

An Autonomous Personal Rapid Transportation Network:

Half-Scale Team

Greg White

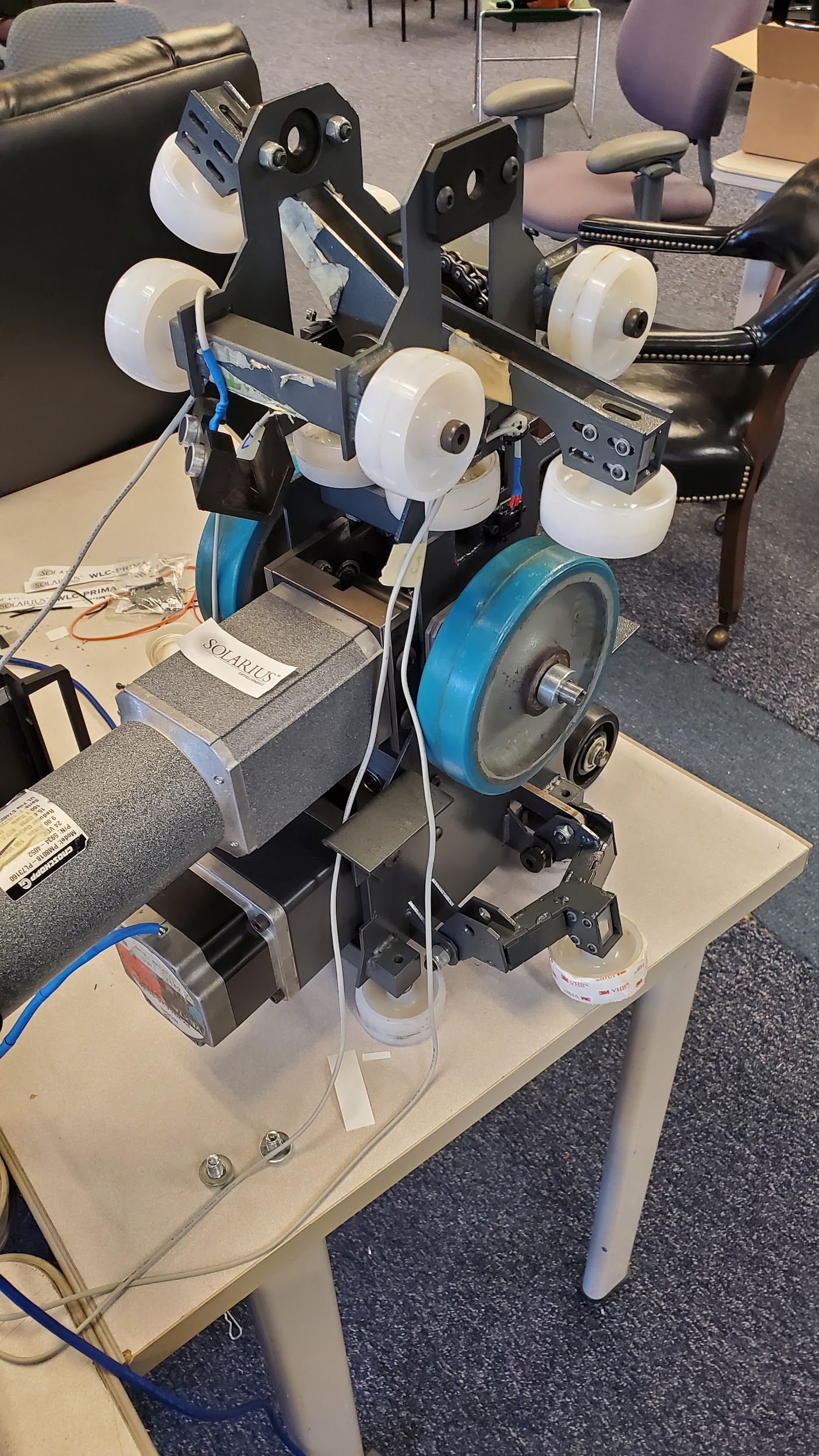
Brandon Keith Scully

Keanu Heggem

Pinqian Lin

December 14, 2019

San Jose State University



Advisor: Dr. Burford Furman

**Abstract**

The research and development of the Spartan Superway personal rapid transit (PRT) system introduce new technology and innovation to appropriately adapt to rising traffic congestion problems. Acknowledging the issues of urban societies, the Superway uses an elevated pod car which is designed to have little to no impact once implemented on public roads. The 2019-2020 half-scale team has proceeded with research contributions from the prior years while troubleshooting problems that the 2018-2019 half-scale team was unable to address on the bogie and track prototype. In order to get the bogie fully operational within the allotted time, the team has directed focus on diagnosing bugs in the system wiring and code while also addressing the drive system and non-functional switch mechanism. In order to diagnose and find a solution to the mechanical, mechatronic, and electrical errors we face; we have fully disassembled one bogie and rewired the control box to reduce the number of short circuits. At the moment, the stepper motor and switch arms do not function due to the short circuits in the control box.

The research was undertaken by the 2019-2020 half-scale team to utilize and better understand the interaction between the bogie and track prototype. Trial runs using the assembled bogie were documented in order to analyze its behavior in relation to the track. The track-vehicle interaction analysis assisted in selecting the issues most important to address while also revealing which operating conditions influence proper functionality the most. With the trial runs of the bogie revealing faulty operating characteristics in the mechanical and mechatronics aspect of the design, we have identified the track-bogie interaction parameters. Any modifications already made and future improvements are due to interactions documented when field testing the bogie on the track. Modifications to the code to simplify commands testing the bogie’s basic functionality and maneuverability have been implemented. Alterations to the pins controlling the motion of the switch arms have been made, preventing the stepper motor and linkage from becoming misaligned and locking up.

In order to complete many of the tasks we have planned while achieving the goal of the bogie fully operational on the track, we will be continuing our research into the winter break. The CAD designs for the control box housing will be completed by the end of winter break as it requires more analysis using SolidWorks. In addition to the control box housing, the team is in the process of designing a wireless controller and an onboard battery pack system that would leave the bogie free to run on the track without having to follow behind with the control box. The bogie will receive several mechanical updates as well as a redesign of many of the components connected to the switch arm and gearing.

**Acknowledgments**

We would like to thank Dr. Furman and Eric Hagstrom for supervising us throughout the semester and for proposing solutions whenever a problem arises. Additionally, we would like to thank Mr. Swenson for providing the facilities, necessary equipment, and funding to carry out this project. Furthermore, we would like to thank Oscar Balvaneda and Hector Gomez for their advice and input in carrying out our goals for the 2019-2020 Spartan Superway Half-Scale Team.

**Table of Contents**

**Executive Summary 4**

**Introduction and Project Description 6**

**Member’s Role** 7

**Objectives 8**

**Structure of Project** 8

**State of the Art Literature Review 9**

**2019-2020 Research 11**

**Design Solution 12**

Design Specifications 12

Prime Design 12

* Power Management 12
* Carriage 12
* Heat Dissipation 13
* Wireless Communication Module14

**Plans for fabrication** 15

**Conclusions and Future Work 16**

**References 17**

**Appendix A: 19**

* Bill of Materials 19
* Gantt Chart20

**Appendix B: 22**

* Figures 22

**Appendix C: 25**

* Code25

**Executive Summary**

The exponential growth of the nation has created a high demand for the expansion of transportation systems while reducing environmental impact. The byproduct of these demands is overcrowded public transportation systems that cannot meet the needs of many commuters due to rapid population expansion. Many of the transit methods readily accessible to the general public are outdated, have a limited service area, and of lower service frequency. The Superway will serve as one of the primary methods of transportation in order to meet demands placed on the transit industry to increase the efficiency of vehicles on the road. The Spartan Superway project was created by Dr. Buford Furman and engineer Ron Swenson which features a suspended personal rapid transportation system that transfers commuters from point A to B, avoiding all traffic and delays. The main focus of the 2019-2020 half-scale bogie team is to redesign several aspects of the switch mechanism and mechatronic components in order for the bogie to successfully navigate the entire track. The half-scale group is comprised of four members divided into two sub-teams which are focused on mechanical design and mechatronics and carriage design.

The 2019-2020 half-scale team is furthering work to the bogies mechatronics and mechanical design by previous half-scale teams. The functional requirements of the design are: the bogie has to successfully transverse the track and “Y” switch supporting the weight of the carriage and passengers which is about 700 pounds. The goal is to make one bogie complete the switch reliably and if the new design is functional, the design will be implemented on the second bogie. Modifications to the bogie design include the use of a keyed metal shaft and gears for the switch arm and stepper motor to correct the issue of the bogie becoming misaligned and slipping when the chain is under tension as well as an additional tooth to the gear connecting to the switch arm.

The most recent configuration of the switching links on the top portion of the bogie consists of a shift fork that is also to be redesigned to accommodate the keyed shaft of the stepper motor. Additional wheels will be added for extra support when switching tracks. The control box of the bogie will be housed in a suspended carriage which is to be controlled wirelessly via nRF24L01 module. A 36-Volt rechargeable battery with a 5200 mAh capacity will be used in order to power the bogie control system, stepper motor (12 Volts) and drive motor (24 Volts) using a voltage divider. The keyed upper and lower shafts, as well as the keyed gears, are going to be created from carbon steel with a yield strength ranging from 60,000 to 75,000 psi.

For the mechatronics portion of the design, Researching suitable parts for wireless control of the bogie, the nRF24L01+ 2.4GHz Wireless Transceiver Module was chosen. The controller will feature an LCD screen that serves as a voltage and battery level indicator while also displaying commands. As the control box is already using Arduino controls, the controller will use a secondary Arduino as the source of controls.

The bogie has not been performing as expected and the majority of the first semester was spent troubleshooting the code to look for bugs that are causing the second stepper motor driver to not function. Upon diagnosing the issues in the code the control box shorted out and both stepper motors are currently inoperable. To resolve this issue, tests of the outputs of all the components in the system using a multimeter are going to be done throughout the winter break. Shafts need to be ordered for the switch mechanisms control arms and they will need to be correctly dimensioned in order to pair up with the keyed gears

Collectively, the half-scale bogie team must prioritize troubleshooting the control boxes stepper motor issues to complete our end of year goal of having the bogie successfully traverse the track and switch between them. The control box burning out and not being operational throughout the semester has shifted our focus from the switch mechanism to rewiring the control box and adding wireless control. Design changes were based on the instructor’s recommendations, and last year’s project reports. The team will complete the control box carriage and the diagnosis of issues with the stepper motor driver through the winter break in order to remain on track for next semester's goals. During the spring 2020 semester, the team will be focusing on getting the bogie to switch properly assuming that the carriage design and wireless functionality will already be complete by the end of the winter break.

**Introduction and Project Description**

Population growth has led to considerable congestion in urban areas. Cities are growing at a rate that existing infrastructure cannot handle, and as such these populated areas are becoming overwhelmed with vehicles. This is not a new trend. In 1984 urban freeway congestion was estimated to cause 1.2 billion hours of delay annually (Lindley, 1987). This is making commutes take longer and use more gasoline, resulting in more fuel consumption in everyday life. Furthermore, these same vehicles that are congesting the roadways in cities are also emitting toxic gases. It is well known that petroleum-based vehicular transport is responsible for a large number of greenhouse gas emissions (Kenworthy, 2003). The amount of vehicular transportation on the streets results in a massive flux of harmful vapors into the atmosphere. This issue is exasperated by the aforementioned growth in traffic congestion in urban areas. This combination causes a steep rise in greenhouse gas emissions in these places.

Urban areas often have public transportation systems to both ease congestion in cities and provide people with a cheap means to move about everyday life without a personal vehicle. However, these transit systems rarely can stand up to the task. Constant population growth means that the systems are outdated within a few decades, and can be expensive to expand further. The primary of these systems, public buses, can get caught up in the congestion as well. Their more expensive and more private counterparts in taxis suffer from the same issue. While some cities manage to make these systems dependent on energy sources that are less emissive than petroleum, the time that they can spend bound in traffic is still an unsolved issue. Other transit systems such as the BART and the New York subway system can avoid the surface level traffic issue but exist underground and are therefore expensive to build and maintain. (Reilly, 2000). They are also more limited by geology than aboveground systems would be. This often leads to cost overruns and significant delays. Furthermore, tunneling can be very difficult in urban areas due to the density of pre-existing facilities (Reilly, 2000). This points towards an elevated solution to the congestion problem.

Yet one other issue that public transit encounters is convenience. Public transit is typically bound to assigned roadways, and this means that passengers must take specific paths and detours to get to destinations. As the average income within urban areas rises, the people living within them find ownership of a car to be increasingly attractive due to greater independence and the lack of rigid schedules that public transit must adhere to (Poudenx, 2008). This has led to less dependence on public transit and a continued rise in greenhouse emissions through personal vehicular transport. Other factors such as economic development can also be to blame, as they have resulted in heavily populated nations such as China pushing their populations towards vehicle ownership (Poudenx, 2008). In order to turn urban residents back towards public transit a reliable and flexible transit system that can adhere to the individual rider’s needs and schedule. This is where ATNs has arisen as a potential solution.

Automated Transit Networks, or ATNs, are personal vehicles designed to run on dedicated guideways. They are typically known as Personal Rapid Transit or PRT vehicles, or more colloquially as podcars (Furman et al, 2014). These vehicles are different from other public transit solutions in that they transport individual parties. This is similar to taxis, but the advantage of ATNs is that they do not share their guideways with any other modes of transportation. This system would provide the convenience that so many other transit systems lack and could therefore significantly reduce the number of petroleum-dependent vehicles on the road.

The Spartan Superway aims to take this concept a step further. The Superway is a plan to create an ATN that would run above currently existing roadways and is entirely dependent upon renewable energy resources. The ATN would consist of a fleet of podcars that automatically respond when called by a user at a station. This system would be unimpeded by the congestion of the roads below and thus would be more effective than current bus transit. The system’s planned reliance on renewable energy resources would drastically cut transportation-related greenhouse emissions by reducing the number of cars on the road.

**Member’s role**

Greg White is the project lead. Additionally, he is responsible for the mechanical switch mechanism and ensuring the bogie can run smoothly through its various functions. Keanu Heggem is responsible for the code pertaining to the function of the bogie, including optimization of the operations. Edward (Pinqian) Lin is responsible for reorganizing and compacting the control system so that it will run without overheating and can be hung from the bogie during operation. Brandon Scully is responsible for the CAD documentation of the project and for modeling planned changes to the bogie.

**Objectives**

The primary objective of the 2019-2020 Half-Scale Team is to develop a switching mechanism that gives the bogie the ability to smoothly navigate the Y-section of the track in either direction. Additionally, the team intends to redesign the control box so that it better ventilates and protects the electronics inside and can be hung from the bogie. The team also seeks to improve the means of inputting directions to the master Arduino by replacing the LCD panel inside the control box with a controller that connects to it via Bluetooth. Furthermore, the team intends to fix deficiencies within the drive system of the bogie.

The objectives would be considered met if the bogie is capable of navigating the Y-section of the track without fail if the control box can enclose the electronics and be hung from the bogie without overheating or shorting of wires and if the bogie can be controlled through a device that is not connected to the control box or the bogie in any physical manner.

**Structure of the Project**

The half scale-team is divided into two sub-teams, the mechanical team, and the control system team. The mechanical team will be focusing on solving the problem on the bogies and the control system team will be focusing on completing the control system, both hardware, and software. The first thing the team did at the beginning of the semester was going through the project report from previous years. In order to catch up with the project, each team member went through reports from different years and debriefed on the design concept and details to gain an understanding of the project. From the 2018-2019 half-scale team final report, the team learned that the bogie had not been tested on the switch portion of the track. With the new facility of Spartan-Superway, the team only obtain part of the track in the room.

Based on the report, the team tested the bogie and the control system on the track so that we can visualize the problem we are facing. Based on the testing result, the bogie has both mechanical and mechatronic issues that need improvement. Greg White and Brandon Scully will be focusing on solving the mechanical issues, such as the dimensions of the bogie and the track, the switching mechanism and the switching arm. Pinqian Lin and Keanu Heggem will be focusing on completing the control system such as designing and building the control system container, completing the hardware connection, changing power source from AC to DC, placing an in-built Bluetooth module for control, and completing the code for the control system. The goal for the mechanical team is to enforce gripping strength on the arms and adjust the parts on the bogie to fit into the track without slipping. The goal for the control system team is to design the container box with DC power supply to hang under the bogie with Bluetooth control and to finish the software programming. All team members elaborated on the presentations and documentation of the project together. All ideas were taken into consideration.

**State of the Art Literature Review**

In this day and age, self-driving vehicles have increased in popularity and have become a huge topic of conversation. As such, it is more likely that the transportation system would adopt this idea of having autonomous vehicles since labor costs would be lowered (Guerra, 2016, p. 210). Currently, there are transportation agencies that use driverless trains. For example, the KLIA Aerotrain in Malaysia acts as a “complimentary shuttle” between two terminals and travels on a fixed guide path (KLIA Aerotrain, n.d.).

For this project, the research is focused on elevated transportation systems. Specifically, the aim of the SPARTAN Superway project is to create an elevated automated transportation networking system where we power our vehicles with renewable energy. Companies such as JPods, FuTran and SkyTran are already in the process of making this dream a reality (Personal Rapid Transit (PRT), Personal Automated Transport (PAT), PodCar, IPM and ATN Quicklinks., n.d.). One company, ModuTram, has a very similar design to how we design and operate our bogie/passenger carriage system (Personal Rapid Transit (PRT), Personal Automated Transport (PAT), PodCar, IPM and ATN Quicklinks, n.d.). As shown in Figure **1** Their design includes the use of small podcars and an interface system that groups people going to a specific location and has an adjustable schedule to meet the demands of passengers (Personal Rapid Transit (PRT), Personal Automated Transport (PAT), PodCar, IPM and ATN Quicklinks, n.d.). One company, SkyTran, designed their transportation networks based on the fact that they want to be cost-effective and to increase the speed of the trains for better transportation experience (SkyTran, n.d.). However, the one distinguished design between SkyTran and the Spartan Superway is that they rely on a magnetic system for their suspension system compared to ours where we rely on a mechanically-operated system.

FuTran has emerged as the leading competitor in elevated ATN’s with a suspended guideway system (Futran, 2018). One of the few places which FuTran has tested their system is in South Africa. This is because overpopulation and the lack of efficiency in transporting goods and people outside the city have negatively impacted their society. FuTran’s elevated guideway is designed in such a way that it eliminates the congestion caused by road transportation. This correlates with the Spartan Superway’s philosophy of designing a suspended track/guideway that eliminates the use of road transportation to increase the safety and space for people to walk on the streets.

In terms of cost, it was found that light rails across the country and the Bay Area have a high cost of use per mile. For example, the Boston Green Line Extension costs $490 million for every mile of track (Bosselman, 2018). BART (The Bay Area Rapid Transit ) costs $780 million per mile to go to San Jose (Levy, 2018). However, one transportation system similar to the Superway’s, JPods, has an estimated cost of $10 million for every mile of track (Competitive Advantages, n.d.). From these findings, we can conclude that compared to regular modes of transit networking systems, the future of ATN will ensure an affordable, reliable, innovative and most importantly, safe way of transportation.

To summarize the significance of what previous Half-Scale teams did over the years, the 2016-2017 half-scale team managed to complete the fabrication and assembly of the bogie. Their research was based on building a guideway that was based on the functionalities of a roller coaster track and integrating the bogie on a 17-degree inclined guideway. To support an average 600-pound carriage, two bogie systems were used. The 2018-2019 half-scale team continued the progress done by previous years by improving the mechatronics side of the project: delivering powertrain and steering controls. On the mechanical design side, they improved the design of the upper and lower control arms along with the shafts to improve the tilting of the switching mechanism at a Y-section. As part of the 2019-2020 team, we are striving to improve on the designs of previous years by troubleshooting and enhancing the switch mechanism. We are also planning on adding a compact carriage box under the podcar to store the electrical components. Lastly, the mechanical design team is troubleshooting the mechanical aspect of the bogie and planning to improve the designs made on the upper control arm as well as the shaft connected to the driver.



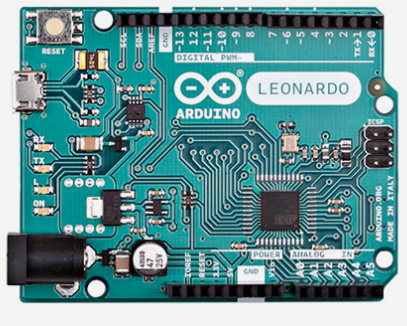
**Figure 1:**Podcar for ModuTram

**2019-2020 Research**

A component we are trying to integrate into our bogie/controls system box is the ArduinoBoard Leonardo. The reason for using this component is because one of its functions includes an in-built Bluetooth that is capable of remote control through a phone device. Our team is planning on implementing this board into the system to manually guide the bogie along the track using our mobile devices as shown in Figure **2**. The Arduino Leonardo is a microcontroller that consists of 20 digital input/output pins, a 16 MHz crystal oscillator, a micro USB connection, a power jack, an ICSP header and a reset button (Arduino Board Leonardo, n.d.). It also operates at 5V and has a recommended input voltage of 7-12 V. This board is shown in Figure **3.** What distinguishes the Leonardo from other boards is that it has an in-built USB communication which eliminates the need for a secondary processor. As a result of this, Leonardo can be seen as being connected to a computer as a mouse or a COM port (Arduino Board Leonardo, n.d.).



**Figure 2:** Example of a function for Arduino Leonardo



**Figure 3:** Arduino board for Arduino Leonardo

**Design Solution**

**Design Specifications**

The bogie is required to run along a preconstructed iron track. The bogie is to be able to reach any point along the track from any other point on the track. The bogie must operate in a pair with another bogie which faces the opposite way, meaning the bogie must be able to move effectively in both directions. The track is to be passive, carrying out no mechanical changes and simply supporting the weight of the bogie. This stipulates that the bogie must execute track switches independently. The half-scale bogie, therefore, must have a switching mechanism that can reliably force the bogie onto one side of a Y-section with no external aid. This mechanism must be controlled through user input. The bogie must also be able to move using a power source that is either bound to the bogie and able to be recharged without either it or the bogie being removed from the track or is bound to the track and is able to power the bogie at all times.

Previous teams also had a requirement that the bogie system is able to traverse up a 17-degree incline while carrying 600 lb of external weight. This is a relatively minor concern for this year’s team, as the track segment that is currently available to the team does not support such an incline and the work of this team is focused on a single bogie instead of on a bogie pair. This current team is focused primarily on the switch mechanism and on optimizing the electronics that govern the bogie’s operation, with other functions of the bogie being secondary concerns.

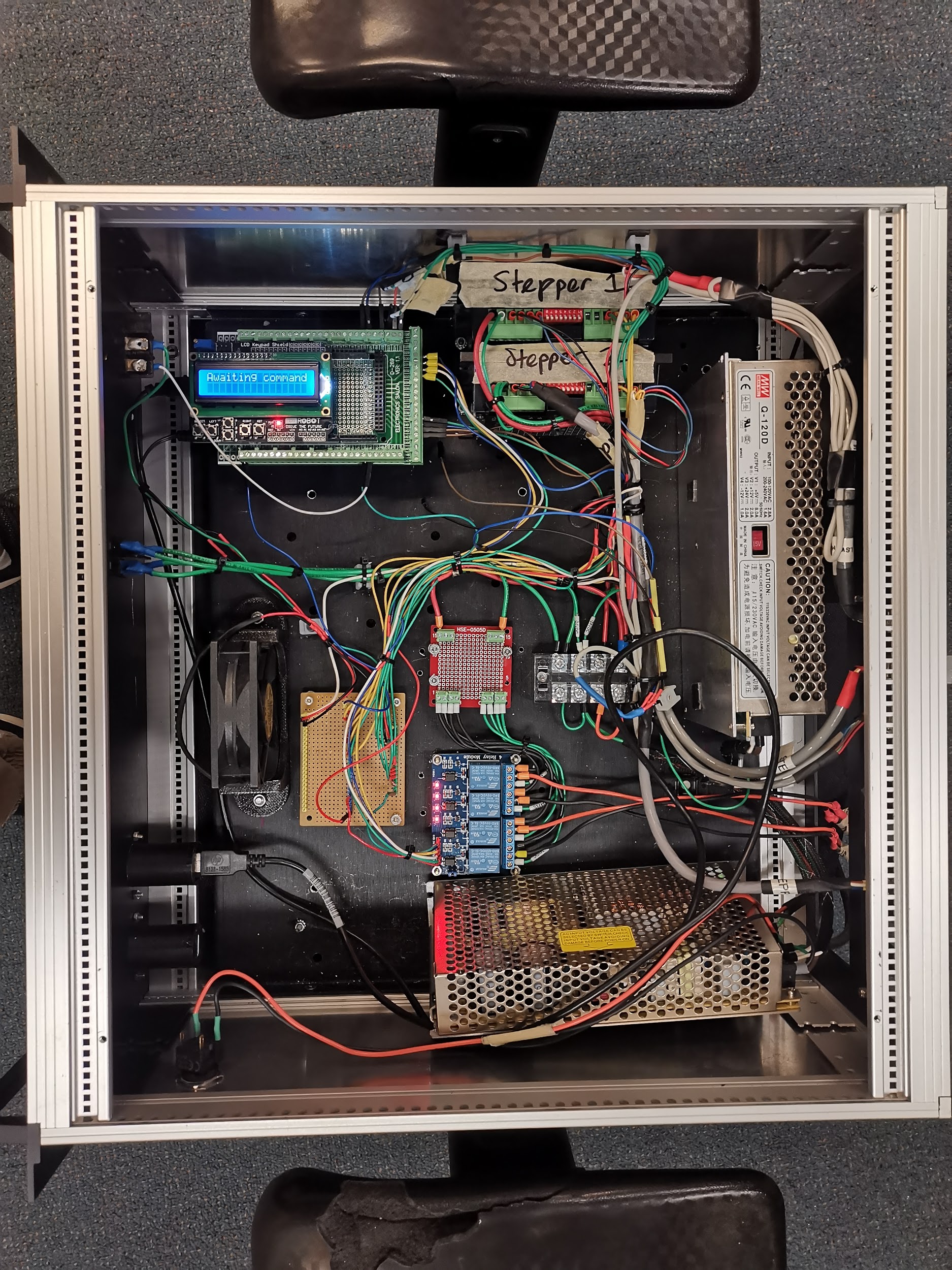
**Prime Design**

* Power Management

The control system designed from last year is used AC voltage for the power supply. Using AC voltage limits the mobility of the control system because of the cables for the plug. In order for the bogie to navigate the track freely, implementing an onboard rechargeable battery system is necessary. To convert the control system to a DC power supply, the team came up with two solutions. Factoring in the overall cost of the project, we used as many sourced parts as we could find. Using a 36 V battery a team member had on hand and purchasing a current/voltage converter, the 36 V DC voltage can be divided into 12 V and 24 V respectively. An alternative solution is to have two batteries, one 12 V and the other 24 V to power the system.

* Carriage

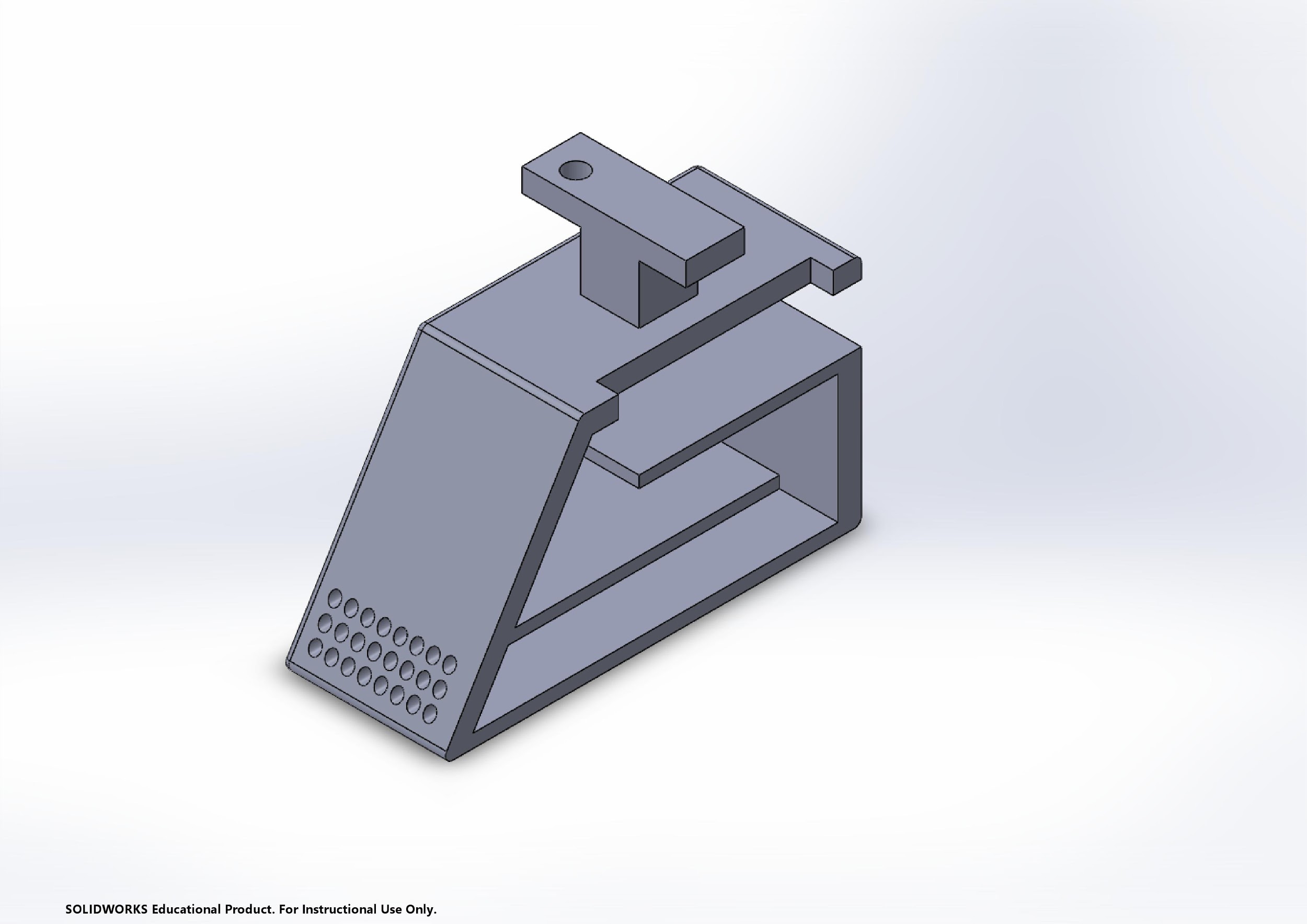
The current control system is constructed into a computer host shell. Having the control system attached to the bogie as it moves on the track will better demonstrate the full functionality of the half-scale project. Changing the control system carriage will involve fabricating a carriage specifically for all the components. In order to manufacture a carriage for the control system, each component must be analyzed for the correct placement. To design the carriage for the control system, the dimensions of each component is needed. Since all the components are soldered, we must take the system apart for proper measurement referring to the current circuit diagram. The control system box designed by the previous year is shown in Figure **4**.



**Figure 4**: Previous years control box

* Heat Dissipation

A new carriage for the control system is needed, heat transfer and dissipation must be taken into account. A cooling fan was added to the control system to prevent overheating but it was not very effective in reducing the temperature. The fan used to cool down the system is not as efficient in the current location as it is not directly cooling the components of the highest temperature output. By the end of the Fall 2019 semester, the half-scale team drew the circuit diagram and finalized the design of the new control system carriage. The new control system carriage will have a ventilation pipe design. The advantage of using the ventilation pipe concept is air circulation will be able to go through each of the components effectively. Figure 5. shows the cross-sectional view of the new carriage. The component will be placed on three different levels. The fan will be placed at the back of the carriage of the top-level, as well as the most heated components. Holes will be cut out at the bottom level for air outflow. The plan for the winter break will be to focus on taking measurements of each component and put in the CAD file for designing the carriage. Since the carriage will be designed for half-scale, the size of the carriage might be too big for 3-D printing, a modular structure will be used.



**Figure 5:**New control box carriage

* Wireless Communications Module

The code has been modified to test various solutions. The group found that the bogie could exercise the switch mechanism in motion if programmed to do so. However, after extensive testing, it was determined that the single 6 button panel on the bogie was insufficient to perform all the functions that the team would like the prototype bogie to perform. The panel is a 6 button LCD and all 6 buttons are on a single circuit and occupying a single input pin. Therefore, it is impossible to combine inputs from the panel, the lowest voltage would be the only value read by the Arduino. This means that the bogie can only perform 6 independent actions at most with the LCD panel. The practical application of the bogie would be most efficient if it could switch while in motion, the bogie must be able to do at least 8 functions at a time.

An additional concern is that the control box is intended to be mounted under the bogie and therefore must move with it. This would require an operator to walk alongside the bogie while operating it, which is inconvenient due to the space that the track occupies. This is also a safety hazard during testing as it would place the operator’s head directly beneath the bogie while it is attempting to switch. The panel will be replaced by a controller with a wirelessly connected joystick, allowing for upwards of eight input ranges that would give the bogie the ability to operate the switch mechanism while in motion in either direction and still maintain its ability to exercise the mechanism in a stationary setting and move forward or backward without using the mechanism. These eight directional inputs would be in clockwise order: forward, forward-right switch, right switch, backward-right switch, backward, backward-left switch, left, and forward-left switch. The diagrams of the controller and the master Arduino setup are attached in the appendix as Figures **11** and **12**. The flowchart for the codes of the controller and receiver are also attached in the appendix as Figures **9** and **10**.

**Plans for Fabrication**

Research and review of inventory revealed that it would be relatively inexpensive to design and build a controller for this purpose instead of purchasing one from the internet, due to the team finding a spare Arduino leftover from the previous team. When this is completed and implemented into the redesigned carriage the bogie will be able to be controlled without any physical contact with the bogie or the carriage. The team intends to create the controller by connecting the spare Arduino with an NRF24L01+ 2.4GHz Wireless Transceiver Module. This module would transmit to an identical module on the “server”, which is the master Arduino that sits within the control box and controls the bogie’s functions. A joystick would then be connected to the spare Arduino and would be used to input the commands to the bogie. This joystick would be programmed into eight input ranges based on the combinations of x and y values. The entire setup would then be housed within a 3D printed shell in the same manner as the redesigned control box. The controller will use a 36 V battery power source which will be implemented to the controller by the beginning of the Spring semester.

The completion of the control system will require a voltage converter and new wires. These will be purchased during Winter break. Additionally, the control system components and carriage design will be finished. As soon as the CAD file of the control system is completed, it can be 3-D printed. Finishing the design of the control system carriage will allow the team to 3-D print it when the Spring 2020 semester starts. As soon as the carriage is 3-D printed and the control system is finished, the team will test the functionality of the system and test it with the bogie on the track.

**Conclusions and Future Work**

At the last meeting of the Fall semester, the control box suffered a short which damaged one of the steppers and possibly the Arduino as well. As of yet, the team has not been able to confirm the extent of the damage or what will have to be replaced but will do so thus upcoming winter break. This setback emphasized the already stated need for a redesign of the control box. The previous team did an excellent job despite suffering several setbacks, but the current control box is not suitable to run for extended periods of time or for hanging beneath the bogie as originally intended. The primary focus of the team over the break before the Spring semester is to redesign the control box and diagnose stepper motor errors so that no more issues arise from the electrical components of operations.

Once the winter break tasks have been completed, the team will attempt to redesign the switching mechanism. The twisting of the bogie requires a lot of work to fix, and the team has considered several times throughout the semester that replacing the mechanism entirely may be a more effective solution than fixing what is currently on the bogie. The recent burning out of the steppers in the control box due to a short is just the latest of the reasons to replace the currently existing design. The team found a pair of linear actuators leftover from the previous year and is considering replacing the stepper motor and chain with them. Further design is needed to solidify this alternate switching mechanism. A prototype code for a switching mechanism utilizing the linear actuators can be found in the appendix after the currently applied code for the switching mechanism.

The team also has obtained rechargeable batteries that could be used to power the bogie independently. The team plans to integrate those into the bogie in the following semester in order to run the bogie without it being constantly plugged into the wall. This would open up the possibility of a recharge station on the track that would cover the needs for which the previous years tried to utilize wayside power. The team will also attempt to utilize the variety of sensors on the bogie to begin work on autonomous mode. The ultrasonic sensors and limit switches are good starting points on this, as they allow the bogie to properly operate the switching mechanism and exercise collision avoidance. The hall effect sensor, which would allow the bogie to detect the presence of the track in front of it and therefore find the switch point on the track, will have to be left for a future year to implement into autonomous mode. The reason for this is that the hall effect sensor depends on detecting a magnetic field generated by an electrical current flowing through a magnetic metal. As we are not working alongside wayside power this year, the hall effect sensors cannot be tested. In pursuance with the redesign of the controller input layout, the team will build a wireless controller that will connect with the Arduino in the control box. A spare Arduino board left over from the previous team’s work will make this very cheap. A wireless transceiver module has been located and should suit this purpose.

**References**

Arduino Board Leonardo. (n.d.). Retrieved December 14, 2019, from https://www.arduino.cc/en/Main/Arduino\_BoardLeonardo.

Bosselman, A. (2018, June 18). Bay Area subway and rail costs: Why are they among the highest in the world? Retrieved December 14, 2019, from https://sf.curbed.com/2018/6/18/17464616/bay-area-subway-train-rail-costs-price-bart-muni.

Competitive Advantages. (n.d.). Retrieved December 14, 2019, from <https://www.jpods.com/aboutjpods/advantages.html>

Futran (2018) Retrieved from <https://futrangroup.com/>

Guerra, E. (2016). Planning for cars that drive themselves: Metropolitan planning organizations, regional transportation plans, and autonomous vehicles. *Journal of Planning Education and Research*, *36*(2), 210-224.

Kenworthy, J.R. (2003). Transport energy use and greenhouse gases in urban passenger transport systems: A study of 84 global cities. In: International Sustainability Conference, 17 - 19 September, Fremantle, Western Australia

KLIA Aerotrain. (n.d.). Retrieved from http://klia.info/arrivals/aerotrain.htm.

Levy, A. (2018, December 19). Why Does Urban Rail Cost So Much in the U.S.? Retrieved from https://www.citylab.com/transportation/2018/01/why-its-so-expensive-to-build-urban-rail-in-the-us/551408/.

Lindley, J., (1987). Urban Freeway Congestion: Quantification of the Problem and Effectiveness of Potential Solutions. *ITE Journal*, *1*(1), 27-32

Personal Rapid Transit (PRT), Personal Automated Transport (PAT), PodCar, IPM and ATN

Quicklinks. (n.d.). Retrieved December 14, 2019, from <http://faculty.washington.edu/jbs/itrans/prtquick.htm>.

SkyTran. (n.d.). Retrieved December 14, 2019 from <https://www.skytran.com/>

Poudenx, P., (2008). The effect of transportation policies on energy consumption and greenhouse gas emission from urban passenger transportation, *Transportation Research Part A: Policy and Practice, 42*(6), 901-909

Reilly, J.J. (2000). The management process for complex underground and tunneling projects. *Tunneling and Underground Space Technology, 15*(1), 31-44

**Appendix**

**Appendix A**

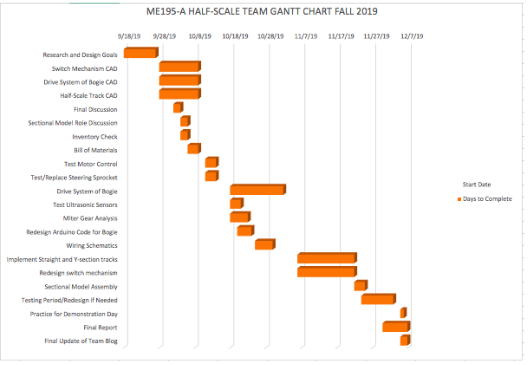
**Table 1**: Bill of materials

|  |  |  |  |
| --- | --- | --- | --- |
| Product Name | Description | Quantity | Cost Per Unit |
| NRF24L01+ 2.4GHz Wireless Transceiver Module | Wireless Communications adapter for Arduino | 4 | $0.945 ($3.78 as a pack) |
| Kohree DC/DC Converter Regulator Reducer 36V Step Down to 12V 10A Car Power Supply Module Voltage Converter Regulator Electronic Power Supply Transformer Volt Module | Voltage converter from 36V to 12V with 3A needed | 1 | $13.99 |
| Cllena DC 36V 48V Step Down to 24V 10A 240W Converter Voltage Reducer, Waterproof DC/DC Buck Transformer Power Supply | Voltage converter from 36V to 24V with 7.3A needed | 1 | $30.99 |

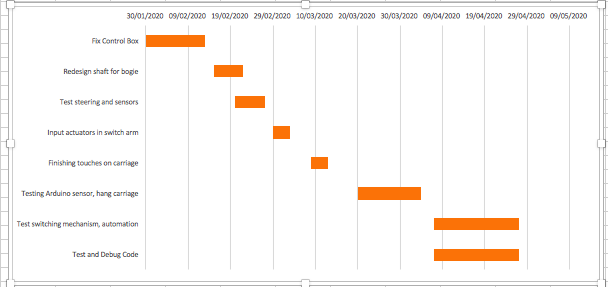
**Gantt Chart**



**Figure 6**: Gantt Chart

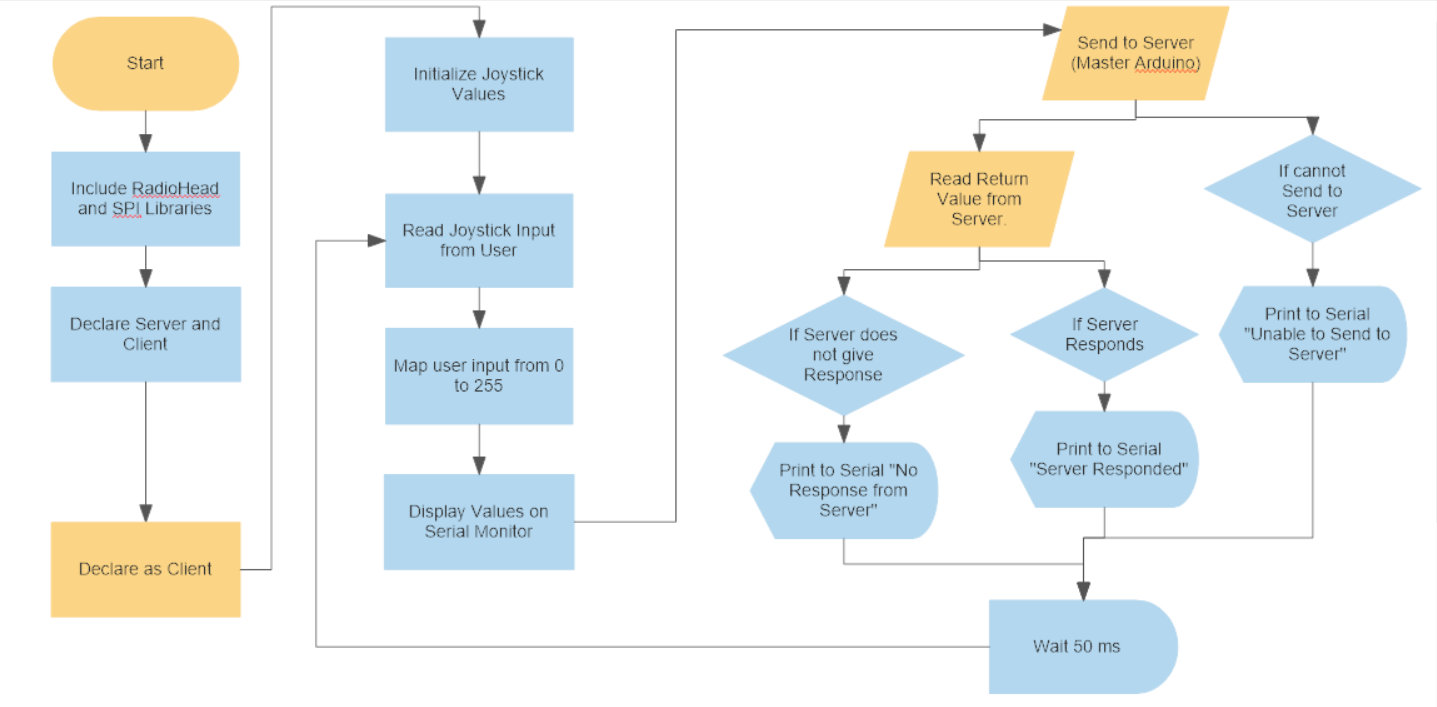


**Figure 7:** Fall 2019 Gantt Chart

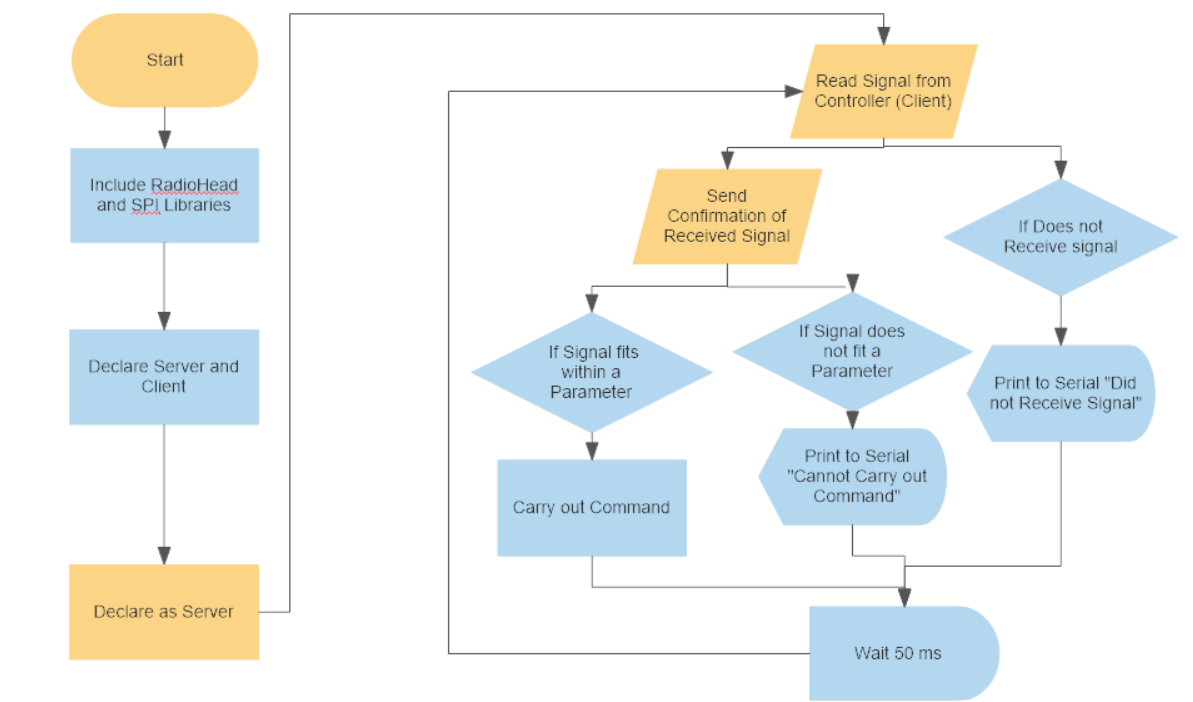


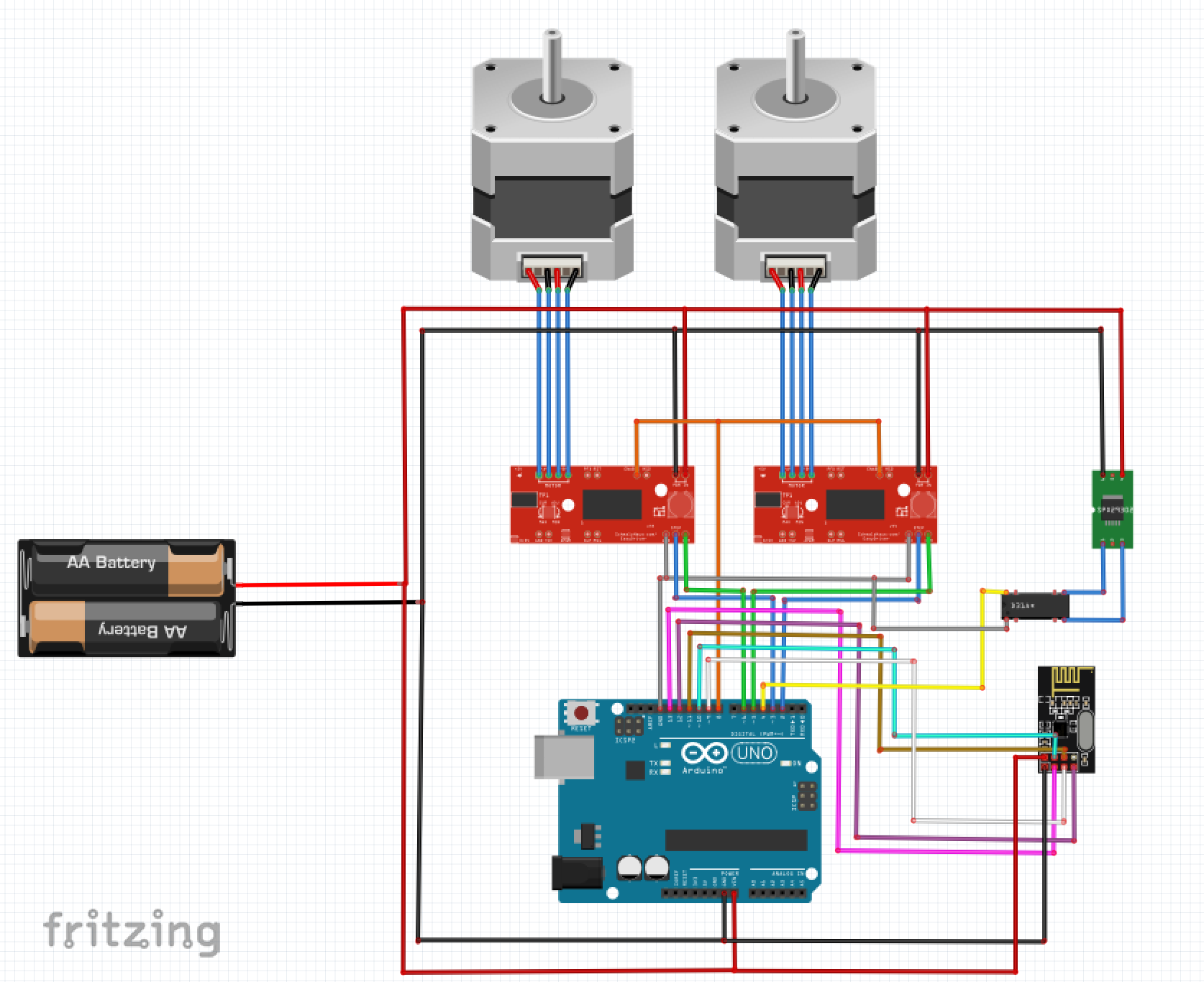
**Figure 8**: Spring 2020 Gantt Chart

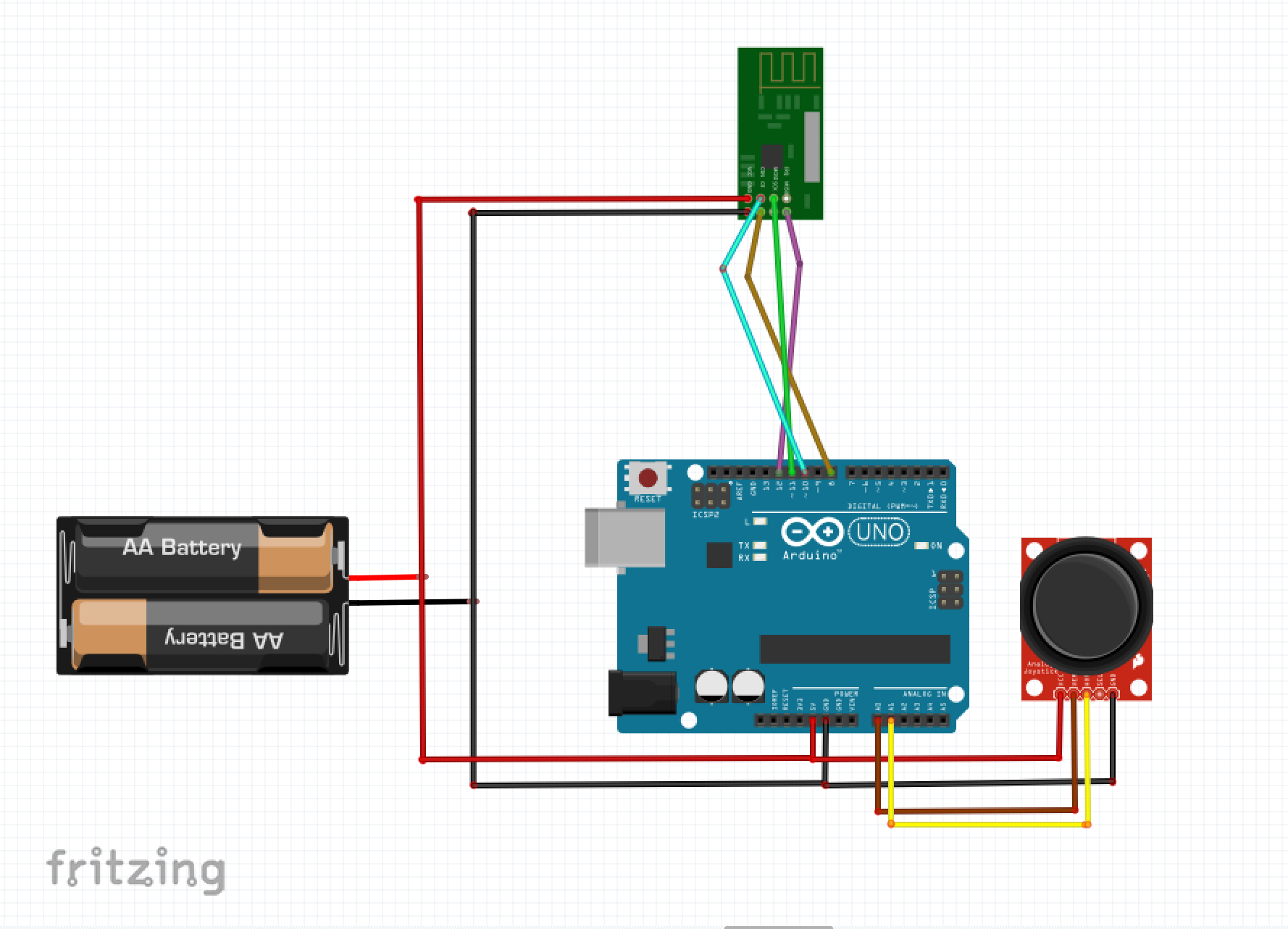
**Appendix B**

-

**Figure 9:**Flowchart of Controller Code

**Figure 10**: Flowchart of Receiver Code

**Figure 11**: Master Arduino Schematic With Wireless Communicator

**Figure 12:** Wireless Controller Schematic

**Appendix C**

**Currently applied code using Stepper Motors to operate Switching Arm**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Half Scale Bogie

2019

Written by: Keanu Heggem

UPDATE LOG:

V1.3

INTEGRATED 4-CHANNEL RELAY BOARD INTO POWERTRAIN

ADDED SUPPORT FOR BOGIE 2 STEPPER MOTOR LIMIT SWITCHES

V1.4

CHANGED LIMIT SWITCH BOGIE 1 PINS (REVERSED)

ADDED STOPBOGIE FUNCTION TO NO BUTTON CASE

CHANGED NOTES ON POWERTRAIN PIN DEFINITION TO DEFINE FORWARD/REVERSE COMMANDS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#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

//Limit Switches for oversteer protection

//Bogie 1

const byte limitL1 = 20; //attachInterrupt pins can only be 2, 3, 18, 19, 20, 21

const byte limitR1 = 21;

//Bogie 2

const byte limitL2 = 19;

const byte limitR2 = 18;

#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 in1 30 //Bogie motor 1 YELLOW

#define in2 31 //ORANGE

#define in3 32 //GREEN

#define in4 33 //Bogie motor 2 BLUE

#define dirB1 34 //PURPLE

#define dirB2 35 //GREY

#define hallF 44

#define hallR 45

int test = 3000; //The two values for the stepping size

int test2 = -3500;

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

Stepper stpr2(onerev, pulse2, stprDIR2);

//Other

int combinevar = digitalRead(A0);

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 for bogie 1

attachInterrupt(digitalPinToInterrupt(limitL1), isr, FALLING);

attachInterrupt(digitalPinToInterrupt(limitR1), isr, FALLING);

pinMode(limitL1, INPUT\_PULLUP);

pinMode(limitR1, INPUT\_PULLUP);

//Limit Switches for bogie 2

attachInterrupt(digitalPinToInterrupt(limitL2), isr, FALLING);

attachInterrupt(digitalPinToInterrupt(limitR2), isr, FALLING);

pinMode(limitL2, INPUT\_PULLUP);

pinMode(limitR2, 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(in1, OUTPUT); //motor 1 - HIGH = FORWARD | LOW = REVERSE

pinMode(in2, OUTPUT); //motor 1 - LOW = FORWARD | HIGH = REVERSE

pinMode(in3, OUTPUT); //motor 2 - HIGH = FORWARD | LOW = REVERSE

pinMode(in4, OUTPUT); //motor 2 - LOW = FORWARD | HIGH = REVERSE

//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);

lcd.clear();

lcd.print("Turning Right");

rightlimit();

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.println(adc\_key\_in);

Serial.println(combinevar);

lcd.clear();

lcd.print("Going Forward");

forward();

break;

}

case btnDOWN:

{

Serial.println(adc\_key\_in);

Serial.println(combinevar);

lcd.clear();

lcd.print("Going Backwards/n Right Turn");

backwards();

break;

}

case btnNONE:

{

stopBogie();

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(limitL1);

while (l == LOW) {

delay(200);

digitalWrite(in1, LOW );

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

stpr1.setSpeed(STOP);

stpr1.step(STOP);

Serial.println("Steering left limit triggered, stop turning");

break;

}

while (l == HIGH) {

delay(200);

digitalWrite(in1, LOW );

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, 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(limitR1);

while (j == LOW) {

delay(200);

digitalWrite(in1, LOW );

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

stpr1.setSpeed(STOP);

stpr1.step(STOP);

Serial.print("Steering right limit triggered, stop turning");

break;

}

while (j == HIGH) {

delay(200);

digitalWrite(in1, LOW );

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, 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() {

delay(200);

digitalWrite(in1, HIGH );

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

Serial.println("\t \t \t Moving Forward");

}

void backwards() {

delay(200);

digitalWrite(in1, LOW );

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

Serial.println("\t \t \t Moving Backwards");

}

void stopBogie() {

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

digitalWrite(in3, LOW);

digitalWrite(in4, LOW);

delay(200);

Serial.print("\t \t \t Bogie Stopped");

}

void autonomousMode() {

monitorFront();

monitorRear();

}

void isr() {

stpr1.step(STOP);

Serial.print("ISR");

}

**Prototype code using Linear Actuators instead of Stepper Motors.**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Half Scale Bogie

2019

Written by: Keanu Heggem

UPDATE LOG:

V1.3

INTEGRATED 4-CHANNEL RELAY BOARD INTO POWERTRAIN

ADDED SUPPORT FOR BOGIE 2 STEPPER MOTOR LIMIT SWITCHES

V1.4

CHANGED LIMIT SWITCH BOGIE 1 PINS (REVERSED)

ADDED STOPBOGIE FUNCTION TO NO BUTTON CASE

CHANGED NOTES ON POWERTRAIN PIN DEFINITION TO DEFINE FORWARD/REVERSE COMMANDS

V1.5  
 REPLACED STEPPER MOTOR PINS WITH OUTPUTS FOR LINEAR ACTUATORS.

PROGRAMMED ACTUATORS TO GIVE OUT INVERSE INPUTS FOR THE DIRECTIONS THEY FACE.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#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

//Limit Switches for oversteer protection

//Bogie 1

const byte limitL1 = 20; //attachInterrupt pins can only be 2, 3, 18, 19, 20, 21

const byte limitR1 = 21;

//Bogie 2

const byte limitL2 = 19;

const byte limitR2 = 18;

#define left1 22 //Color brown on breadboard - stepper 1 on bogie 1

#define right1 23 //Color orange on breadboard - stepper 1 on bogie 1

#define left2 24 //Stepper 2 on bogie 2

#define right2 25 //Stepper 2 on bogie 2

#define autoSwitch 53 //Autonomous mode switch

#define in1 30 //Bogie motor 1 YELLOW

#define in2 31 //ORANGE

#define in3 32 //GREEN

#define in4 33 //Bogie motor 2 BLUE

#define dirB1 34 //PURPLE

#define dirB2 35 //GREY

#define hallF 44

#define hallR 45

int test = 3000; //The two values for the stepping size

int test2 = -3500;

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;

//Other

int combinevar = digitalRead(A0);

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 for bogie 1

attachInterrupt(digitalPinToInterrupt(limitL1), isr, FALLING);

attachInterrupt(digitalPinToInterrupt(limitR1), isr, FALLING);

pinMode(limitL1, INPUT\_PULLUP);

pinMode(limitR1, INPUT\_PULLUP);

//Limit Switches for bogie 2

attachInterrupt(digitalPinToInterrupt(limitL2), isr, FALLING);

attachInterrupt(digitalPinToInterrupt(limitR2), isr, FALLING);

pinMode(limitL2, INPUT\_PULLUP);

pinMode(limitR2, INPUT\_PULLUP);

//Steering

pinMode(left1, OUTPUT); //Define number of staps for a full rev

pinMode(right1, OUTPUT);

pinMode(left2, OUTPUT);

pinMode(right2, OUTPUT);

//Steering sensors

pinMode(hallF, INPUT);

pinMode(hallR, INPUT);

//Serial

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

//Powertrain

pinMode(in1, OUTPUT); //motor 1 - HIGH = FORWARD | LOW = REVERSE

pinMode(in2, OUTPUT); //motor 1 - LOW = FORWARD | HIGH = REVERSE

pinMode(in3, OUTPUT); //motor 2 - HIGH = FORWARD | LOW = REVERSE

pinMode(in4, OUTPUT); //motor 2 - LOW = FORWARD | HIGH = REVERSE

//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);

lcd.clear();

lcd.print("Turning Right");

rightlimit();

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.println(adc\_key\_in);

Serial.println(combinevar);

lcd.clear();

lcd.print("Going Forward");

forward();

break;

}

case btnDOWN:

{

Serial.println(adc\_key\_in);

Serial.println(combinevar);

lcd.clear();

lcd.print("Going Backwards/n Right Turn");

backwards();

break;

}

case btnNONE:

{

stopBogie();

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 rightlimit() {

int l = 0;

l = digitalRead(limitL1);

while (l == LOW) {

delay(200);

digitalWrite(in1, LOW );

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

stpr1.setSpeed(STOP);

stpr1.step(STOP);

Serial.println("Steering left limit triggered, stop turning");

break;

}

while (l == HIGH) {

delay(200);

digitalWrite(in1, LOW );

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

digitalWrite(left1, LOW );

digitalWrite(left2, HIGH);

digitalWrite(right2, LOW);

digitalWrite(right1, HIGH);

stpr1.setSpeed(fast);

stpr1.step(test);

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

break;

}

}

void leftlimit() {

int j = 0;

j = digitalRead(limitR1);

while (j == LOW) {

delay(200);

digitalWrite(in1, LOW );

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

stpr1.setSpeed(STOP);

stpr1.step(STOP);

Serial.print("Steering right limit triggered, stop turning");

break;

}

while (j == HIGH) {

delay(200);

digitalWrite(in1, LOW );

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

Serial.print("Turning Right for: \t \t");

digitalWrite(left1, HIGH );

digitalWrite(left2, LOW);

digitalWrite(right2, HIGH);

digitalWrite(right1, LOW);

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 backwards() {

delay(200);

digitalWrite(in1, HIGH );

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

Serial.println("\t \t \t Moving Forward");

}

void forward() {

delay(200);

digitalWrite(in1, LOW );

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

Serial.println("\t \t \t Moving Backwards");

}

void stopBogie() {

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

digitalWrite(in3, LOW);

digitalWrite(in4, LOW);

digitalWrite(left1, LOW);

digitalWrite(left2, LOW);

digitalWrite(right2, LOW);

digitalWrite(right1, LOW);

delay(200);

Serial.print("\t \t \t Bogie Stopped");

}

void autonomousMode() {

monitorFront();

monitorRear();

}

void isr() {

stpr1.step(STOP);

Serial.print("ISR");

}