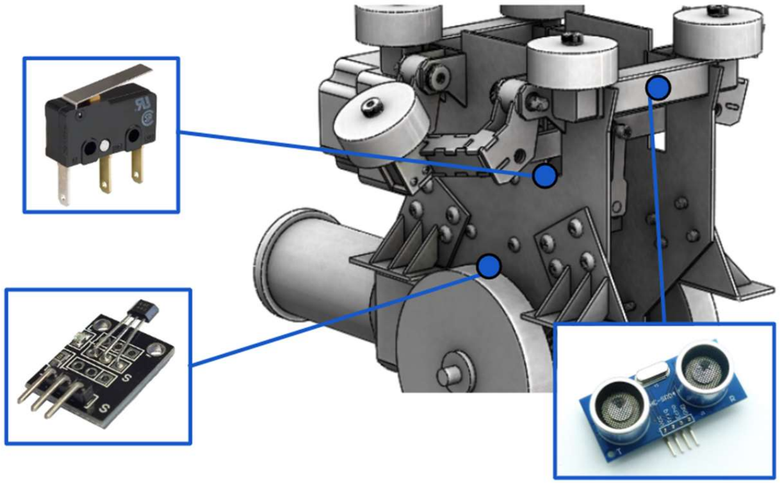


Figure 21: *A picture representation of where the sensors will be placed on the bogie.*

Mechatronics

The Half Scale Bogie circuit is composed of several sensors and motors. For ideal and simplified display reasons, the circuit schematic is displayed with a simple block diagram in ***Figure 22***. The schematic displays the components used in the half-scale bogie control design. All the sensors and electronic hardware being used are represented in the block-schematic.

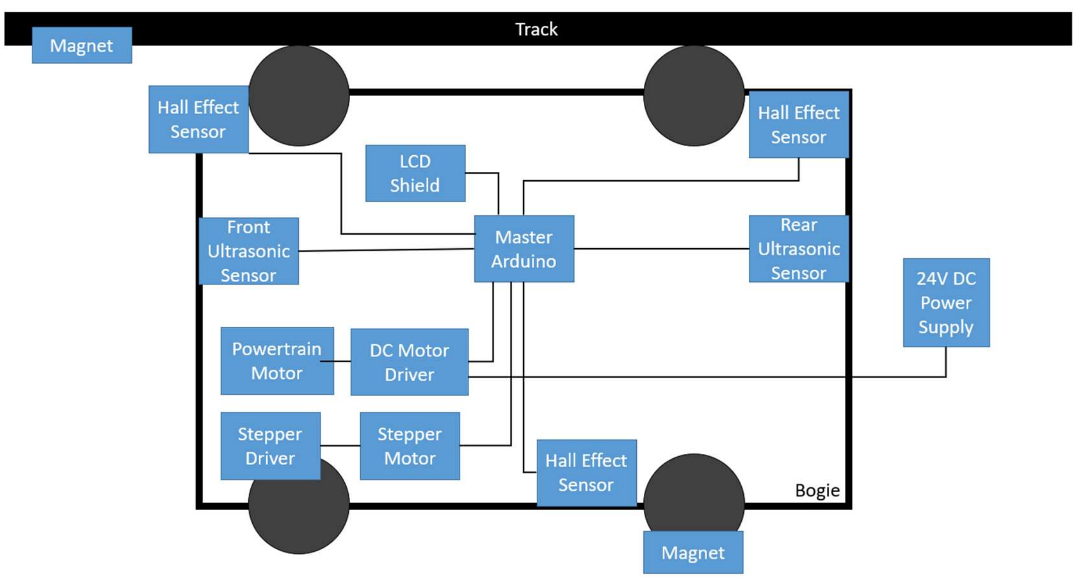


Figure 22: *The simplified circuit schematic for the half-scale bogie.*

Before going further in depth about the components, it is important to understand the control algorithms, as each method of operation requires different components to properly function. All these components are situated on the bogie and in the control box, shown in ***Figure 23***. Power supplies keep the motors satisfied with power requirements, along with smaller electronics such as the Arduino and DC CPU fans.

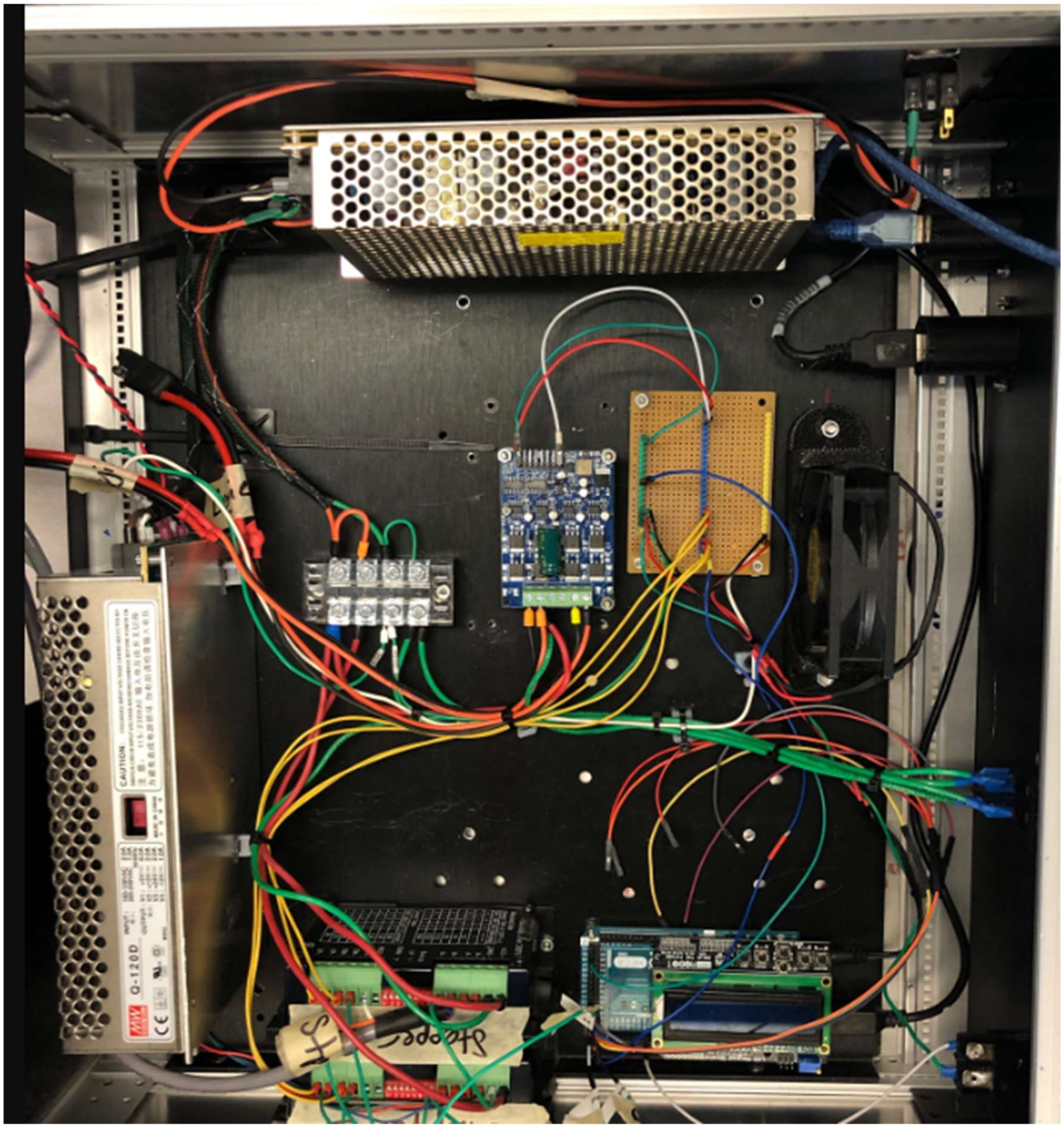


Figure 23: *Top view of the control box with the cover removed.*

Manual Controls

The manual controls of the half-scale bogie offer the most primitive and rapid method of control. Here, the LCD shield is the principal device in commanding the bogie to move or turn. The idea behind this is the need to have a method to manually control the bogie in the event of a sensor or motor failure. If there were no manual controls, any failure in autonomous operation would leave the bogie stranded and cause more trouble for the operator. Additionally, manual controls can be used to precisely test the bogie functions and sensors prior to automation. This will result in a more reliable autonomous system as each individual component has been tested and verified to be operational on the track. Each button on the LCD shield, shown in ***Figure 24***, corresponds to a different function to control the bogie. While the button is held down (pressed) the active functions in the button case will continue to run until internal conditions are met or the button is depressed. The master code, displayed in ***Appendix G***, features all the programming for the manual controls.

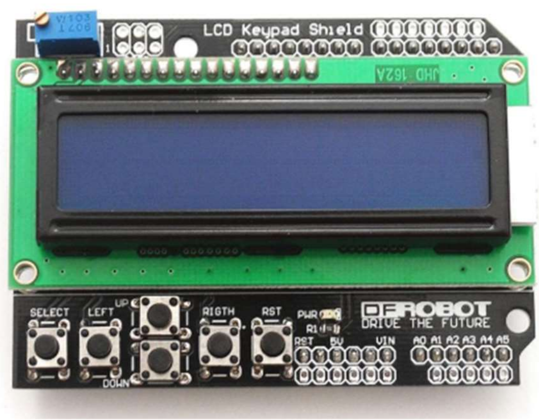


Figure 24: *The LCD shield used in the half-scale controls.*

The up/down buttons operate the powertrain by performing logic HIGH/LOW inversions to the DC motor driver, shown in ***Figure 25***, direction pins (DIR), which in turn, drive the bogie forward or backward. Additionally, an analog value is sent to the PWM pins for each respective motor. This analog value is within the range (0 to 255) where 0 is equivalent to a 0% duty cycle and a 255 is the maximum possible power (100% duty cycle). The logic in these functions is exhibited in ***Figure 26***, where the function "forward()" will command the direction pins to be high and run the bogie in that direction at a set speed on the PWM signal. A similar function is programmed to the down button on the LCD shield that will operate the bogie with a "backwards()" function that reverses the direction of the motor by sending a reverse logic signal to the directional pins. The motor driver controls a 24V DC motor at 6 amps (the output of our power supply), although they are rated for 9.1 amps. The wiring configuration for the motor driver is shown in ***Figure 27***.

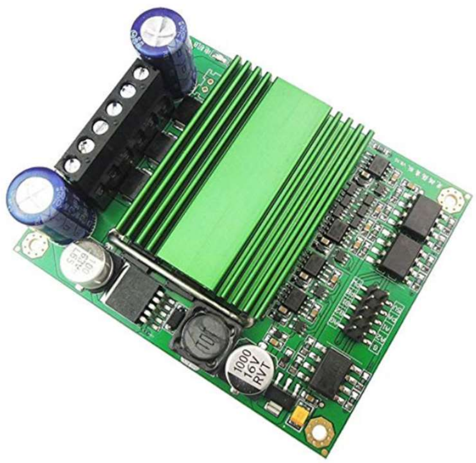


Figure 26: *PWM DC motor driver.*

```
void forward() {
  digitalWrite(dir1, HIGH);
  digitalWrite(pwm1, HIGH);
  analogWrite(pwm1, 200);
  digitalWrite(dir2, HIGH);
  digitalWrite(pwm2, HIGH);
  analogWrite(pwm2, 200);
```

Figure 25: *Powertrain code.*

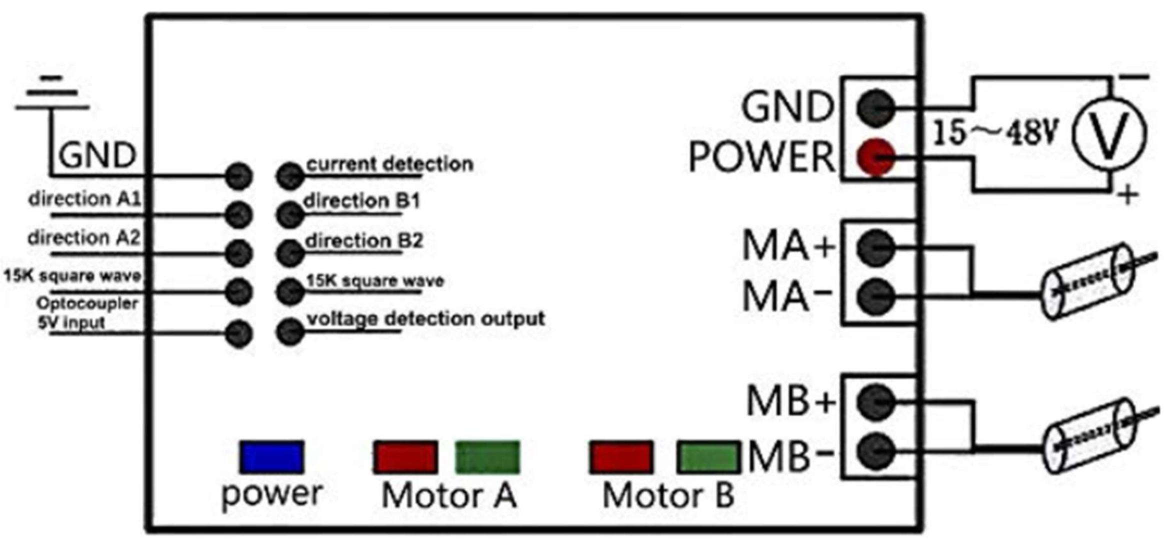


Figure 27: *The motor driver schematic.*

The bogie steering is powered by a high precision stepper motor. The PK296A2A Vexta 2-Phase stepper motor works in conjunction with the CW230 driver, ***Figure 28***, to provide a smooth and precise step control in order to choose between tracks when turning. Dip switches on the driver can be changed to adjust input current and step resolution. ***Table 1*** displays the wiring configuration for the stepper motor to the motor driver. The motor driver pulse and direction signal terminals were properly wired in order to allow control with the Arduino stepper library. The pulse signal positive and direction signal positive, CP+ and CW+ respectively, were connected directly to 5V+ from the Arduino. The negative terminals for each were connected to Arduino pins and used in the stepper function for control.

Table 1: *Stepper motor pinout to a DB9 connector.*

Stepper Motor Wire	Description	DB9 Pin
White	Coil center tap	4
Yellow/Brown	Coil center tap	5
Black	Phase A+	6
Green	Phase A-	7
Red	Phase B+	8
Blue	Phase B-	9

The steps per revolution were determined in ***Equation 2*** using the datasheet specs for step angle. The ideal step size resulted to be approximately 7000 steps in either direction (clockwise or counterclockwise) for a complete turn on the track. The left/right buttons on the LCD shield command the stepper motor to turn in opposite directions with respective turning functions. Furthermore, there are limit switches in place under the steering arms to protect in the case of oversteering. The position of the limit switches is shown in ***Figure 30***. The programming for the stepper motors includes a logical operator that negates any motion in the overturn direction of the stepper motor when a limit switch is actively pressed. ***Figure 29*** displays a turning function from the master code, “leftlimit()”. If the limit switch is depressed, the stepper motor will perform the step at a certain predetermined speed for predefined step size.

$$\textit{Steps per revolution} = \textit{Gear Ratio} \cdot \frac{360°}{\textit{Step Angle}} \quad (2)$$

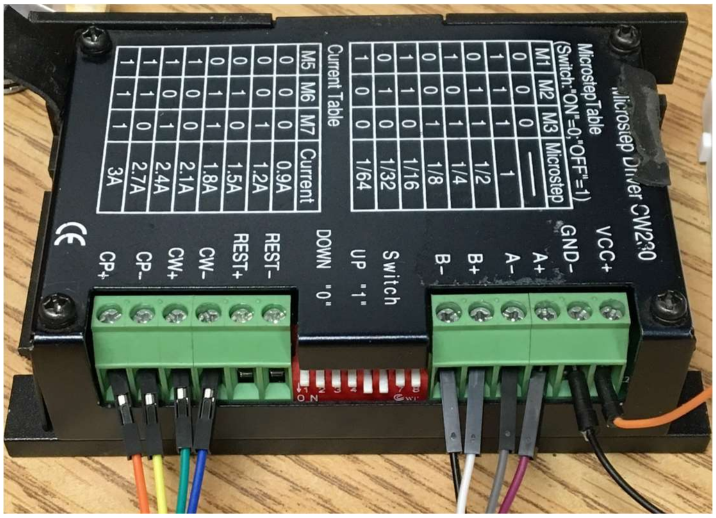


Figure 28: *Microstep stepper motor driver.*

```
void leftlimit() {
  int l = 0;
  l = digitalRead(limitL);
  while (l == LOW) {
    stpr.setSpeed(STOP);
    stpr.step(STOP);
    Serial.println("Steering left limit triggered, stop turning");
    break;
  }
  while (l == HIGH) {
    Serial.print("Turning Left for: \t \t");
    Serial.print(test);
    Serial.println(" steps");
    stpr.setSpeed(fast);
    stpr.step(test);

    // this is to alternate directions for autonomous mode test = ~test;
    break;
  }
}
```

Figure 29: *Stepper motor program to turn left.*

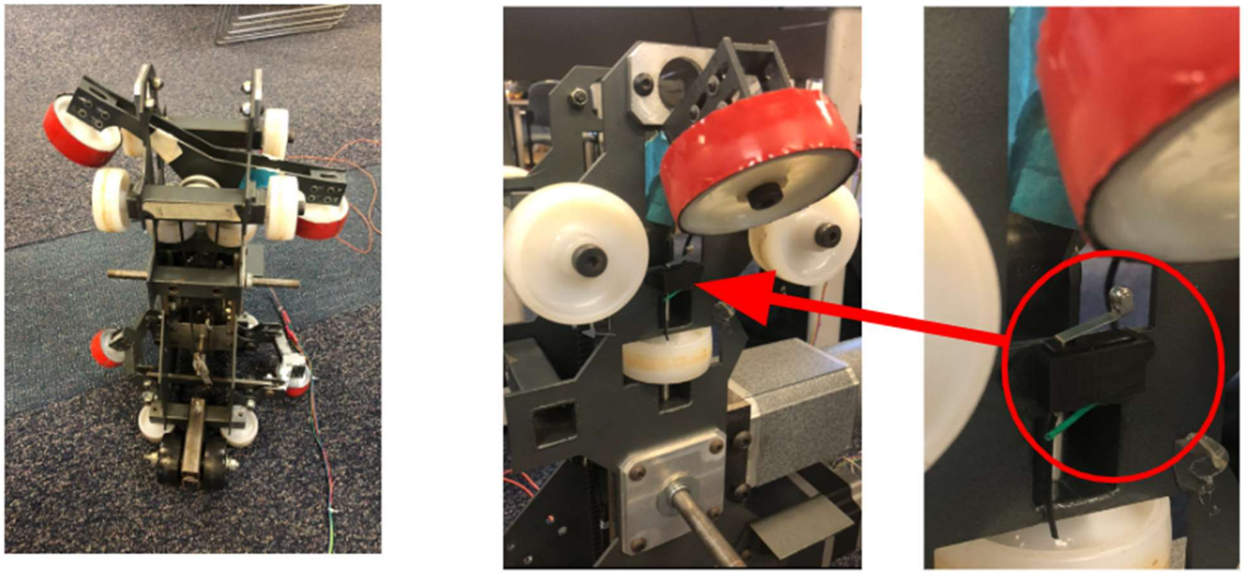


Figure 30: *The placement of the limit switches under the steering arms.*

Autonomous Mode

The second method of operation for the bogie was an autonomous mode. This method of operation integrated the sensors with the motors to create motion and control without an operator. A toggle switch positioned on the front of the control box, shown in ***Figure 31***, activates the autonomous mode. The toggle switch runs a ground wire to an Arduino pin, which is read in the programming to run the autonomous mode when the toggle switch is active to allow the pin to be read HIGH. Also displayed in the figure is a switch to control the power to the motors. This switch runs in between the ground wires for both the stepper motors and the DC motors. In the event of any motor issue, we can quickly shut off the motors and leave the rest of the circuit in operation. Additionally, there is an EMO switch on the side of the control box which runs in line with the line cable from the AC power for the power supplies. Activating this switch will shut down the entire system and is only meant to be used in an emergency. Finally, the two USB ports on the front of the control box are easy access terminals for both Arduinos on the system. Each port is labeled so the operator can understand which serial monitor needed to be read. This can also serve as a rapid method to upload new programming to either microcontroller.

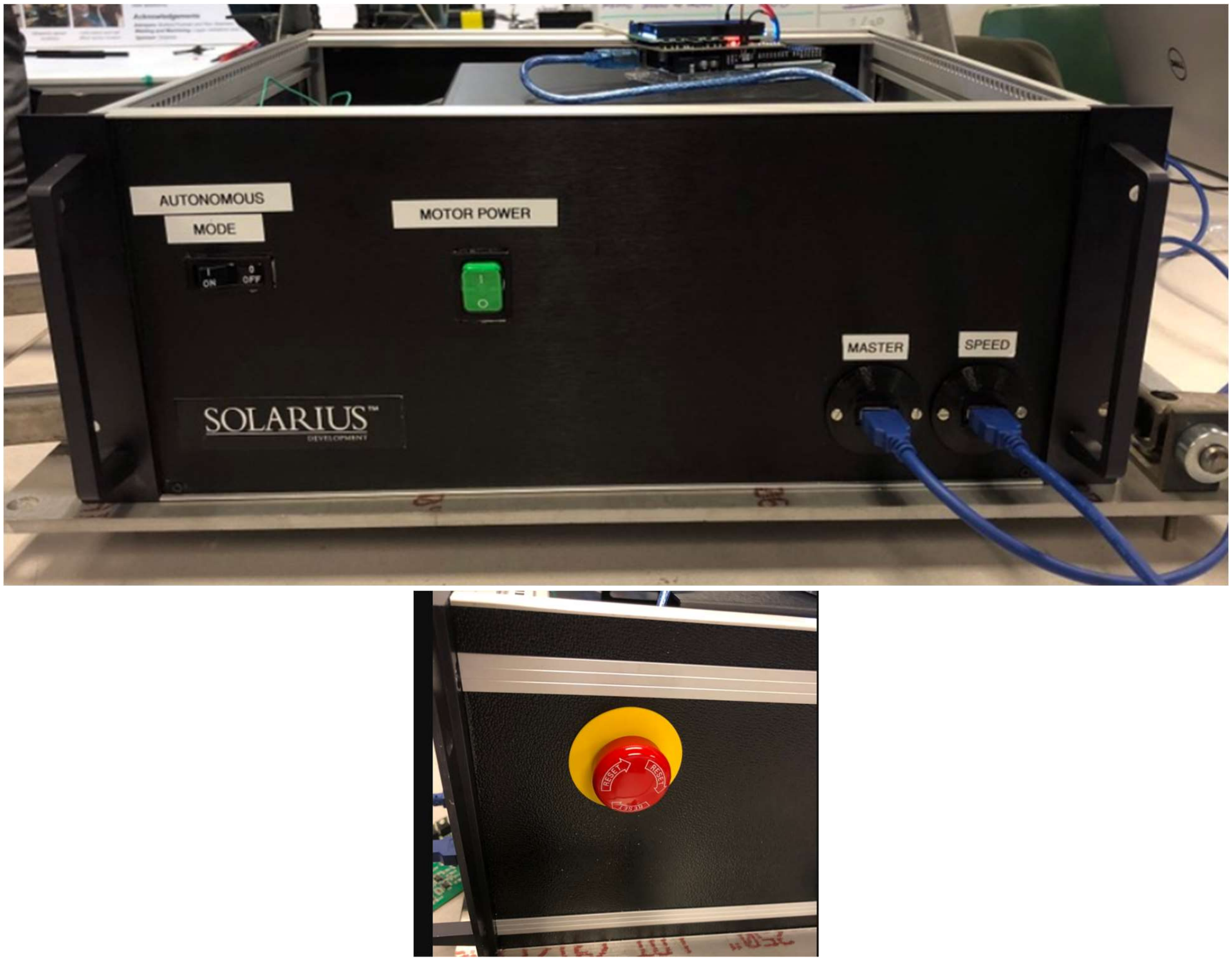


Figure 31: *The front of the control box with the autonomous mode toggle switch and a side view of the EMO switch.*

The powertrain was controlled with the use of front and rear ultrasonic sensors. These sensors continuously read the surface in front of them for any object to fall within certain criteria (defined by us). The bogie would move in one direction until the condition was met. Once this was met, the motors would stop and then begin to travel in the opposite direction. This would prevent the bogie from colliding with any object placed in front or behind it while it was moving.

Bogie turns are dependent on the triggering of a hall effect sensor placed on the bogie. There is a magnet on the track placed directly in the path of the sensor so as to be read whenever the bogie approaches the location. Once the hall effect sensor is triggered, the stepper motor will be commanded to move in order for the bogie to make a turn. Then in the case of overturning (oversteering), the limit switches will stop the stepper motor and the bogie will continue with the program. The programming will be further discussed in the following section.

Program Flow Chart

As previously mentioned, the autonomous mode is dependent on sensors for the operation of the DC powertrain motor and steering stepper motor. The main autonomous program, found in ***Appendix G***, consists of a function that is run when no button is pressed on the LCD screen and the toggle switch is activated. ***Figure 32*** displays the autonomous function that consists of two independent monitor functions for forward and backward motion. This function is declared in the "no button" case of the LCD shield case structure, shown in ***Figure 33***.

```
void autonomousMode() {
  monitorFront();
  monitorRear();
}
```

Figure 32: *The autonomous function.*

```
case btnNONE:
  {
   while (int x = digitalRead(autoSwitch)  == LOW) {
     lcd.clear();
     lcd.print("Autonomous Mode");
     Serial.println("Autonomous Mode");
     autonomousMode();
     break;
   }
```

Figure 33: *The LCD no button case.*

Figure 34 displays the program flow chart for the powertrain. Once the toggle switch for the autonomous mode is turned on, the bogie will begin motion as shown in the flow chart. The bogie will begin moving forward while continuously reading the front ultrasonic sensor. We established the criteria for safe travel to be any distance greater than 30 centimeters in front of the ultrasonic sensor. Once this condition is met, the bogie will have approached the end of the track or an object will have crossed the path of the bogie. The program will then stop the bogie and reverse the direction. The same conditions are applied to the backward motion. This will satisfy the powertrain autonomy as the bogie will continuously cycle through both cases and travel the track back and forth.

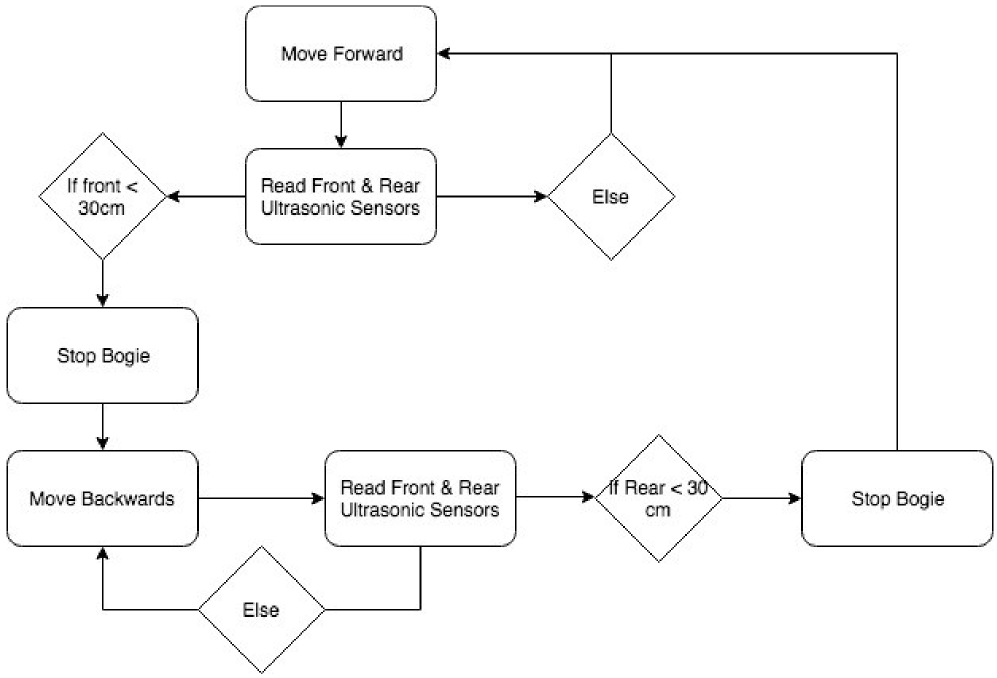


Figure 34: *The program flow chart for the powertrain system.*

In further detail, the program for forward/reverse motion is displayed in ***Figure 35***. Both monitors are similar in structure with the main difference being the direction of travel. The ultrasonic sensor reading is stored as an integer data type and compared to several if-else conditions. The bogie will travel for every condition that reads greater than 30 centimeters. Additionally, the hall effect sensor reading is stored as an integer data type and included in one of the if-else conditions. If the bogie is okay to keep traveling (30 centimeters of space in front/behind it) and the hall sensor is triggered (input = HIGH), then the stepper motor performs a turn while the bogie continues to run. The step direction is then inverted so the next time the hall effect sensor is triggered, the bogie will step in the opposite direction and change track path.

SPARTAN SUPERWAY

```
void monitorFront() {

  //Ultrasonic sensor begin reading
  int cm = sonar.ping_cm();

  //Look for hall sensor to turn
  int p = digitalRead(hallF);

  if (cm < 30) {
    // stop motor
    stopBogie();
    Serial.println("\t \t STOP BOGIE");
    delay(1000);
  }
  else if (cm >= 30 && cm < 80) {
    forward();
    Serial.println("\t \t \t \t \t \t SLOW DOWN");
  }
  else if (cm > 30 && p == HIGH) {
    forward();
    Serial.println("Turning Left");
    stpr.step(hallturn);
    hallturn = !hallturn;
  }
  else {
    // keep motor running analogWrite(ENA,SPEED);
    forward();
    Serial.println("\t \t \t \t \t \t GOOD TO GO");
  }
}
```

Figure 35: *The front monitor for autonomous mode.*

The program flow chart for the stepper motor is displayed in ***Figure 36***. Once the autonomous mode is initiated, the stepper motor will hold the current position and read the hall effect sensor. When the bogie approaches the magnet on the track, the hall effect sensor will read a logic HIGH signal and command the bogie to perform a turn, as mentioned in the program explanation of ***Figure 35***. After the step is performed, the position will be held and the step direction will be inverted (hallturn = !hallturn). This flowchart will endlessly cycle as long as the bogie keeps traveling and triggering the hall effect sensor. Both powertrain and steering systems work in conjunction with each other, as the powertrain is required in order for the steering to be functional.

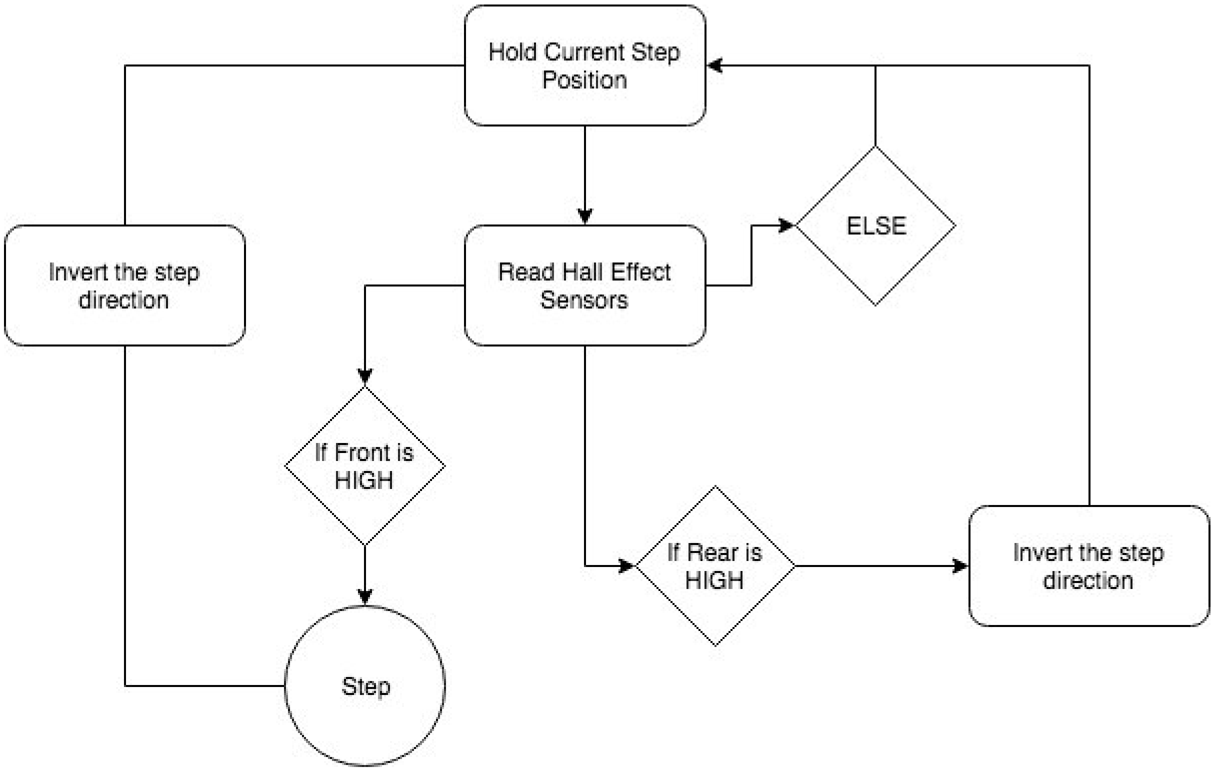


Figure 36: *The program flow chart for the steering system.*

Furthermore, limit switches have been integrated as attachInterrupt cases in the event of oversteering. This means they can be activated at any point in time that the code is running.
When they are activated, they immediately cut any movement occurring (steering or powertrain) and prevent any damage from happening. The “Interrupt Service Routine”, or “ISR” for short, function of the limit switches is displayed in ***Figure 37***, along with the limit switch setup. Both switches are introduced as volatile variables in the attachInterrupt function. The “falling” term refers to which when the condition for the activation is to happen. In this case, falling refers to when the input signal changes from HIGH to LOW.

```
//Limit Switches
attachInterrupt(digitalPinToInterrupt(limitL), isr, FALLING);
attachInterrupt(digitalPinToInterrupt(limitR), isr, FALLING);

void isr() {
  stpr.step(STOP);
  Serial.print("ISR");
}
```

Figure 37: *The attachInterrupt program for the limit switches.*

SPARTAN SUPERWAY

The final subsystem of the half-scale bogie features a velocity monitor. This will be situated on a separate Arduino and LCD shield in the control box. The velocity program can be found in ***Appendix G***. The program flow chart is shown in ***Figure 38***. The program initiates by setting up the millis time counter. Then an ISR function is introduced to create a numerical counter of the triggering on the hall effect sensor. The counting begins once the magnet is read in the main loop. Then the time between the magnet triggering is used in the calculation for the output velocity. If the magnet is not read and the time exceeds 2 seconds, the output velocity is declared to be zero and written to the serial monitor.

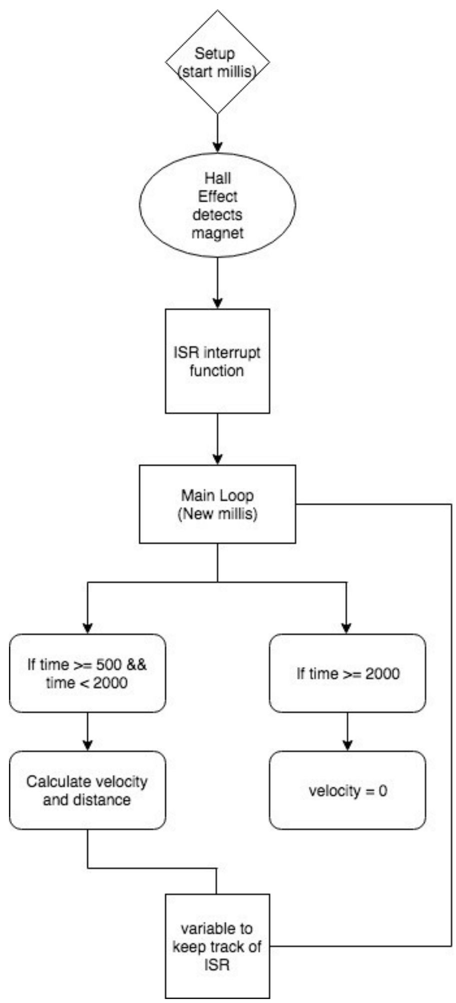


Figure 38: *The program flow chart for the velocity display.*

The calculation for the output velocity is displayed in ***Figure 39***. This calculation incorporates the wheel radius to calculate the circumference. The program determines the difference of how many times the magnet has been detected and then begins to monitor time elapsed since initiated. This time difference is measured in intervals between each magnet trigger and used in the formula to calculate output velocity, ***Equation 3***. This allows the instantaneous velocity to be determined as each time interval between magnet triggering results in output velocity. The resultant velocity is then written to the lcd screen with the use of the lcd.println((String(velocity) + "in/s"); command. The velocity will continuously update and be changing every time the hall effect detects the magnet on the wheel.

$$V_{out} = \frac{Circumference \cdot currentWheelSpinning \cdot 1000}{\Delta t} \quad (3)$$

```
void loop() {

  currentTime = millis();
  currentWheelSpinning = WheelSpinning - previousWheelSpinning;

  if(currentTime - startTime >= 500 && currentTime - startTime < 2000 ){

  velocity = (circumference * currentWheelSpinning *1000)/(currentTime - startTime);
 // distance+= circumference;
  lcd.setCursor(0,0);
  lcd.print("Speed= ");
  lcd.println(String(velocity) + " in/s");
 // lcd.setCursor(0,0);
  //lcd.print("distance= ");
  //lcd.println((distance) "in");

  previousWheelSpinning = WheelSpinning;
  startTime = millis();

  } else if ( currentTime - startTime >= 2000) {
    velocity = 0;
    startTime = millis();
  }
}
void magnet() {

  WheelSpinning++;
}
```

Figure 39: *The main program loop to calculate instantaneous output velocity.*

Analysis/Validation/Testing

Prior to the assembly of the bogie control box, major subsystems were vigorously tested for consistency and reliability. The first testing phases were initiated during the first semester of the project. The powertrain was amongst the first mechatronic tests. Once this was verified to be stable, testing was done on the stepper motor, new mechanical chain, gears, and integrated sensors.

Mechanical

On the mechanical design, the analysis on the gear ratios were calculated to get it closely to the past years design specifications. The calculations are demonstrated in ***Figures 61 and 62 in Appendix D***, the new metal gears were able to output a new gear ratio of 2.78 for the lower control arm. The gear ratio for the upper control arm stayed the same. The stepper motor outputs a torque of 106 inch-pounds and a speed of 50 RPM to the new 3/8 in shaft, and which ultimately the steering mechanism will be able to handle with no issue. The gear ratio setup was designed to increase the torque while obtaining a permissible speed for the bogie when making turns and switches along the track. As everyone knows, if the torque increases the speed decreases and vice versa. This allowed the lower control arm to travel two times faster than the upper control arm since it requires more speed to travel when the bogie is preparing to turn.

Analysis was done using SOLIDWORKS FEA for the shafts that are were replaced. All the analysis from SOLIDWORKS can be seen in the ***Figures 63 through 68*** of ***Appendix E*** . The FEA results gave insight and visualization of whether the new designs would function successfully. Also, the results can be compared to the current designs of the shafts. Since there was no documentation on the material of the current shafts, the assumed material for each shaft was carbon steel per instructor recommendation . The new upper and lower shafts are going to be carbon steel with a yield strength of 60,000 psi. The drive shaft will be carbon steel with a yield strength 75,000 psi. The upper shaft connected to the stepper motor receives a torque of 106 inch-pounds, which yielded a safety factor of 9.41 and displacement of 0.091 inches. The safety factor was much higher than expected, which was probably due to the fact that FEA included the stepper coupler that connects the upper shaft with the stepper motor. The bottom shaft experiences a torque of 294.44 inch-pounds, which resulted in a safety factor of 2.80 and displacement of 0.14 inches. For the drive shaft, a calculated torque of 416.66 inch-pounds as shown in ***Figure 67***, resulted in a safety factor of 3.8 and displacement of 0.326 inches. Overall, all analyses gave great feedback and resulted in a better safety factor relative to the current setup.

Lastly, once the shafts were manufactured to the correct diameter the new selected carbon steel gears and roller chain were mounted onto the bogie. This gears and roller chain gave the bogie a clean finish looked which made it look more the traditional mechanical design. Once the roller chain was put on the gears, it resulted on being a little loose and had some. The roller chain was tested to see if there could be any possibility of slippage when the bogie would be operiona. Several tests were ran with the stepper motor on making the steering mechanism go back and forth to ensure the roller chain was properly working. Results turned out be efficient since there was no slippage occuring and the roller chain was grabbing on to the gears as expected. When the stepper motor was giving a command to turn to the left, both steering arms turned to the left with no issue and vise versa when the stepper motor was commanded to turn the right. Also, removing one link from the roller chain was taken into consideration as well, but it results on the roller chain being too tight and not fit onto the gears perfectly.

Mechatronics

Powertrain

During the first semester of the project, the powertrain was tested in a manual operation to ensure consistent motor performance prior to autonomous development. The mechatronic design of the DC motor control system involved the use of a DC motor driver, which was believed to be the simplest and easiest way to achieve a reliable setup. The driver, shown in ***Figure 40***, featured a dual H-bridge transistor configuration with PWM (Pulse Width Modulation) capability. Limit switches were integrated into the directional pins, between our microcontroller inputs. When pressed, the driver would receive the logical command to command the motor either forward or backward. This seemed to be functional until the driver overheated and damaged one of the transistors. The reason for failure was believed to be due to the high current used (6 amps from our 24V power supply) and insufficient heat management, as there were no heat sinks or cooling fans on the driver.

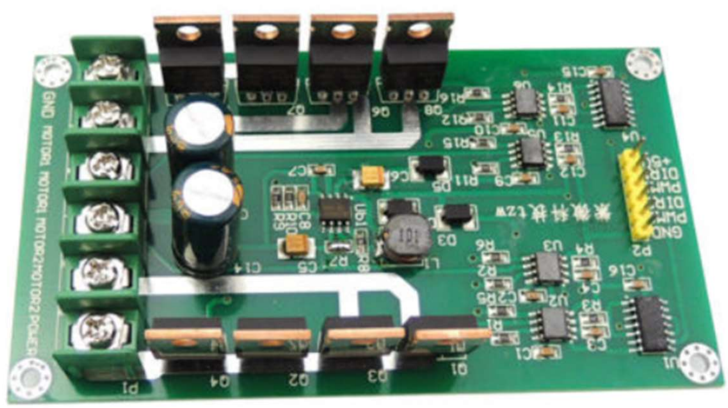


Figure 40: *RioRand DC PWM Dual Motor Driver Controller.*

Since it was believed that the issue was due to be a lack of heat management, the same motor controller was ordered and setup with CPU cooling fans, as displayed in ***Figure 41***. However, the new fan integration did not resolve the overheating issue. The motor driver again overheated and damaged one of the transistors. The damage is circled in red on ***Figure 42***.

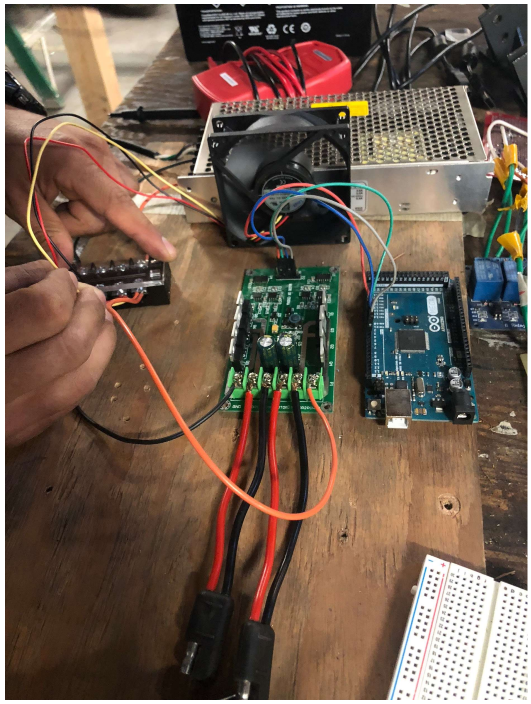


Figure 41: *The setup with the cooling fan pointed at the motor driver. The fan base and switches are not yet mounted.*

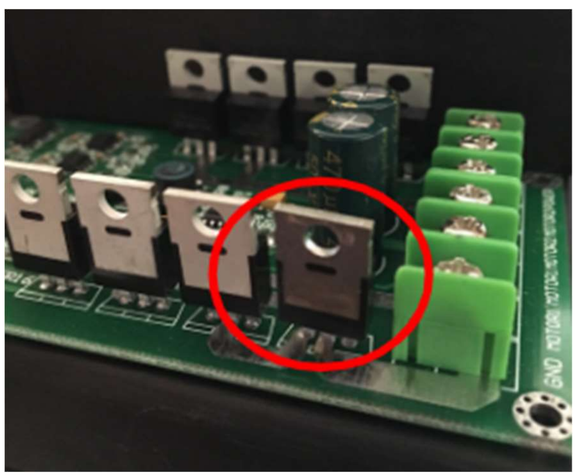


Figure 42: *Closeup on the damaged MOSFET on the motor driver. MOSFET is circled in red.*

After this second failure, it was decided to find a more suitable motor controller with better heat management. This is when the 16 Amp DROK dual H-bridge motor driver was selected for use, shown in ***Figure 42***. During initial testing, the new driver worked fine. But upon higher usage, it eventually emitted an audible snap, or "pop", and then became entirely nonfunctional. The driver had not overheated this time, rather it encountered an unknown error. Based on the manufacturer's description, the scenario in which this driver could fail as it did would have been a short circuit. The electrical system was revised and the driver was then again replaced.

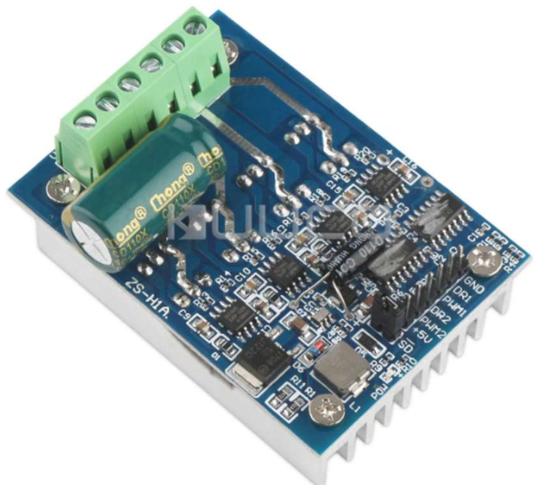


Figure 43: *DROK 16 Amp dual H-bridge motor controller***.**

For the beginning of the second semester of the project, we decided to purchase a high amperage motor driver to ensure there was no issue with overloading during operation and to establish a secure and reliable powertrain, as this became a persistent issue. Additionally, we ordered a secondary backup motor controller similar to the DROK driver in ***Figure 43***. The new motor driver, "100A DC Drive Module Motor Speed Controller Dual Channel H-bridge Optocoupler Isolation", displayed in ***Figure 44***, was soldered with a DB9 connector and integrated into the control box for testing.

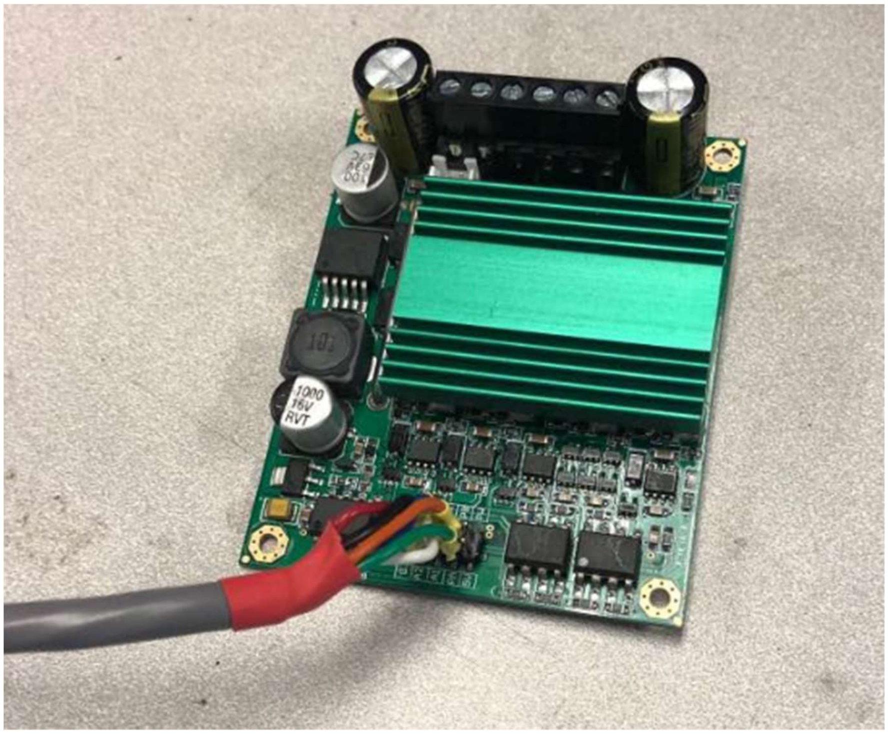


Figure 44: *100A DC Drive Module Motor Speed Controller Dual Channel H-bridge Optocoupler Isolation with soldered DB9 connector.*

However, the high amperage motor driver was nonfunctional. There was no output voltage output to the motors. This was monitored with a multimeter as the motor commands were pressed with the LCD shield. All the driver pins were carefully inspected to ensure there was no problem with the soldering. The Arduino output pins were then metered to ensure the HIGH and LOW signals were being sent to the driver. Everything was inspected as required and nothing seemed out of the ordinary. It was concluded that the new high amperage motor driver could be defective so it was returned to the manufacturer and will be replaced and retested during the summer. This left the DROK 16 amp motor as a backup for the project demonstration. But as had happened before, it failed during the demonstration and forced the team to use the emergency shut-off switch for the control box as the motor driver quickly began to smoke.

Steering

The stepper motor was successfully integrated into the programming for manual and autonomous control. Once the working wiring configuration was set up, the stepper motor properly turned the required step size reliably and accurately. This was independently tested before being mounted on the bogie, and tested once mounted by articulating the new chain and sprocket gears to ensure there is no issue with the new setup. The bogie was able to articulate both arms in the movement. The selected step size and speed was deemed to be suitable for operation.

Sensors

The first sensors to be tested were the ultrasonic sensors. Integrating one ultrasonic sensor was no issue, as there were plenty of online resources to facilitate this. The challenge was integrating multiple ultrasonic sensors and display their readings. This was done before any mounting on the bogie. Two ultrasonic sensors were set up in the same program and Arduino was able to use both to display live readings. The resultant readings are displayed in ***Figure 45***.

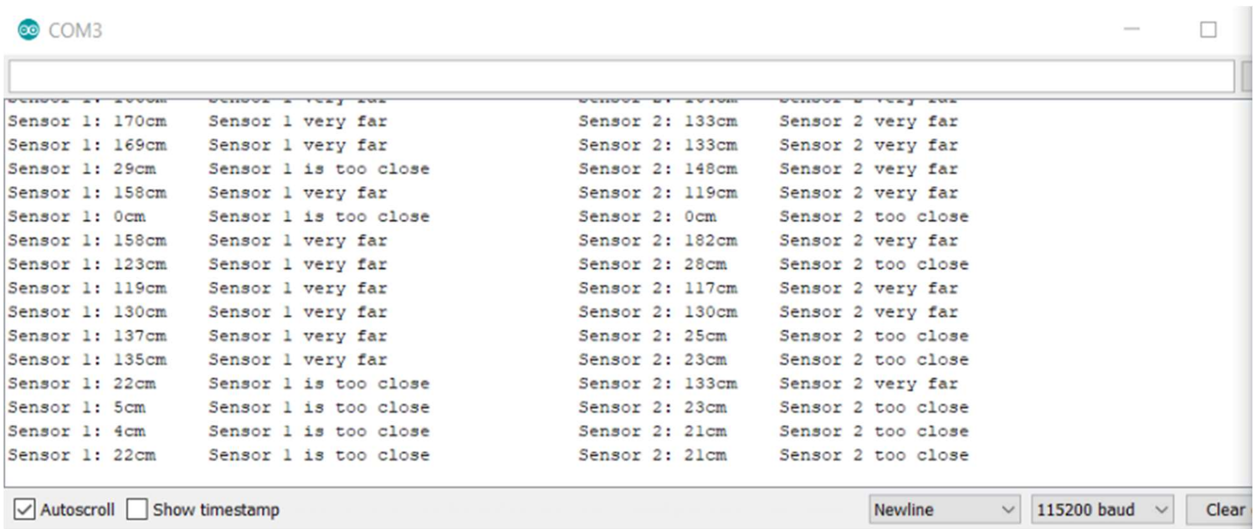


Figure 45: *The serial monitor displaying two ultrasonic sensors.*

The limit switches functioned well in the oversteering protection operation. When the bogie overturned, the limit switch immediately stopped the stepper motor and the bogie held the current position. This was tested with the bogie off the track, where it was possible to relive the issue if the limit switches failed.

The final sensor required for testing was the hall effect sensor. In order to test these, we used a modified version of the velocity code to display a serial message whenever the sensor was triggered. This properly functioned as we rapidly moved the magnet in and out of the working range of the sensor.

Money Spent on the Project

As mentioned in the Acknowledgments, Solarius provided a budget of $750 for this project after being presented with a proposal. The budget accounted for mishaps and faulty materials and devices. Many of the motor drivers ended up being faulty or burning out which used up most of the backup funds. Most of the budget was allocated to new materials and any equipment required for assembly. This included chains, sprockets, and bearing for upgrading the steering and powertrain, soldering equipment, metric screws and nuts, and various electrical components such as wires, a multimeter, and MOSFETs. Some funds were also used for purchasing and machining shafts. The remaining budget was allocated to sensors required for automation. These consisted of hall effect sensors with magnets, ultrasonic sensors, Xbee wireless communication devices, and Arduino microcontrollers with LCD (Liquid Crystal Display) shields. This project was a continuation from the 2016-2017 semesters, so some equipment and materials did not need to be purchased. This included the stepper and powertrain motors, the materials and machining for the bogie chassis, wheels, a power supply, and various tools such as drills, screwdrivers, and hex keys. Ultimately, the project went $77 over budget for a grand total of $827. A complete breakdown of the described budget can be found in ***Appendix B***, ***Tables 3*** and ***5***.

Results and Discussion

This project proved this system can be functional and acts as a solid proof of concept for the Superway as a whole. This also goes to show that the half scale bogie and controls is worth the time and money for future development. The 2018-2019 team made a lot of progress for this project and set a foundation for the upcoming years which involve both the mechanical and controls systems designed this year. The wiring and organization from the previous mechatronic team made it difficult for this year's team to get caught up to speed and figure out what need to be accomplished moving forward. The team made sure that designs, schematics and the labeling of the control box were documented such that it should not take future teams very long to get up to speed. One of the most elaborate documentations is that of the sensors and motor drivers to Arduino pins. This will be left in the form of an accessible document in the respective subteam Google drive. The stepper motor was established to provide consistent and reliable operation. The sensors were individually tested and integrated into the master code. The protective limit switches successfully worked in stopping the stepper motor when it would overturn and activate the switches. The ultrasonic sensors were successfully integrated into the master code after they were verified to be functional independently. The DC motor was proven to be functional, but an issue persists with the drivetrain controls. DC motor drivers become unresponsive or overheat and become damaged. There have been many approaches to resolve this issue, most of which articulate new motor driver boards to attempt to find a functional one. One final attempt by the 2018-2019 half scale team involved replacing the latest 100 amp motor driver in the case that the first driver was defective. If this were to not work, there will be a final test to construct a dual H-bridge setup with electromechanical relay boards, as this would be the most elementary and robust solution to get the powertrain functional and reliable.

Conclusions and Suggestions for Future Work

One goal the team was not able to achieve was the wireless communication portion of the project. The research for the wireless communication began in the second half of the year and a wireless feature would have been used to control the bogie with a remote. Through research of the various types of wireless modules that could be used, the XBee was recommended to us for its user-friendly interface. The small scale team was already working with the XBee modules prior and so we collaborated with them first instead of purchasing our own chips. From the team not being able to figure out the system within a couple weeks, it was decided that the NRF24L01 module that the full-scale team was working on would be more reliable. The difference in price between both modules also led us into purchasing the NRF24L01 modules instead of the XBee. Each XBee module is priced around $50 while the NRF24L01 modules were less than $1 each. The wiring for the NRF24L01 to an Arduino Uno and Arduino Mega is shown in relation of ***Table 2*** and ***Figure 46*** below.

Figure 46: *Pin Layout on the NRF24L01.*

Table 2: *Wiring Guide for Arduino to NRF24L01 Pin Connection* ***in Figure 46.***

Function	NRF24L01 Pin Number (*Figure 46*)	Arduino (UNO) Pin	Arduino (MEGA) Pin
GND	1	GND	GND
Vcc	2	3.3V	3.3V
CE	3	D7	D7
CS	4	D8	D8
SCK	5	D13	D52
MOSI	6	D11	D51
MISO	7	D12	D50

The diagram that was followed to achieve a working wireless communication is shown below in ***Figure 47.***

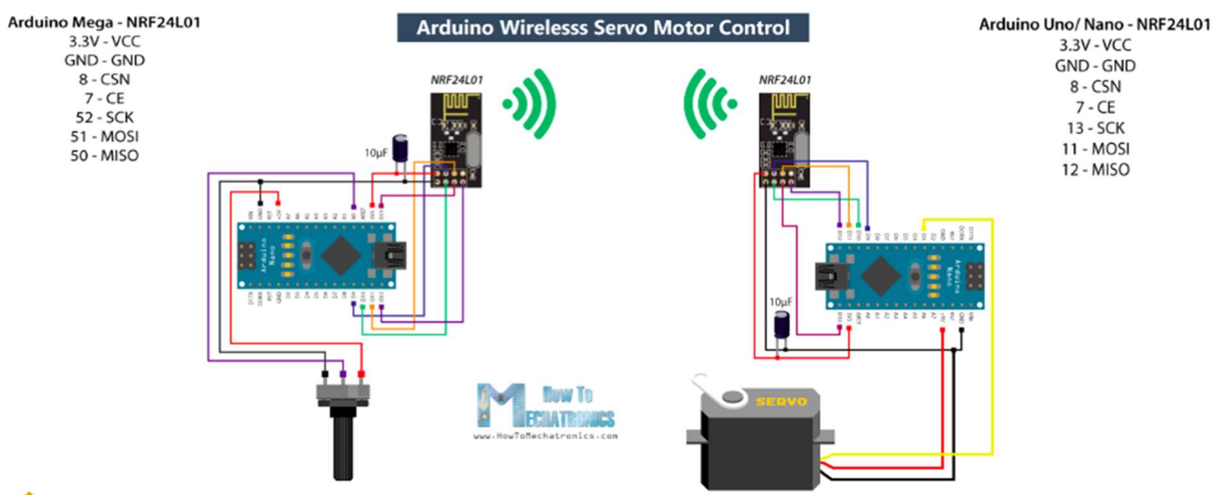


Figure 47: *Wiring Diagram for the NRF24L01 to Arduino, servo, and switch (Dejan, n.d.).*

The code for the NRF24L01 uses its own library that can be downloaded from the GitHub link found on the How to Mechatronics site (Dejan, n.d). It is noticed that two codes can not be run on the same laptop, so two different computers were used to run the receiver and transmitter codes simultaneously. With the RF24 library included in the code, the radio pipe must set the same on both the receiver and transmitter codes as well as the address name. In the Setup, the radio frequency is set and the power amplifier (PA) level is set at maximum for a reduction in possible errors. The writing and reading pipes are opened in the transmitter and receiver, respectfully. Once the functions and parameters are set, the important aspects of the codes are the write and read functions for each. For example the:

```
" radio.write(&text, sizeof(text));
radio.write(&ledState, sizeof(ledState));"
```

in the transmitter and

```
"if(radio.available()) {
  while(radio.available())
  {
    radio.read(&text, sizeof(text));
    radio.read(&ledState, sizeof(ledState));
  }"
```

of the receiver. A code to turn on an LED was written to test the setup of the code. The full codes are included below in **Appendix I**. It is recommended that future groups work with the given example codes provided on the Arduino IDE software and practice integrating the wireless functions into them. This will act as integration of the wireless module with the existing master motor and sensor code of the entire system. With this set up, the wireless module and Arduino can safely be placed in a covered box for protection from the elements.

For integrating wireless communication on the bogie, a sheet metal box and covered was designed to place a slave Arduino onto the bogie. The sheet metal will go on top of the gearbox where there is enough space for it. The purpose of this design was to allow for a slave Arduino to be placed on top of the bogie. All the sensors, such as the hall effect, limit switches, and ultrasonic will be able to connect to this box and free up space from the control box. This will eliminate all the long wires coming from the control box to the bogie. The sheet metal box will ensure the wires will be shorter, since all the sensors are fairly close to the gearbox. Currently, all of the mechatronics are connected to the control box and ideally, the wires should not be coming out of the control box and interfering with the bogies movement. Next year's team can focus on the wireless design and autonomous functionality. The sheet metal box and cover were 3D printed to check the fit. Unfortunately, since wireless communication was not achieved this year, the team did not proceed on manufacturing the sheet metal box and cover. The next team can pick up were the 2018-2019 half scale bogie and controls left off. ***Figures 48-50***, show and represent the location of the sheet metal box and the design.

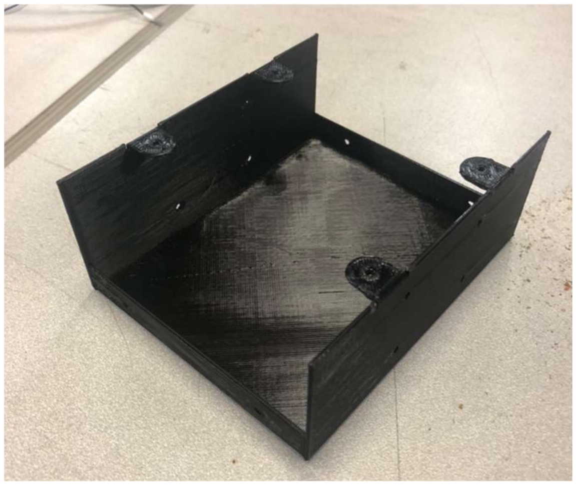


Figure 48: *A picture of the sheet metal box being 3D printed.*

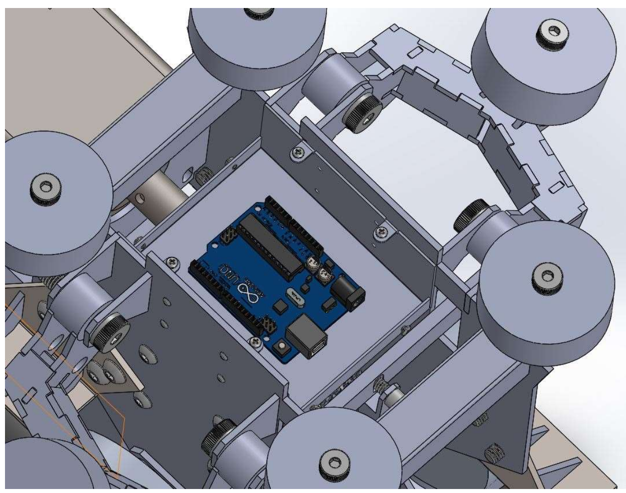


Figure 49: *A CAD picture showing how the slave Arduino fits inside the sheet metal box.*

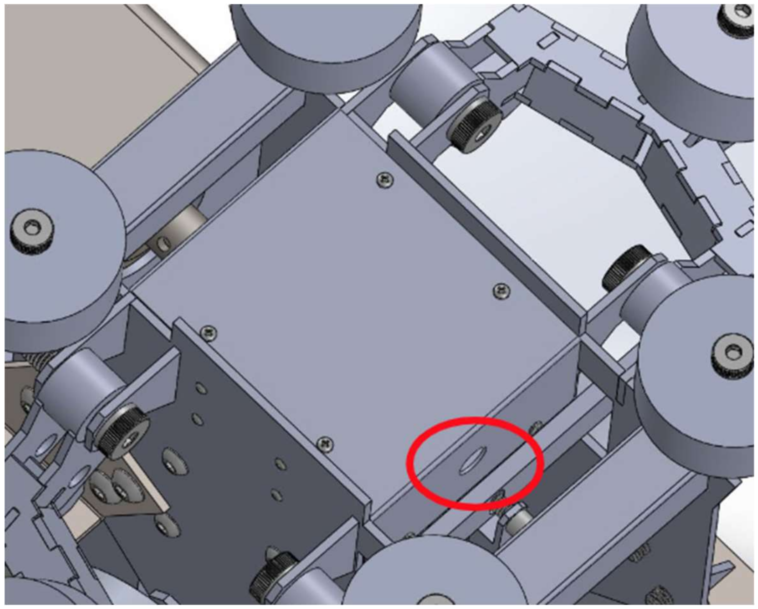


Figure 50: *A CAD picture of the complete assembly of the sheet metal box and cover.*

One problem that was noted by a previous team was that the powertrain shaft was experiencing torsion when the DC motor was running at full power and the bogie was going up the incline of the track. Also, the drive shaft was not symmetrical when it was taken off from the bogie and examined, allowing for an uneven distribution of stresses. Therefore, a new driver shaft was designed to be more symmetrical and eliminate the need for the spacers between the drive shaft and the wheel ***Figure 51***. A symmetric drive shaft with a longer inner diameter of 3/4 inches will reduce the torsion it is currently experiencing. The drive shaft was 3D printed to ensure it fit onto the bogie's powertrain design before it was manufactured. The 3D printed drive shaft can be seen in ***Figure 52***. The new 3D printed drive shaft fit perfectly and it removed the need for the spacers. Unfortunately, due to time constraints the drive shaft was not machined because operation of the bogie with the all sensors was more of a priority. This is something next year's team can accomplish. ***Figure 53***, shows the redesigned driver shaft on the bogie.

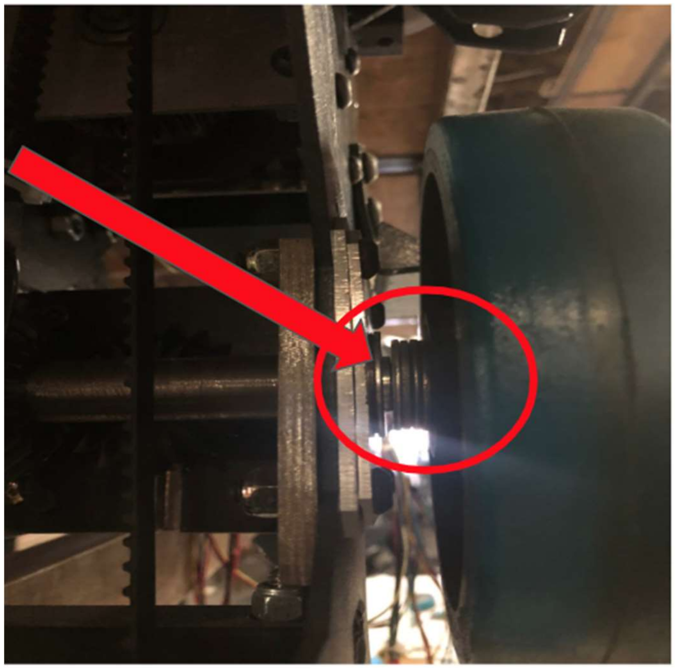


Figure 51: *A picture of the driver shaft showing the spacers.*

Figure 52: *A picture of the 3D driver shaft.*

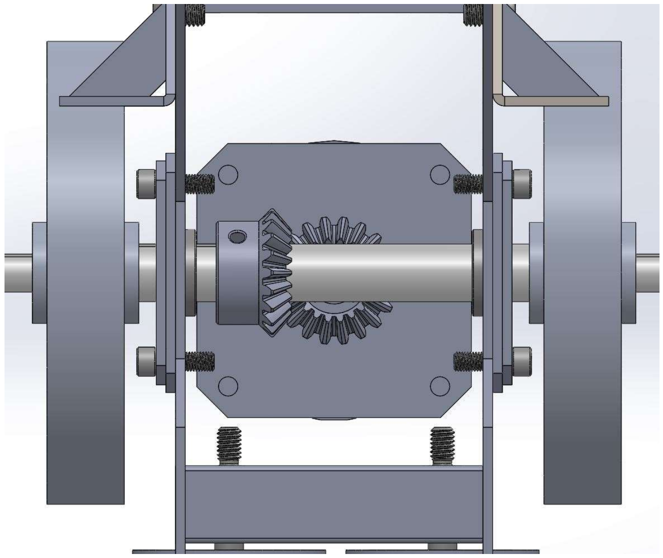


Figure 53: *A CAD picture of the new design of the driver shaft.*

Next year's team can easily fix and focus on replacing the upper carbon steel gear with a new gear with more teeth. The current gear has nine teeth, which is why the roller chain has some slack. It would be wise to get a gear with ten teeth and the same shaft diameter. This new ten tooth gear will ensure the roller chain has adequate tension. Also, this will result in a new gear ratio for the lower control arm of 2.5, which is lower than the current ratio of 2.78. The lower gear ratio should still be fast enough to make proper turns and switches on the track.

As previously mentioned in the results, the powertrain system can make use of a more robust control system. Since there have been various motor drivers that have not worked, it is recommended to pursue alternate methods of control.

More possible areas of future work include the integration of wayside power. This year, the bogie was not able to be placed on the track and tested for full functionality. There was no time to test the bogies on the track with the sensors in place. The team thought of hanging a battery under the bogie and possibly under the control box for wireless power. The 2017-2018 wayside power team designed an energy collecting system on the track, but it was not tested. With the shortened track, the slope was not able to be accounted for. The seventeen degree slope that was removed from the track can be tested separately so that space would not be an issue.

Conclusions and Recommendations

In the end, this project resulted in the successful operation of the entire system on the half scale level, which can be scaled up to the full scale model. The team made the controls system adaptable, allowing for solar power integration. The 2018-2019 half scale team of the Spartan Superway has shed light on many aspects which can be implemented in the full scale system. The strong proof of concept and fast prototyping the half scale team showcases is on par with all the other teams connected through the project. In addition, this year’s team has implemented various modern ideas into both their mechatronics and mechanical design systems which will act as a guide for the future half scale teams.

Further, as one looks at the bigger picture, Spartan Superway is making a huge footprint on the global level in its own small and innovative ways, involving the younger generation in manufacturing a mass transportation of the future being above of them all. Each step Spartan Superway takes is a huge leap towards sustainable development and promoting this long neglected idea of using renewable energy.

SPARTAN SUPERWAY

References

Bart (Bay Area Rapid Transit). East Contra Costa BART Extension FAQ. Retrieved from https://www.bart.gov/about/projects/ecc/faq

Bendix, A. (2015, September 24). How London and Berlin's Daily Travel Habits Compare to the U.S. Retrieved from https://www.citylab.com/transportation/2015/09/how-london-and-berlins-daily-travel-habits-compare-to-the-us/406840/

Futran. The Futran System: The Future of Transportation is Now Today - Background. (2018) Retrieved from https://futrangroup.com/

Inist. (2017, March 14). Spartan Superway Development: A White Paper. Retrieved from https://www.inist.org/library/2017-03-14.FurmanSwensonHagstrom.SpartanSuperwayWhitePaper.SpartanSuperway.pdf

JPods. Advantages of JPods. Retrieved from https://www.jpods.com/advantages

Milotek:Innovating Transportation (n.d.) Retrieved from http://www.milotek.co.za/

SFMTA. (n.d). San Francisco Transportation Trends. P23. Retrieved from https://www.sfmta.com/sites/default/files/reports-and-documents/2018/01/san_francisco_transportation_trends_2.3.15.pdf

NHTSA (National Highway Traffic Safety Administration). Traffic Safety Facts Annual Report Tables: National Statistics. Retrieved from https://cdan.nhtsa.gov/tsftables/tsfar.htm#

SkyTran. (n.d.). Retrieved from https://www.skytran.com/

United States' Census Bureau. (n.d.). Quickfacts: San Francisco County, California. Retrieved from https://www.census.gov/quickfacts/sanfranciscocountycalifornia

University of Washington. (2018, October 1). Urban Transit Technologies - Suspended Vehicles Index Page #2. Retrieved from. https://staff.washington.edu/jbs/itrans/itrans2.htm

JPods. Shaxian, China agreement to build JPods Solar-powered Mobility Networks. Retrieved from: https://www.jpods.com/Shaxian

Dejan. (n.d) Arduino Wireless Communication - NRF24L01. How to Mechatronics. Retrieved from https://howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/
Secondary site: https://github.com/nRF24/RF24

SPARTAN SUPERWAY

Appendix A - Gantt Charts

Table 3: *Gantt chart for the 2018 semester.*

2018 Semester	9/3	10/31	10/10	10/17	10/23	10/31	11/7	11/15	11/30	12/7	12/14
Research											
Chain and Sprocket Design											
Wheel Research											
Motor Calculations											
Arduino Code											
Mechatronic Design											
Test and Debug Code											
Sponsorship											
Bogie Calculations											
Arduino Control, Drivetrain											
LCD Screen											
Arduino Control, Steering											
Arduino Drivetrain/Steering Integration											
Test/Fit/Machine Parts											
Testing Motion (Switches)											
Final Report											
Debug											

Table 4: *Gantt chart for the 2019 semester.*

2019 Semester	**1/30**	**2/13**	**2/20**	**2/27**	**3/6**	**3/20**	**4/24**	**5/8**	**Responsible**
Order Parts									Oscar
Steering									Oscar
Limit Switch integration									Oscar
Manufacture Shafts									Matthew
Integrate to LCD									ALL
Order Ultrasonic Sensor									Oscar
Code Ultrasonic Sensors									Karandeep
Xbee Communication									Karen
Manufacture Mid-Shafts									ALL
Velocity Integration									Hector
Powertrain Integration									ALL
Sensor Housing									ALL
Bogie Assembly									ALL
Full Track Assembly									ALL

Appendix B - Bill of Materials

Table 5: *Bill of materials for the 2018-2019 year.*

Item	Description	QTY	Cost (total)	Source
Gears	Mechanical	6	$ 170.00	Mcmaster
Shafts	Mechanical	6	$ 150.00	Mcmaster
Motor Driver 100 amp	Electrical	1	$ 105.00	Amazon
Motor Drivers	Electrical	4	$ 80.00	Amazon
Electrical	Electrical	10	$ 60.00	Amazon
Chain	Mechanical	2	$ 45.00	Mcmaster
Machining Shafts	Mechanical	1	$ 40.00	Locally sourced
3D Print filaments	Mechanical	1	$ 30.00	Fry's
Tap and Drill Set	Mechanical	1	$ 30.00	Harbor Freight
Bearings	Mechanical	4	$ 30.00	Mcmaster
Breadboards & Wires	Electrical	3	$ 25.00	Amazon
Motor Driver 8 amp	Electrical	1	$ 20.00	Amazon
Fuses	Electrical	12	$ 17.00	Amazon
LCD Shields	Electrical	4	$ 15.00	Amazon
Ultrasonic sensors	Electrical	6	$ 10.00	Amazon
		Total	$ 827.00	

SPARTAN SUPERWAY

Appendix C - Data Sheets

larger image

Item # PK296A2A-SG36, Stepper Motor
Web Price $268.00

Incorporating the SH gears with high permissible torque delivers high resolution, high torque and smooth low-speed rotation.

RoHS

Unit of Measure: Imperial | Metric

Lead Time · Specifications

Lead Time

Available to Ship[1]	10/25/2016

[1] Quoted Ship Date for orders placed before 12:00 pm PST. Quantities may affect Shipping Date.

Specifications

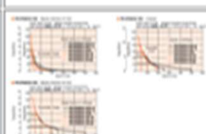
Motor Type	2-Phase
Frame Size	3.54 in
Motor Length ?	4.96 in.
Speed-Torque Characteristics	 Speed - Torque Characteristics larger image
Holding Torque ?	1696 oz-in
Shaft/Gear Type	Spur Gear
Gear Ratio (X:1)	36 :1
Shaft	Single

Figure 54: *The DataSheet for the stepper motor.*

Gearmotor Specifications

Model Single Shaft Double Shaft	Gear Ratio	Holding Torque*		Step Angle	Permissible Speed
		N·m	lb-in		r/min
PK296A1A-SG3.6, PK296A2A-SG3.6 PK296B1A-SG3.6, PK296B2A-SG3.6	3.6:1	2.5	22	0.5°	500
PK296A1A-SG7.2, PK296A2A-SG7.2 PK296B1A-SG7.2, PK296B2A-SG7.2	7.2:1	5	44	0.25°	250
PK296A1A-SG9, PK296A2A-SG9 PK296B1A-SG9, PK296B2A-SG9	9:1	6.3	55	0.2°	200
PK296A1A-SG10, PK296A2A-SG10 PK296B1A-SG10, PK296B2A-SG10	10:1	7	61	0.18°	180
PK296A1A-SG18, PK296A2A-SG18 PK296B1A-SG18, PK296B2A-SG18	18:1	9	79	0.1°	100
PK296A1A-SG36, PK296A2A-SG36 PK296B1A-SG36, PK296B2A-SG36	36:1	12	106	0.05°	50

* Holding torque is the same regardless of the connection type, due to the permissible torque limit of the gearhead.

Figure 55: *The specifications of our stepper motor.*

MOTORTEC™ DATA SHEET

Prepared by: Groschopp.com 01/17/19

Motor: **PM 8018**
Voltage: **24 v DC**
Speed: **2500 rpm 2 Poles**
Gearbox: **None**

Product ID: **4101**

SPECIFICATIONS

Rating:
Speed: 2508.0 rpm
Torque: 5.7lb-in
Current: 8.77 amps
Output: 168 watts
Output: 0.2248 HP

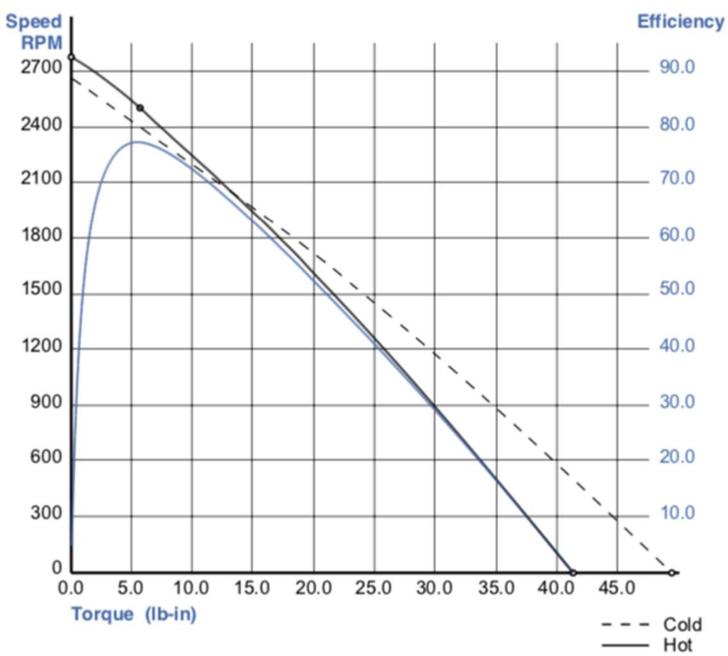
Duty Cycle:
On: Continuous
Off:

Efficiency:
Gearbox:
Motor: 77.4 %
System: 77.4 %

Start/Stall Conditions:
Current: 66.86 amps
Torque: 49.38 lb-in

Constants:
Ke: 8.91 v/krpm
Kt: 0.75 lb-in/amp

Speed, Efficiency Vs. Torque



Notes:

GROSCHOPP
Groschopp, Inc., 420 15th Street NE, Sioux Center, IA 51250-2100 USA
(712-) 722-4135 Phone • (800) 829-4135 Toll Free • (712) 722-1445 FAX
www.groschopp.com

Figure 56: *The DataSheet of the driver shaft.*

SPARTAN SUPERWAY

Appendix D - Calculations

Drivetrain Motor Calculations

Motor Holding Torque Calculation

Mass of the bogie (in lbs)	75	$M = rF_{tr} + \frac{mra_x}{2}$
Mass bogie carries (in lbs)	0	$M - r(\mu mgCos(\alpha)) = I\beta$
Angle of incline (17 degrees in radians)	0.296706	
Coefficient of Friction	0.65	Since $a = r\alpha$
Radius of the wheel (in inches)	3.25	$M - r(\mu mgCos(\alpha)) = (1/2\ Mr^2)(a/r)$
Acceleration due to gravity (in inches/sec^2)	385.826772	
Acceleration up the inclie, a	0	$M = mr\ (\frac{1}{2}a + \mu gCos\alpha)$
Factor of safety, F (used as a standard in automobile industry)	2.5	$F_{ext} - mgSin(\alpha) + \mu mgCos(\alpha) = ma_x$
(Keeping in mind the losses in the geartrain and bearings)	1.1	$F_{ext} = ma_x + mgSin(\alpha) - \mu mgCos(\alpha)$
Holding Torque Required	**416.6649672 in inch lbs**	**→ 47.07480799 in Nm**

Figure 57: *Output torque and power requirements.*

Motor Input Table

$$Gear\ Ratio = \frac{Torque_{output}}{Torque_{input}} = 160:1$$

$$\frac{416.665\ lb\ in}{T_{in}} = \frac{160}{1}$$

$$\textbf{With 73\% efficiency}, Torque_{in} = 3.567\ lb\ in.$$

$$Given\ Motor\ torque\ constant, K_t = 0.69\ lb\ in./amp$$

$$K_t = \frac{T}{I_A}$$

$$I = \frac{T}{K_t}$$

$$I = \frac{3.567\ lb\ in.}{0.69\ lb\ in./amp} = 5.17\ Amps < 9Amps\ (Rating\ on\ each\ motor\ and\ motor\ driver)$$

$$Operating\ at\ 24\ V, Power\ input\ power\ required\ for\ each\ motor:$$

$$P = I * V$$

$$P = 5.17Amps * 24V = 124.08W < 166W\ (Max.\ power\ rating\ on\ each\ motor)$$

$$\therefore Total\ Power\ required\ for\ both\ motors\ combined = 248.16W$$

Figure 58: *Input torque and power requirements.*

SPARTAN SUPERWAY

$$17.629\ rpm * \frac{20.4\ in}{1\ rev} * \frac{1\ min}{60\ s} = 5.999\ \frac{in}{s} = v$$

$$v = \frac{d}{t}$$

$$t = \frac{d}{v} = \frac{140\ ft}{5.999\ in/s} = \frac{1680\ in}{5.999\ in/s} = 280\ s \approx 4.5\ min$$

Figure 59: *Time required to traverse the track.*

Gear Ratio Calculations

Motor Torque Power = 106 lb•in and a
Speed of 50 RPM

Current Gear ratios on the steering arms

$$\text{Gear Ratio} = \frac{Output}{Input}$$

$$\text{Gear ratio of top linkage} = \frac{Output}{Input} = \frac{48}{16} = 3$$

$$\text{Gear ratio of plastic gears} = \frac{Output}{Input} = \frac{28}{10} = 2.8$$

Current Torques on the steering arms

$$T_{out} = T_{in} \times GR$$

$$T_{out(upper)} = (106\ \text{lb•in})(3) = 318\ \text{lb•in}$$

$$T_{out(lower)} = (106\ \text{lb•in})(2.8) = 296.8\ \text{lb•in}$$

Current speed of the steering arms

$$\text{Speed} = \frac{RPM_{in}}{GR}$$

$$\text{UCA speed} = \frac{50\ RPM}{3} = 16.6\ RPM$$

$$\text{LCA speed} = \frac{50\ RPM}{2.8} = 17.86\ RPM$$

Figure 60: *Calculations for the current gear ratios (Gear Ratios, n.d.).*

New Gear ratios on the steering arms

$$\text{Gear ratio of top linkage} = \frac{Output}{Input} = \frac{48}{16} = 3$$

$$\text{Gear ratio with spur gears} = \frac{Output}{Input} = \frac{25}{9} = 2.78$$

New Torques on the steering arms

$$T_{out(upper)} = (106\ \text{lb•in})(3) = 318\ \text{lb•in}$$

$$T_{out(lower)} = (106\ \text{lb•in})(2.78) = 294.44\ \text{lb•in}$$

New speed of the steering arms

$$\text{UCA speed} = \frac{50\ RPM}{3} = 16.6\ RPM$$

$$\text{LCA speed} = \frac{50\ RPM}{2.78} = 17.99\ RPM$$

Figure 61: *Calculations for the new gear ratios (Gear Ratios, n.d.).*

Appendix E - Finite Element Analysis

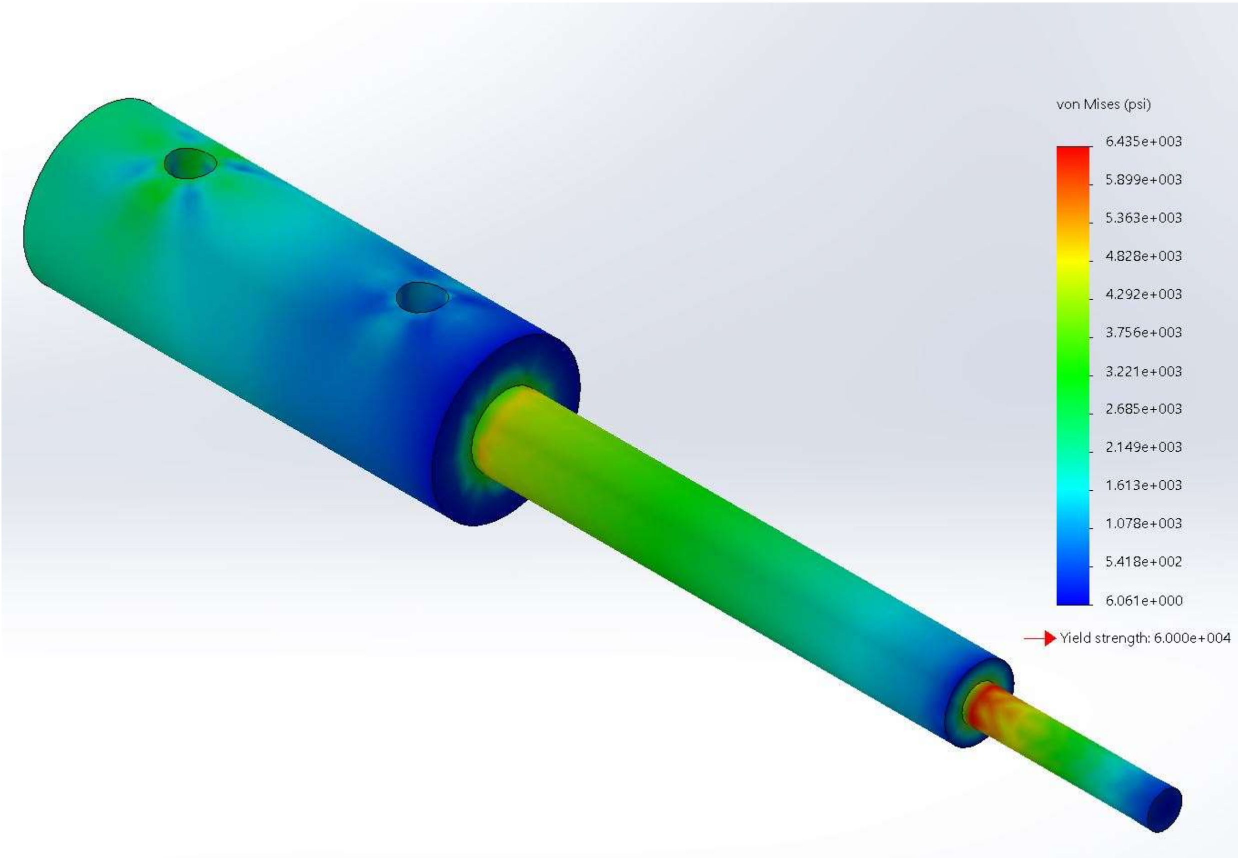


Figure 62: *FEA on the current upper shaft.*

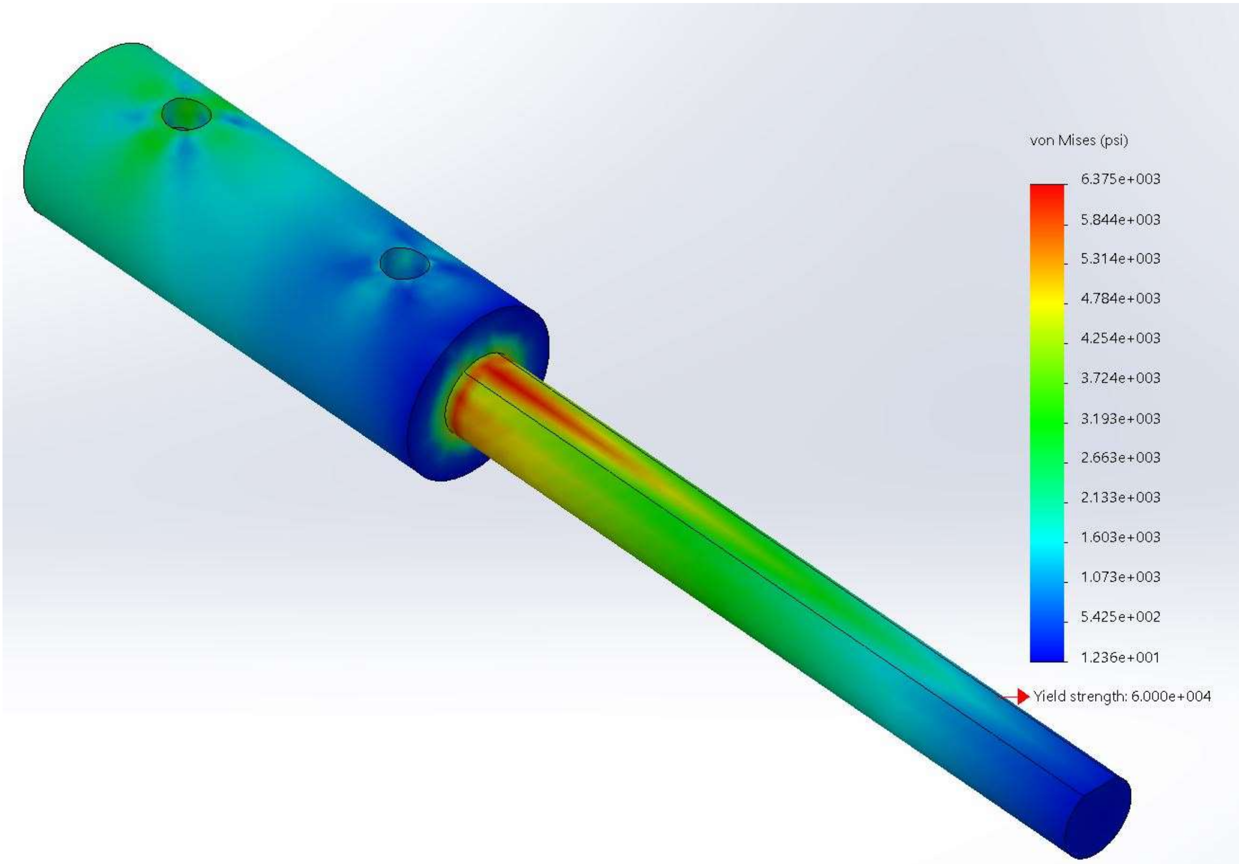


Figure 63: *FEA on the new upper shaft.*

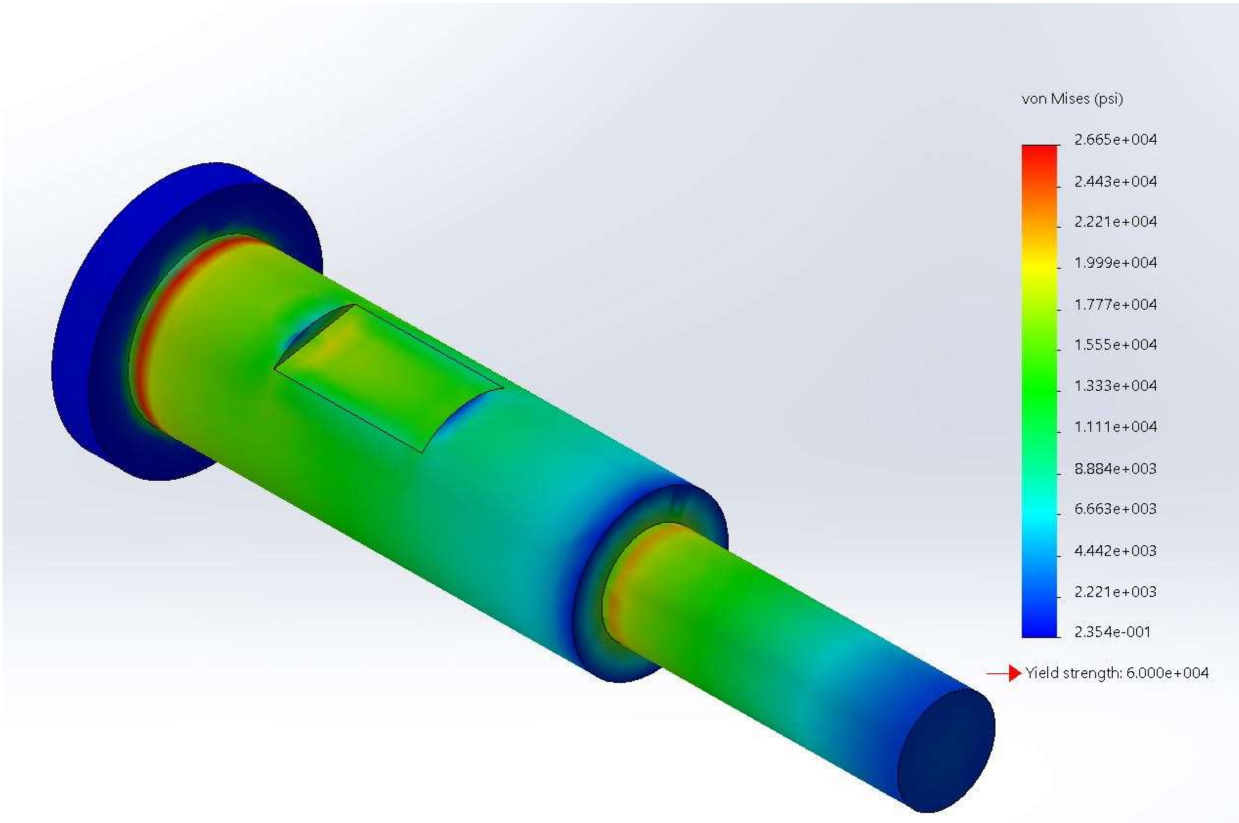


Figure 64: *FEA on the current bottom shaft.*

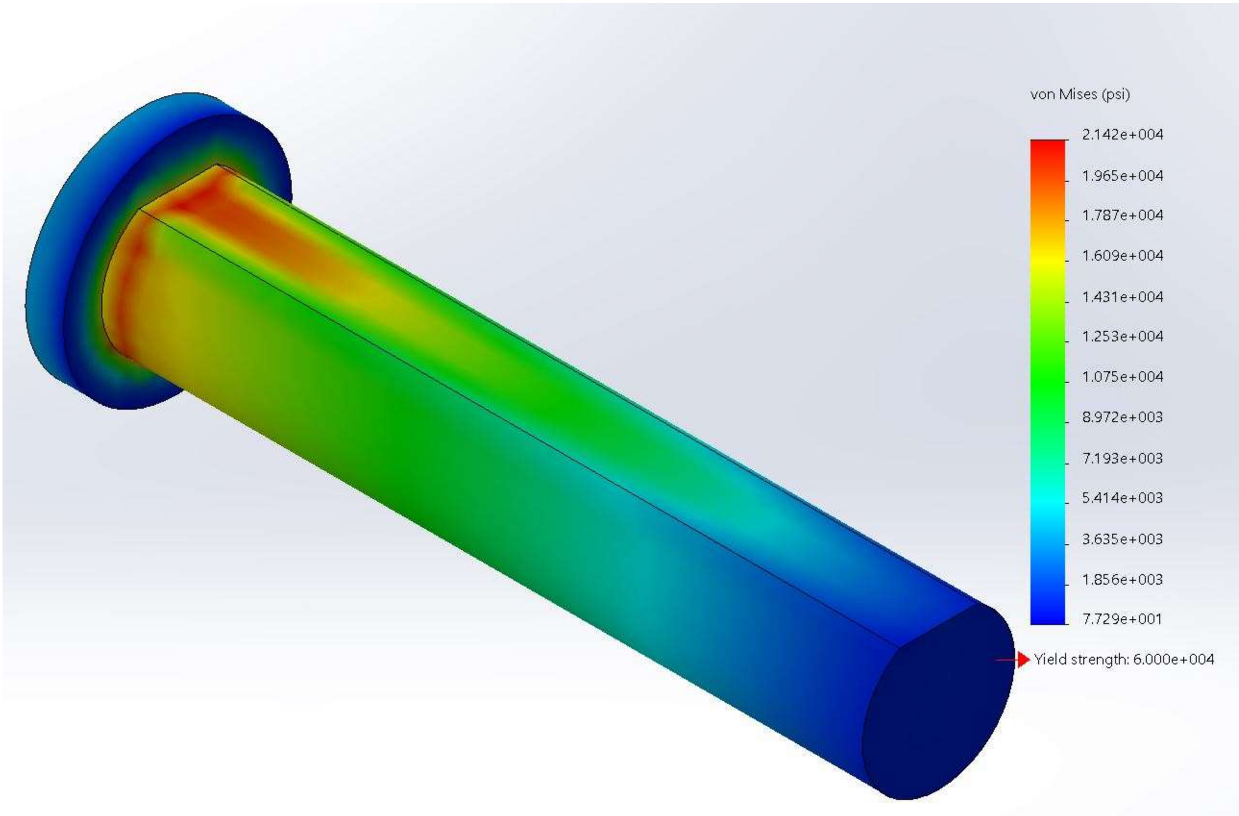


Figure 65: *FEA on the new bottom shaft.*

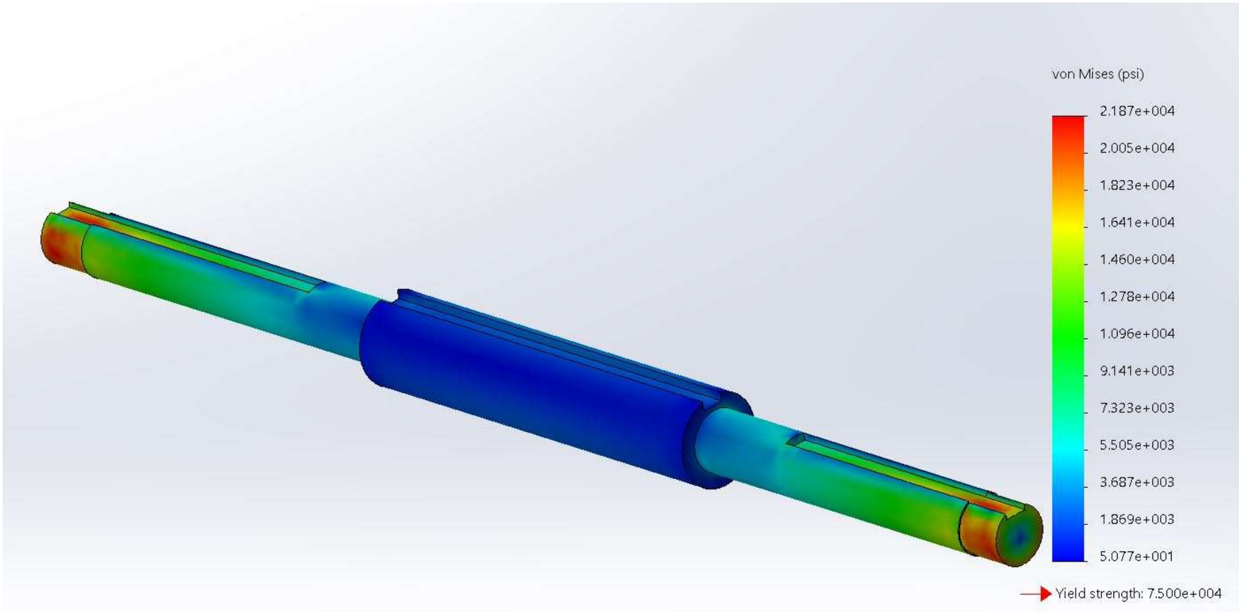


Figure 66: *FEA on the driver shaft.*

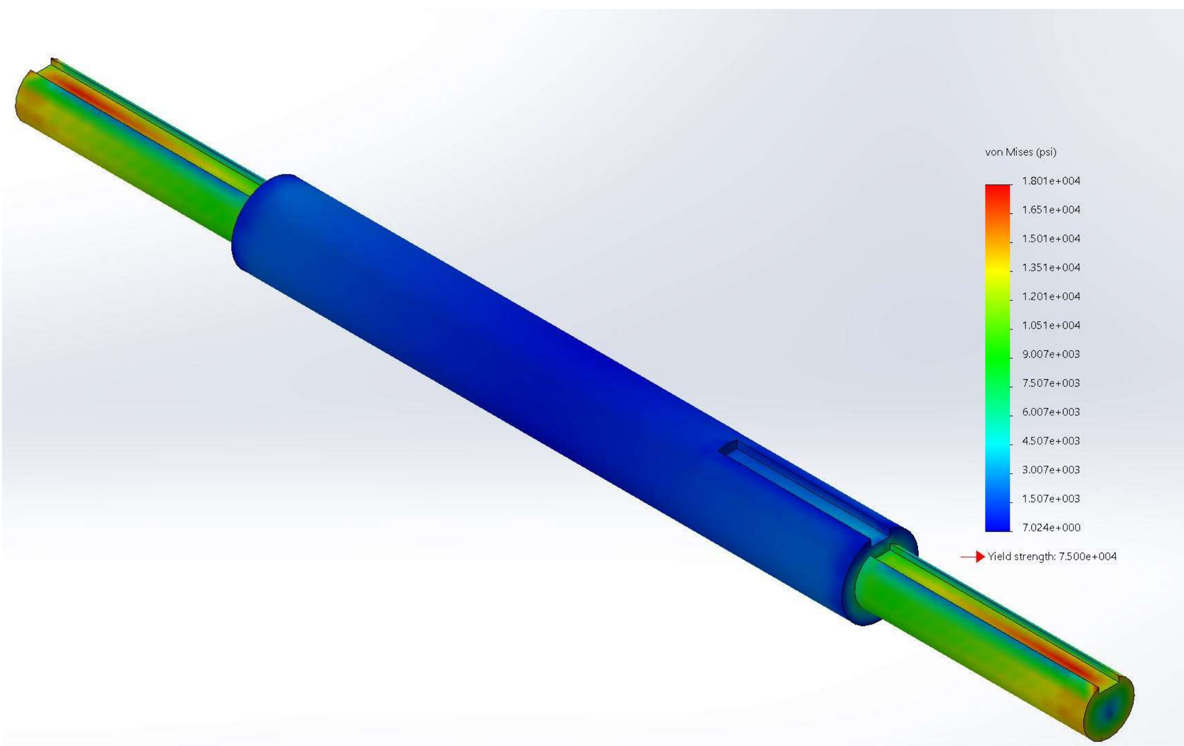


Figure 67: *FEA on the new driver shaft.*

SPARTAN SUPERWAY

Appendix F - Flow Chart

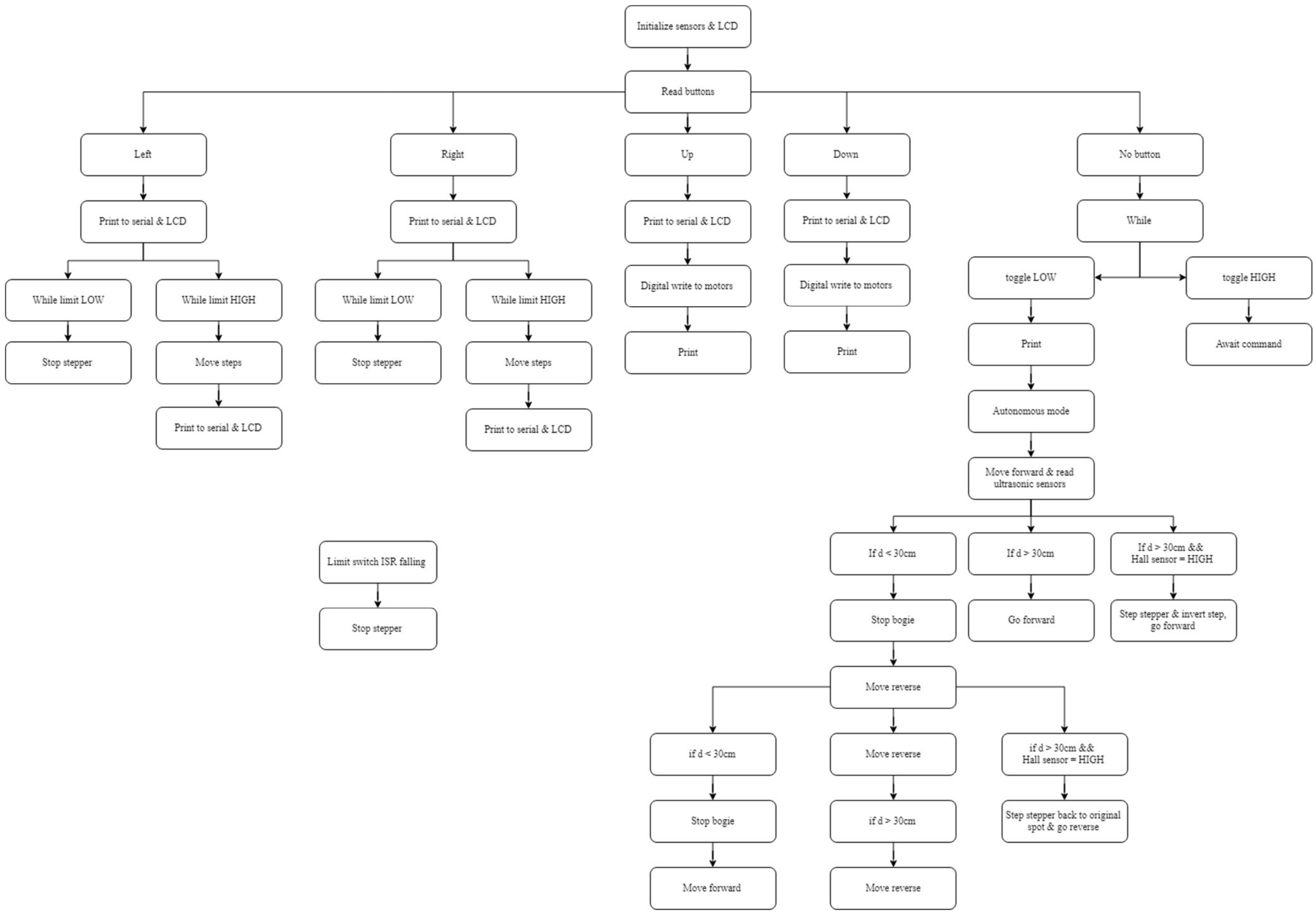


Figure 68: *Master block diagram.*

Appendix G - Arduino Code

Master Code

```
/*****************************************************
  Half Scale Bogie and Controls
  2018-2019

  Written by: Oscar Balvaneda
*****************************************************/
#include <Wire.h>
#include <LiquidCrystal.h>
#include <SoftwareSerial.h>
#include <Stepper.h>
#include <NewPing.h>



// select the pins used on the LCD panel
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);

// define some values used by the panel, buttons, and stepper
int lcd_key     = 0;
int adc_key_in  = 0;
int sensorvalue = 0;

#define TRIGGER_PIN1  40  // Arduino pin tied to trigger pin on the ultrasonic sensor.
#define ECHO_PIN1     42  // Arduino pin tied to echo pin on the ultrasonic sensor.
#define TRIGGER_PIN2  41  // Arduino pin tied to trigger pin on the ultrasonic sensor.
#define ECHO_PIN2     43  // Arduino pin tied to echo pin on the ultrasonic sensor.
#define MAX_DISTANCE 200 // Maximum distance we want to ping for (in centimeters). Maximum sensor distance is rated at 400-500cm.

NewPing sonar(TRIGGER_PIN1, ECHO_PIN1, MAX_DISTANCE); // NewPing setup of pins and maximum distance.
NewPing sonar2(TRIGGER_PIN2, ECHO_PIN2, MAX_DISTANCE); // NewPing setup of pins and maximum distance.
```

```
#define btnRIGHT  0
#define btnUP     1
#define btnDOWN   2
#define btnLEFT   3
#define btnSELECT 4
#define btnNONE   5
const byte limitL = 21;      //attachInterrupt pins can only be 2, 3, 18, 19, 20, 21
const byte limitR = 20;

#define pulse1 22            //Color brown on breadboard - stepper 1 on bogie 1
#define stprDIR1 23              //Color orange on breadboard - stepper 1 on bogie 1
#define pulse2 24            //Stepper 2 on bogie 2
#define stprDIR2 25              //Stepper 2 on bogie 2

#define autoSwitch 53      //Autonomous mode switch

#define pwm1 30           //Bogie motor 1 YELLOW
#define dirA1 31          //ORANGE
#define dirA2 32          //GREEN
#define pwm2 33           //Bogie motor 2 BLUE
#define dirB1 34          //PURPLE
#define dirB2 35          //GREY

#define hallF 44
#define hallR 45

int test = 6000;            //The two values for the stepping size
int test2 = -7000;
int fast = 20;              //The speed the stepper will move at
int onerev = 7200;            //The steps for one revolution
int STOP = 0;


int hallturn = 6500;          // The only one being used right now is hallturn when going forward.
int hallflip1 = 6500;         // It is overwritten to be negative after being triggered going forward
int hallflip2 = 6500;

Stepper stpr1(onerev, pulse1, stprDIR1);      //Define number of staps for a full rev
```

SPARTAN SUPERWAY

```
Stepper stpr2(onerev, pulse2, stprDIR2);



int read_LCD_buttons()
{
  adc_key_in = analogRead(A0);      // read the value from the sensor
  if (adc_key_in > 1000) return btnNONE;
  if (adc_key_in < 850 && adc_key_in > 600)  return btnSELECT;
  if (adc_key_in < 600 && adc_key_in > 408)  return btnLEFT;
  if (adc_key_in >= 0 && adc_key_in < 50)      return btnRIGHT;
  if (adc_key_in < 320 && adc_key_in > 253)  return btnDOWN;
  if (adc_key_in < 150 && adc_key_in > 100)  return btnUP;
  return btnNONE;  // when all others fail, return this...
}


//----------------------------------------------------------------------------------------------------------------
void setup()
{
  //LCD
  lcd.begin(16, 2);
  lcd.setCursor(0, 0);
  lcd.print("Spartan Superway");
  lcd.setCursor(3, 1);
  lcd.print("2019");
  delay(1000);
  lcd.clear();
  delay(100);

  //Limit Switches
  attachInterrupt(digitalPinToInterrupt(limitL), isr, FALLING);
  attachInterrupt(digitalPinToInterrupt(limitR), isr, FALLING);
  pinMode(limitL, INPUT_PULLUP);
  pinMode(limitR, INPUT_PULLUP);


  //Steering
```

```
  pinMode(stprDIR1, OUTPUT);  // CW- pin - direction
  pinMode(pulse1, OUTPUT);  // CP- pin - pulse1/step
  pinMode(stprDIR2, OUTPUT);  // CW- pin - direction
  pinMode(pulse2, OUTPUT);  // CP- pin - pulse1/step
  //Steering sensors
  pinMode(hallF, INPUT);
  pinMode(hallR, INPUT);



  //Serial
  Serial.begin(115200); // Open serial monitor at 115200 baud to see ping results.

  //Powertrain
  pinMode(dirA1, OUTPUT);   //motor 1
  pinMode(dirA2, OUTPUT);
  pinMode(pwm1, OUTPUT);
  pinMode(dirB1, OUTPUT);   //motor 2
  pinMode(dirB2, OUTPUT);
  pinMode(pwm2, OUTPUT);

  //Autonomous Mode
  pinMode(autoSwitch, INPUT_PULLUP);
}


//------------------------------------------------------------------------------------------------------------ -
void loop() {

  //Serial.println(adc_key_in);
  lcd.setCursor(0, 0);
  delay(1000);// read the value from the sensor
  lcd_key = read_LCD_buttons();     // read the buttons
  Serial.println("Initialized");

  switch (lcd_key)                // depending on which button was pushed, we perform an action
  {
    case btnSELECT:
```

```
  {
   Serial.println(adc_key_in);
   delay(100);
   break;
  }
 case btnLEFT:
  {
   Serial.println(adc_key_in);
   lcd.clear();
   lcd.print("Turning Left");
   leftlimit();
   delay(100);
   break;
  }
 case btnRIGHT:
  {
   Serial.println(adc_key_in);
   lcd.clear();
   lcd.print("Turning Right");
   rightlimit();
   delay(100);
   break;
  }
 case btnUP:
  {
   Serial.print(adc_key_in);
   lcd.clear();
   lcd.print("Going Forward");
   forward();
   break;
  }
 case btnDOWN:
  {
   Serial.print(adc_key_in);
   lcd.clear();
   lcd.print("Going Backwards");
   backwards();
   break;
```

```
    }
   case btnNONE:
    {
     while (int x = digitalRead(autoSwitch)  == LOW) {
      lcd.clear();
      lcd.print("Autonomous Mode");
      Serial.println("Autonomous Mode");
      autonomousMode();
      break;
     }
     Serial.println("Awaiting Command");
     lcd.print("Awaiting command");
    }
 }



 // create a switch case to run this :          autonomousMode();

}

void left() {   //unused rn
 stpr1.setSpeed(fast);
 stpr1.step(test);
 Serial.println(test);
}



void right() {    //unused rn
 stpr1.setSpeed(fast);
 stpr1.step(test2);
 Serial.println(test2);
 delay(1000);

}
```

```
void leftlimit() {
 int l = 0;
 l = digitalRead(limitL);
 while (l == LOW) {
  stpr1.setSpeed(STOP);
  stpr1.step(STOP);
  Serial.println("Steering left limit triggered, stop turning");
  break;
 }
 while (l == HIGH) {
  Serial.print("Turning Left for: \t \t");
  Serial.print(test);
  Serial.println(" steps");
  stpr1.setSpeed(fast);
  stpr1.step(test);

  // this is to alternate directions for autonomous mode test = ~test;
  break;
 }
}

void rightlimit() {
 int j = 0;
 j = digitalRead(limitR);
 while (j == LOW) {
  stpr1.setSpeed(STOP);
  stpr1.step(STOP);
  Serial.print("Steering right limit triggered, stop turning");
  break;
 }
 while (j == HIGH) {
  Serial.print("Turning Right for: \t \t");
  Serial.print(test2);
  Serial.println(" steps");
  stpr1.setSpeed(fast);
  stpr1.step(test2);

  break;
```

```
  }
}

void monitorFront() {

  //Ultrasonic sensor begin reading
  int cm = sonar.ping_cm();

  //Look for hall sensor to turn
  int p = digitalRead(hallF);

  if (cm < 30) {
    // stop motor
    stopBogie();
    Serial.println("\t \t STOP BOGIE");
    delay(1000);
  }
  else if (cm >= 30 && cm < 80) {
    forward();
    Serial.println("\t \t \t \t \t \t SLOW DOWN");
  }
  else if (cm > 30 && p == HIGH) {
    forward();
    Serial.println("Turning Left");
    stpr1.step(hallturn);
    hallturn = !hallturn;
  }
  else {
    // keep motor running analogWrite(ENA,SPEED);
    forward();
    Serial.println("\t \t \t \t \t \t GOOD TO GO");
  }
}


void monitorRear() {

  //Ultrasonic sensor begin reading
```

```
  int cm1 = sonar2.ping_cm();

  //Look for hall sensor
  int k = digitalRead(hallR);

  if (cm1 < 30) {
    // stop motor analogWrite(ENA,0);
    stopBogie();
    stpr1.step(hallturn);
    Serial.println("\t \t STOP BOGIE");
  }
  else if (cm1 >= 30 && cm1 < 80) {
    backwards();
    Serial.println("\t \t \t \t \t \t SLOW DOWN");
  }
  else {
    // keep motor running analogWrite(ENA,SPEED);
    backwards();
    Serial.println("\t \t \t \t \t \t GOOD TO GO");
  }
}


void forward() {
  digitalWrite(dirA1, LOW);
  digitalWrite(dirA2, HIGH);
  analogWrite(pwm1, 200);
  digitalWrite(dirB1, LOW );
  digitalWrite(dirB2, HIGH);
  analogWrite(pwm2, 200);
  Serial.println("\t \t \t Moving Forward");

}

void backwards() {
  digitalWrite(dirA1, HIGH);
  digitalWrite(dirA2, HIGH);
  analogWrite(pwm1, 200);
```

```
  digitalWrite(dirB1, HIGH);
  digitalWrite(dirB2, HIGH);
  analogWrite(pwm2, 200);
  Serial.println("\t \t \t Moving Backwards");
}

void stopBogie() {
  digitalWrite(pwm1, LOW);
  digitalWrite(pwm2, LOW);
  Serial.print("\t \t \t Bogie Stopped");
}

void autonomousMode() {
  monitorFront();
  monitorRear();
}

void isr() {
  stpr1.step(STOP);
  Serial.print("ISR");
}
```

Velocity Code

```
/*******************************************************
 Half Scale Bogie and Controls
 2018- 2019

Written by: Hector Gomez
*******************************************************/

#include <LiquidCrystal.h>

// select the pins used on the LCD panel
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);

// define some values used by the panel and buttons
int lcd_key     = 0;
int adc_key_in  = 0;
#define btnRIGHT  0
```

```
#define btnUP     1
#define btnDOWN   2
#define btnLEFT   3
#define btnSELECT 4
#define btnNONE   5

#define PIN_HALL   2

float radius = 3.25; // (inches)
float circumference;
float distance = 0;
float velocity = 0;

volatile byte WheelSpinning;

unsigned long startTime;
unsigned long currentTime;
unsigned long previousWheelSpinning = 0;
unsigned long currentWheelSpinning = 0;

void setup() {
  pinMode(PIN_HALL, INPUT_PULLUP);
  circumference = 2*3.14*radius;
  attachInterrupt(digitalPinToInterrupt(PIN_HALL), magnet, RISING);

  lcd.begin(16, 2);
  lcd.setCursor(0,0);
  lcd.print("SpartanSuperway");
  lcd.setCursor(3,1);
  lcd.print("2019");
  delay(1000);
  lcd.clear();
  delay(1000);

  startTime = millis();
  previousWheelSpinning = WheelSpinning;
}
void loop() {

  currentTime = millis();
  currentWheelSpinning = WheelSpinning - previousWheelSpinning;

  if(currentTime - startTime >= 500 && currentTime - startTime < 2000 ){
```

```
  velocity = (circumference * currentWheelSpinning *1000)/(currentTime - startTime);
 // distance+= circumference;
  lcd.setCursor(0,0);
  lcd.print("Speed= ");
  lcd.println(String(velocity) + " in/s");
 // lcd.setCursor(0,0);
  //lcd.print("distance= ");
  //lcd.println((distance) "in");

  previousWheelSpinning = WheelSpinning;
  startTime = millis();

  } else if ( currentTime - startTime >= 2000) {
    velocity = 0;
    startTime = millis();
  }
}
void magnet() {

  WheelSpinning++;
}
```

Wireless Communication code

```
/*****************************************************
 Half Scale Bogie and Controls
 2018- 2019

Written by: Karen Tran
*****************************************************/

Transmitter Code to turn on an LED:

#include <SPI.h>
#include "RF24.h"
const int led = 43;
int ledState = 0;

RF24 radio (7, 8); // CE, CSN. “radio” can be renamed to anything
byte address[6] = {"0"};
```

```
char text[100] = "Hi";


void setup() {
  Serial.begin (9600);
  delay(1000);
  radio.begin();
  radio.setChannel(115); // set frequency to channel 115
  radio.setPALevel(RF24_PA_MAX);
  radio.setDataRate (RF24_250KBPS);
  radio.openWritingPipe(address[0]);
  pinMode(led,OUTPUT);

  delay(1000);
}

void loop() {
 digitalWrite (led,HIGH); // LOW = off, HIGH = on
 ledState = digitalRead (led);
  radio.write(&text, sizeof(text));
  radio.write(&ledState, sizeof(ledState));
  Serial.print(text);
  Serial.print("\n");
  Serial.print(ledState);
  Serial.print("\n");
  delay(1000);
}
```

Receiver code to turn on an led:

```
#include <SPI.h>
#include "RF24.h"
RF24 radio(7,8);

byte address[6] = {"0"};
```

```
char text[100]= " ";
const int ledPin = 43;
const int SW1 = 2;
int ledState;

void setup() {
  Serial.begin(9600);
  delay(1000);
  radio.begin();
  radio.setChannel(115);
  radio.setPALevel(RF24_PA_MAX);
  radio.setDataRate(RF24_250KBPS);
  radio.openReadingPipe(1,address[0]);


  pinMode(ledPin,OUTPUT);
  pinMode(SW1,INPUT);

}

void loop() {
  radio.startListening();
  delay(10);
  if(radio.available()) {
    while(radio.available())
    {
      radio.read(&text, sizeof(text));
      radio.read(&ledState, sizeof(ledState));
    }

    Serial.print(text);
    Serial.print("\n");

    if(ledState == 1) {
      digitalWrite(ledPin,LOW); //LED OFF
      Serial.print("OFF");
      Serial.print("\n");
    }
```

```
    else if (ledState == 0){
      digitalWrite (ledPin,HIGH);
      Serial.print("ON");
      Serial.print("\n");
    }
    }

else{
    radio.stopListening();
    int buttonState = digitalRead(SW1);
    Serial.print(buttonState);
    Serial.print("\n");
    radio.write(&buttonState, sizeof(buttonState));
    //radio.write(&buttontext, sizeof(&buttontext);

    delay(1000);
}
  }
```

Appendix H - Engineering Drawings

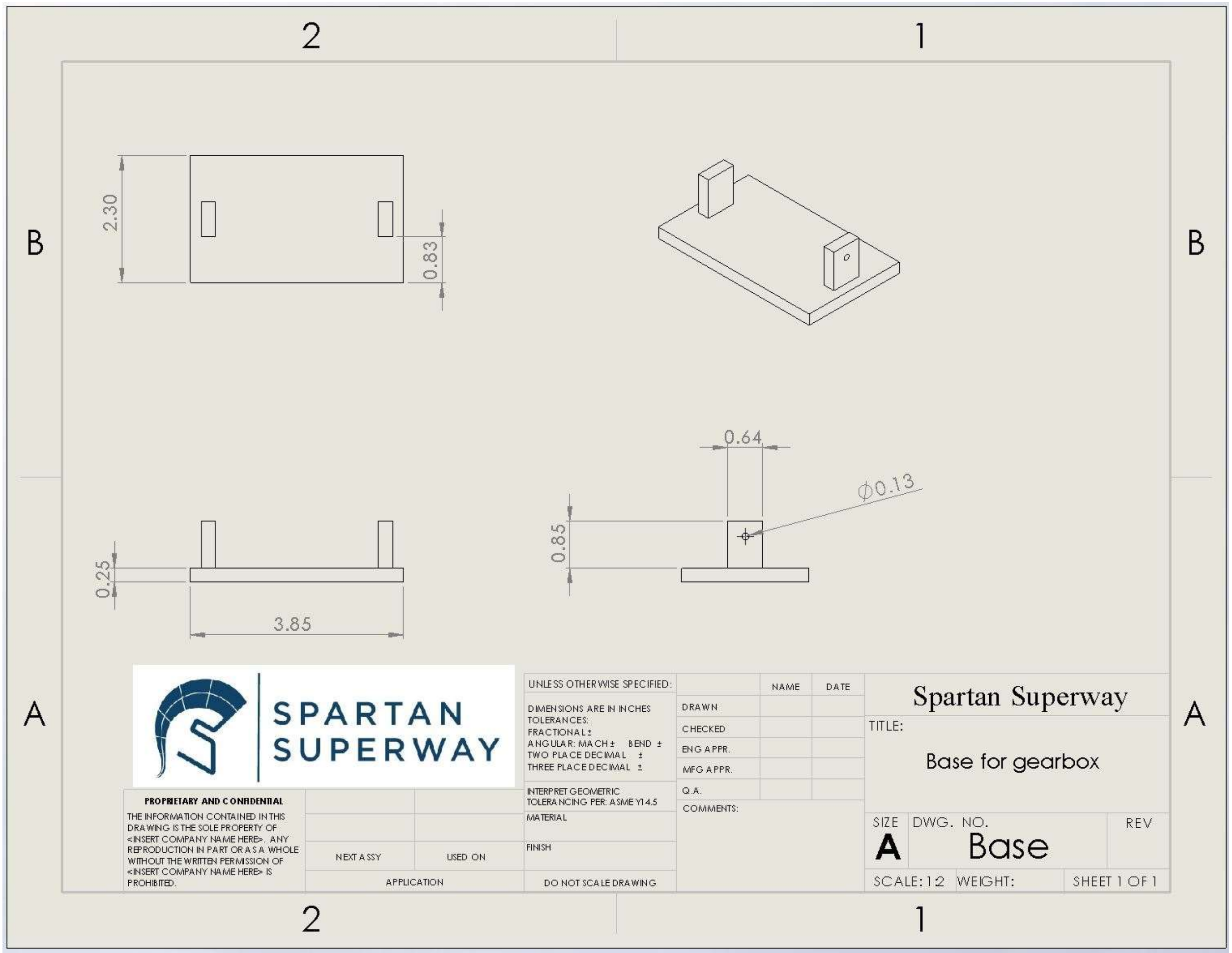


Figure 69: *The base for the gearbox.*

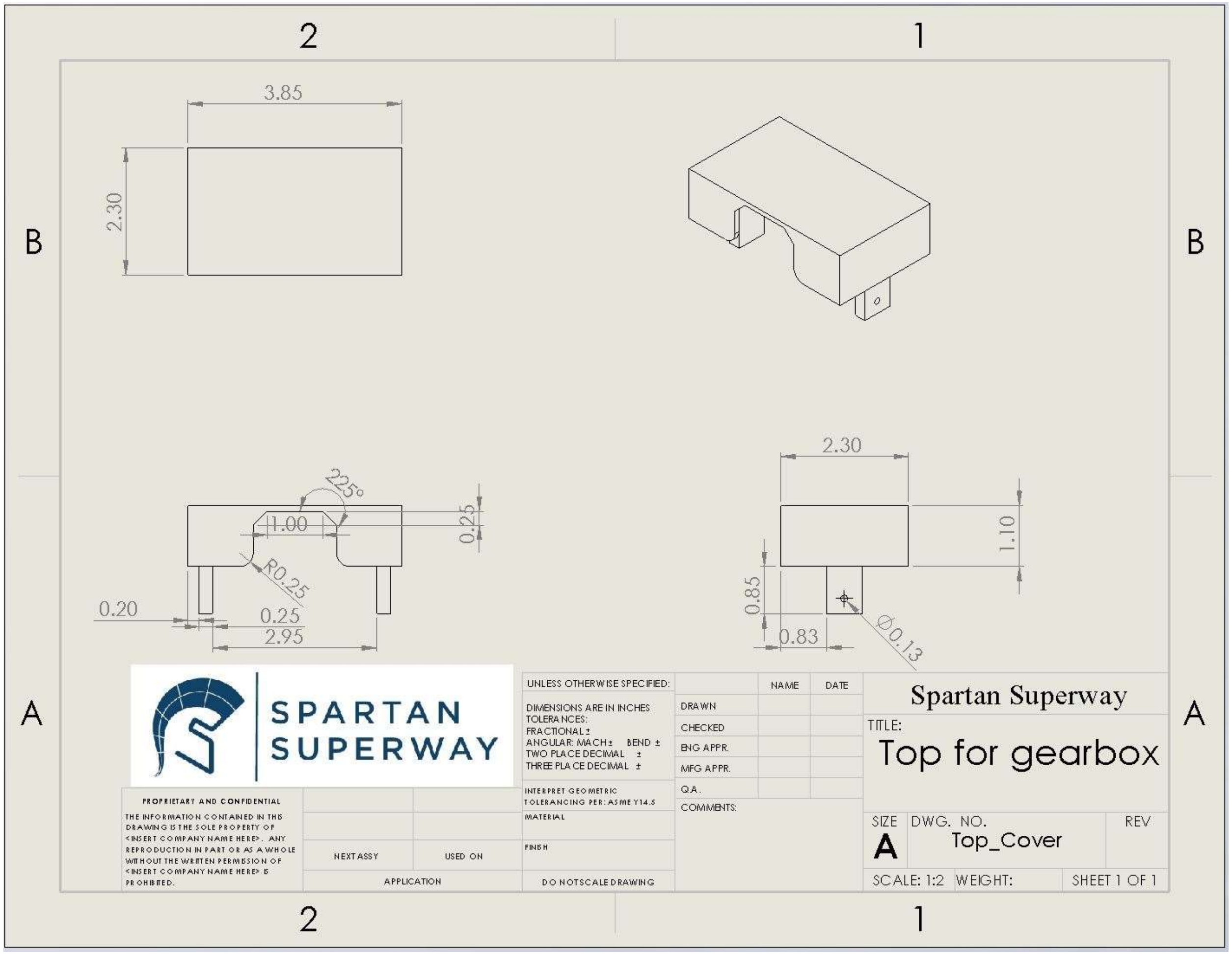


Figure 70: *The top cover for the gearbox.*

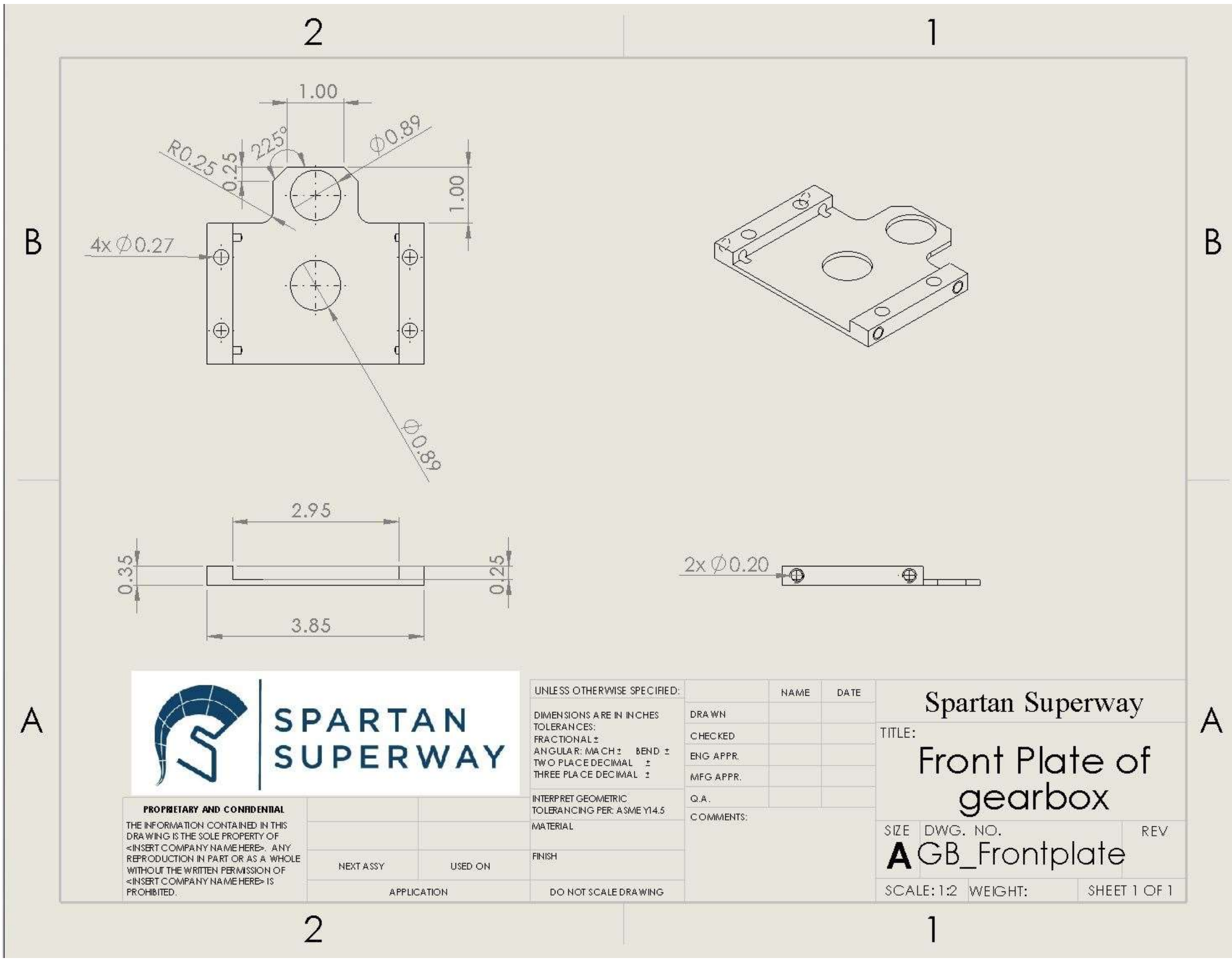


Figure 71: *The front plate for the gearbox.*

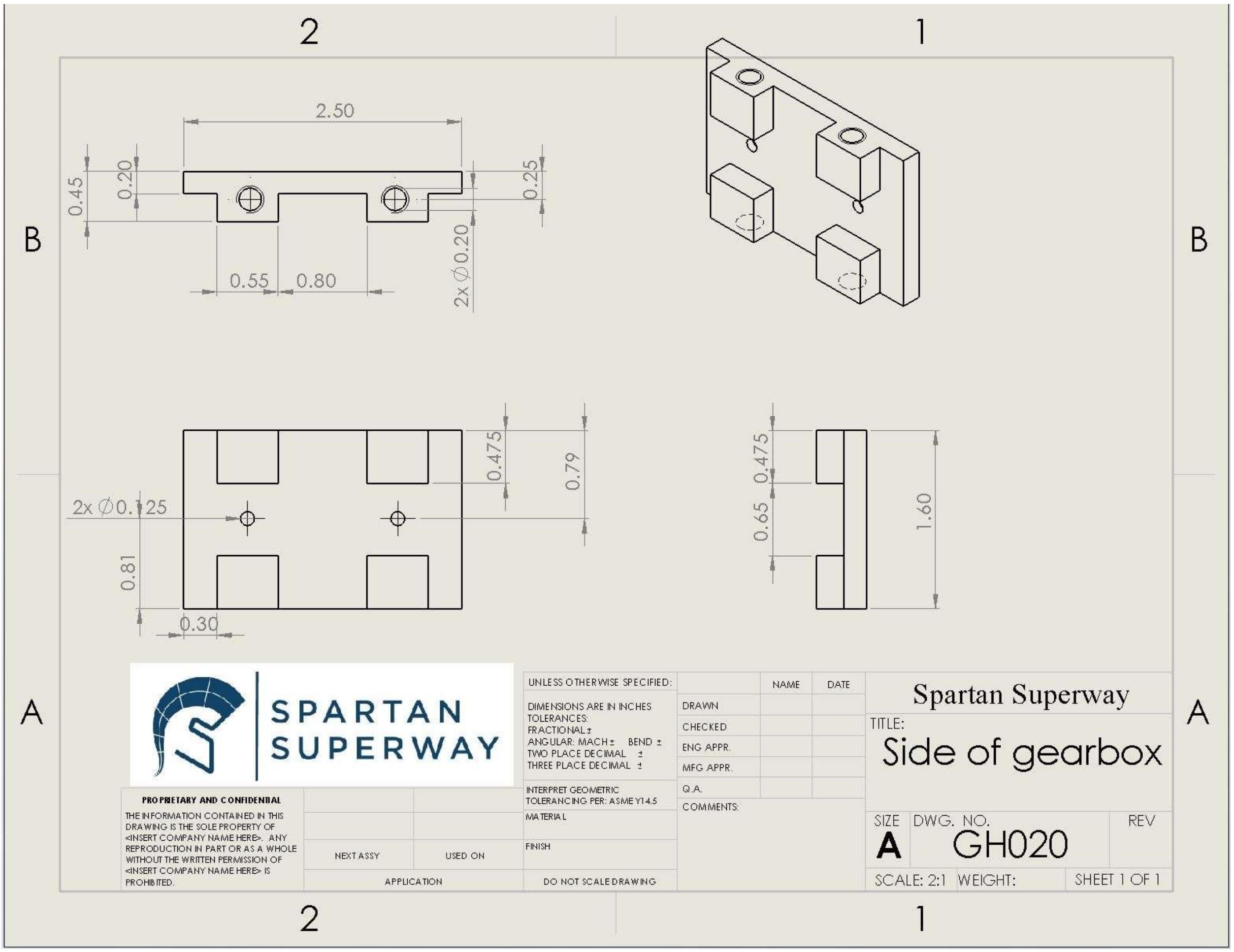


Figure 72: *The side plate for the gearbox.*

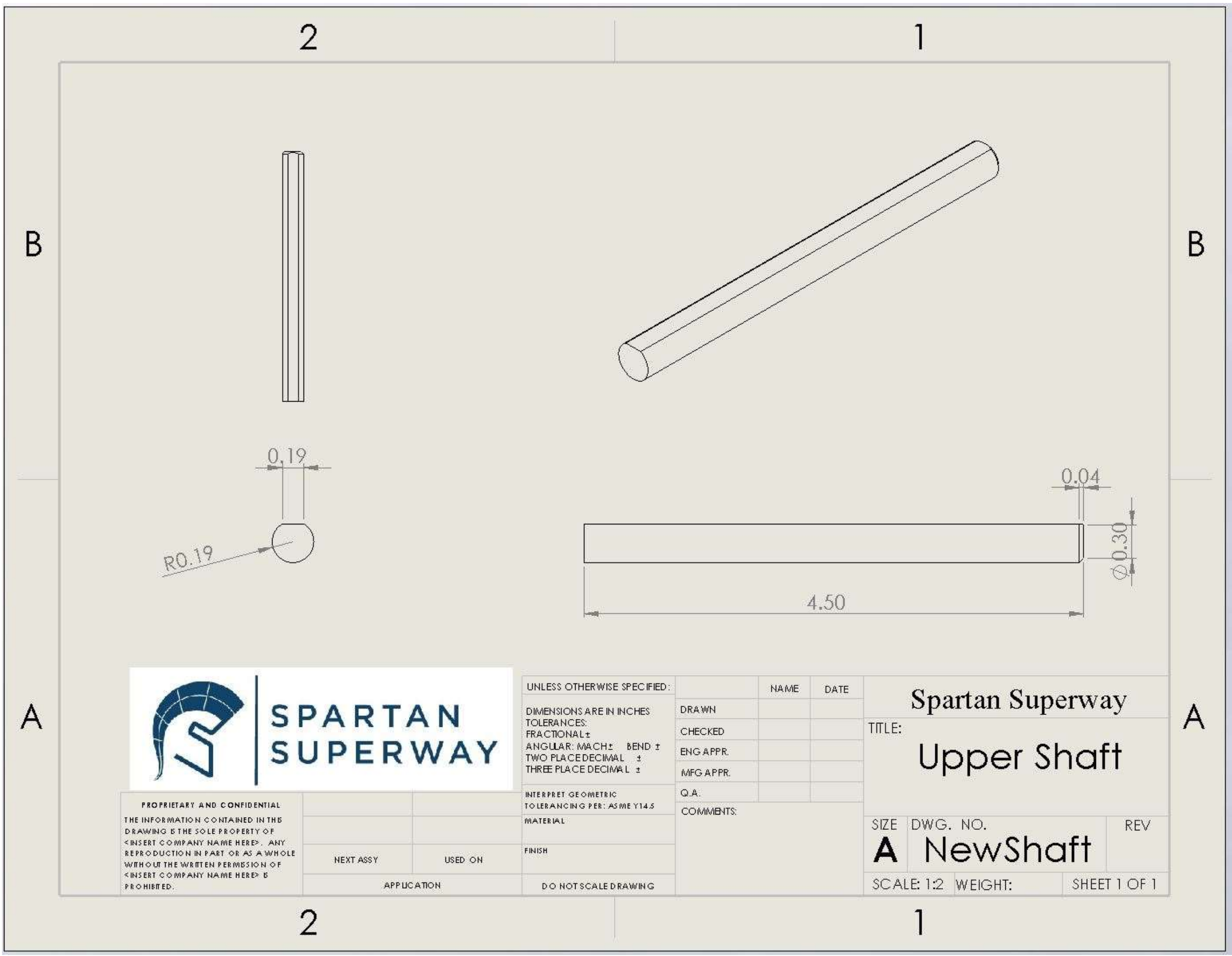


Figure 73: *The upper shaft.*

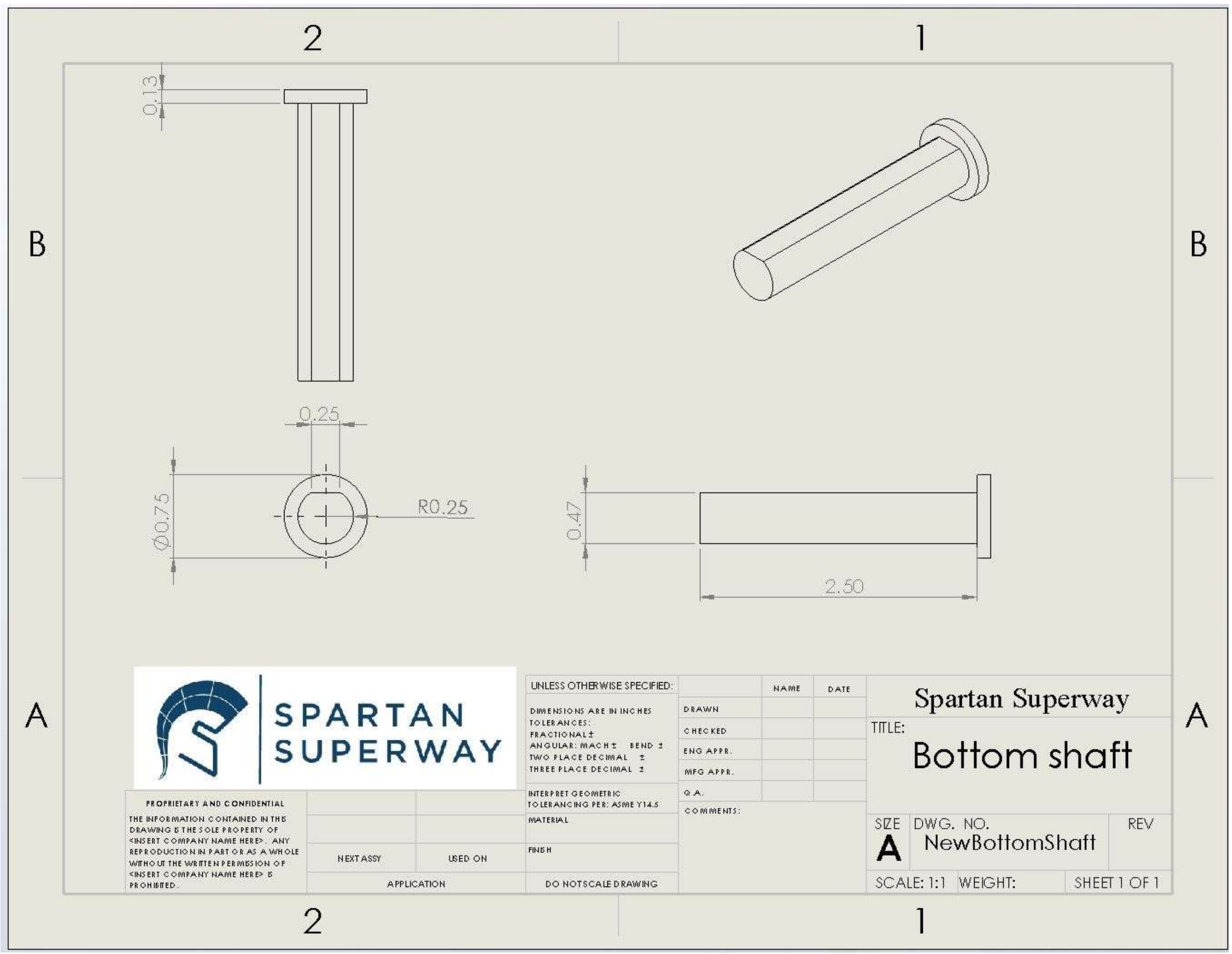


Figure 74: *The bottom shaft.*

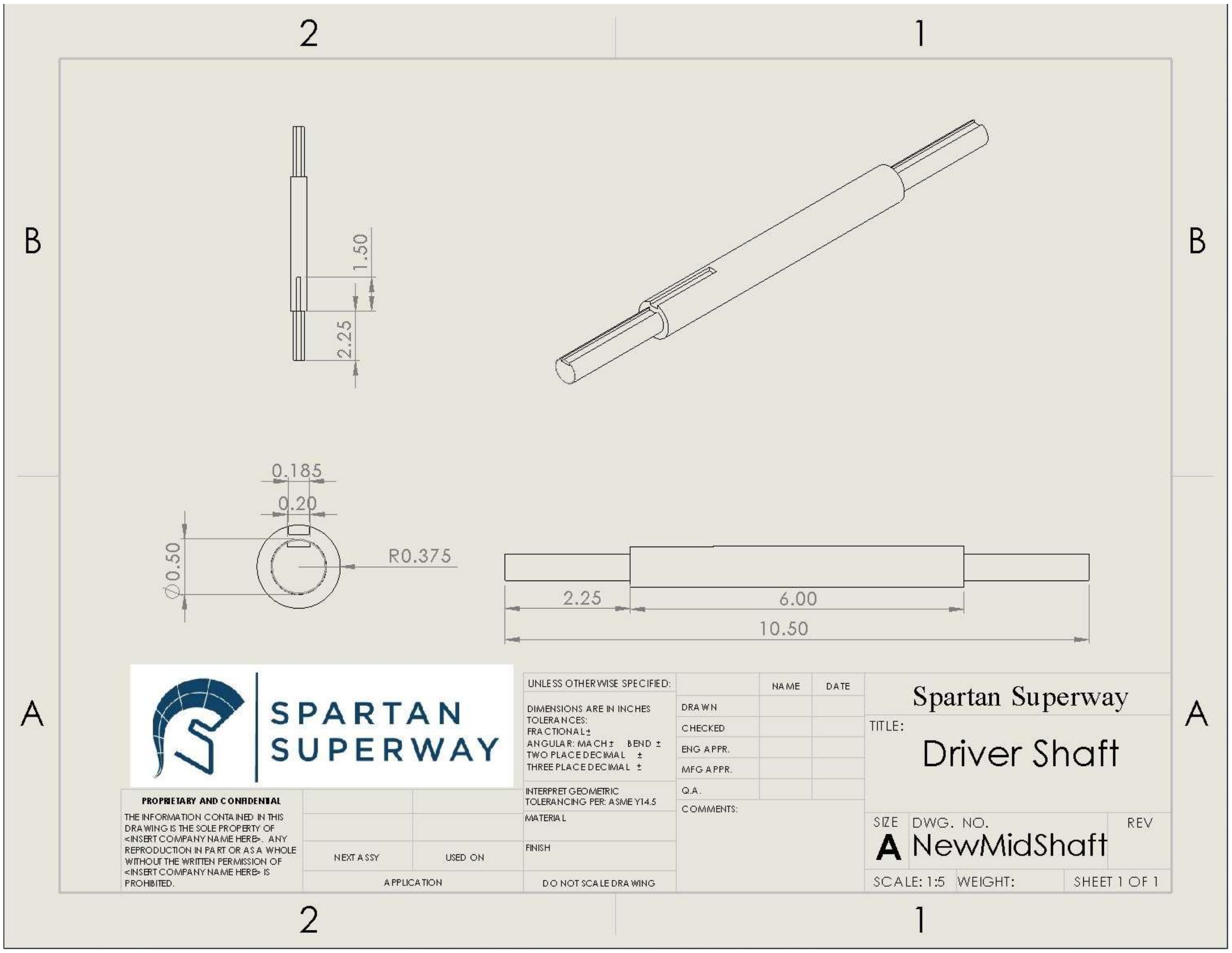


Figure 75: *The driver shaft.*

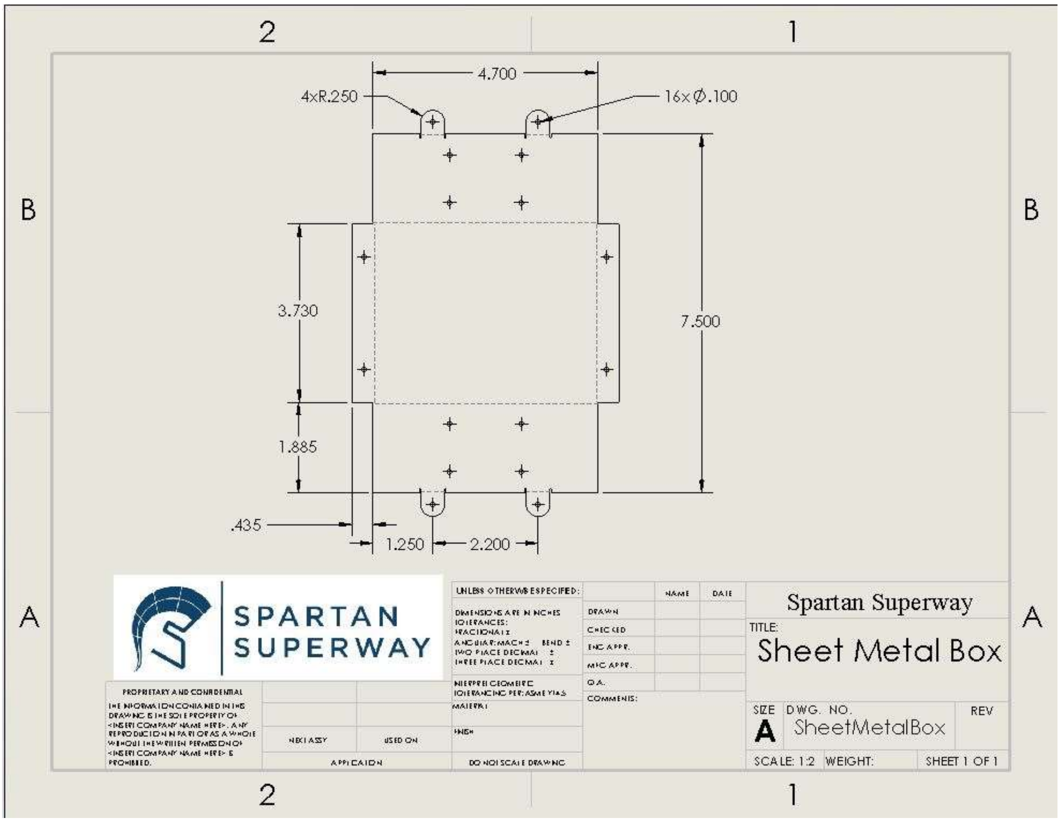


Figure 76: *The Sheet Metal Box.*

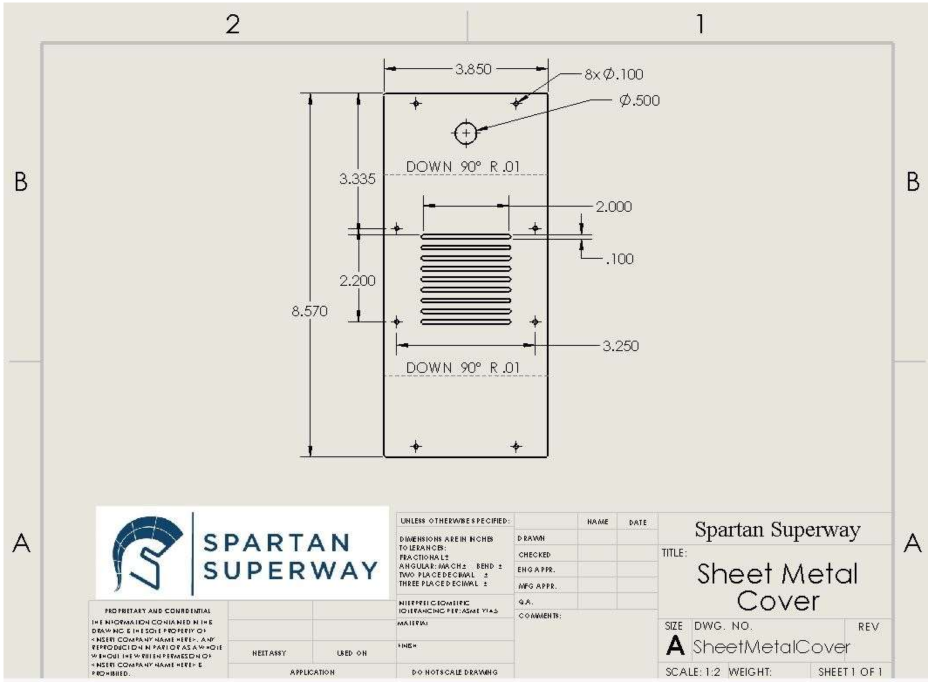


***Figure** 77: The Sheet Metal Cover.*

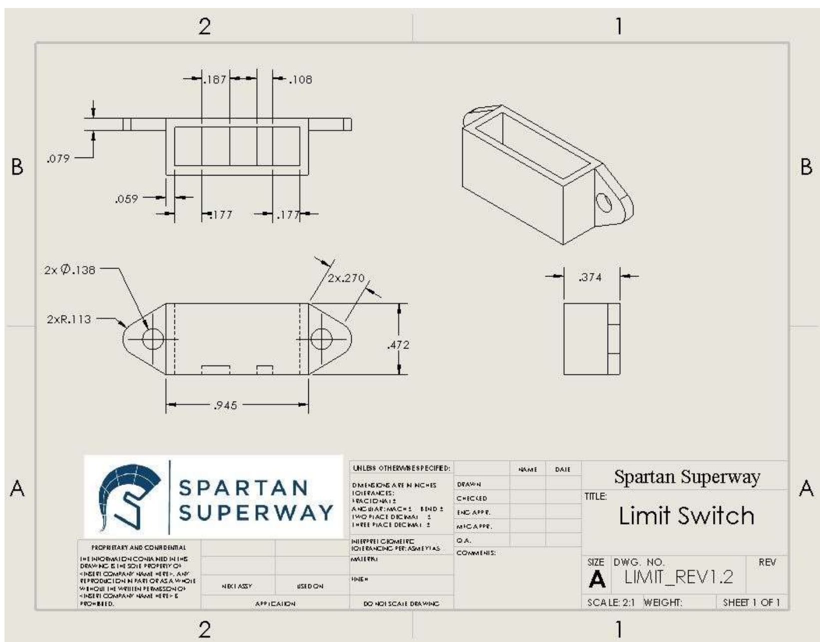


***Figure 78:** The Limit Switch.*

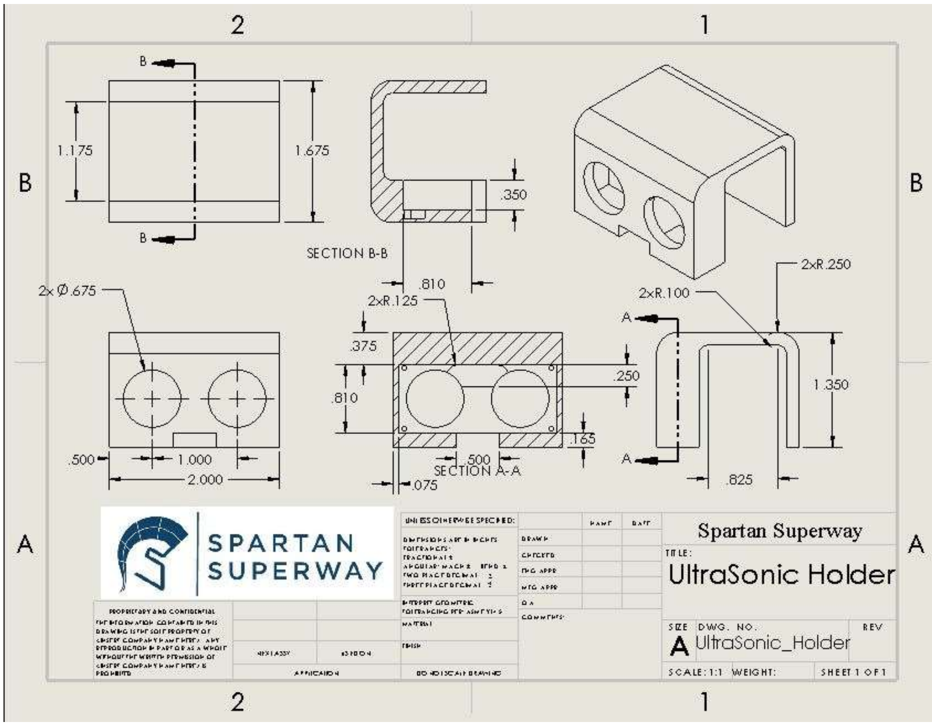


Figure 79: *Ultrasonic sensor.*

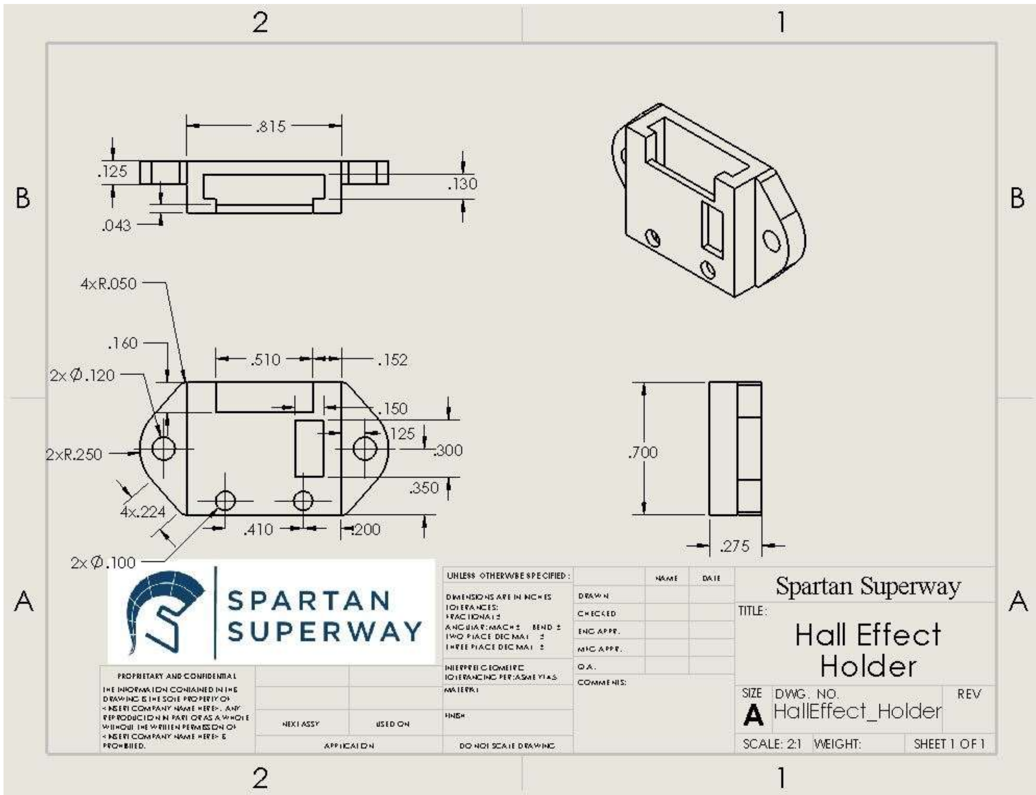


Figure 80: *Hall Effect sensor.*