Autonomous Subway Inspection Vehicle

Omar Aziz, Noah Bolzoni, Brandon Castro, Anthony Lucero, Sean Maggio, Marcin Wisniowski

Phase 5 Documentation

Prototype Building & Performance Testing Plan

04/13/2020

Project Description

The New York City subway is the busiest metro service in the United States. Annual subway ridership in New York City surpassed 1.68 billion in 2018 and daily passengers exceed 5.6 million. Therefore, each issue found on the subway track inevitably leads to the inconvenience and delay of millions of Americans living in New York City. Even with the current advancements in technology and automation, many railway track inspections are conducted regularly by human inspectors. In order to satisfy current regulations prescribed by the Federal Railroad Administration, a mainline track is usually required to undergo a weekly inspection and a geometry car inspection is not considered acceptable for meeting this required inspection frequency. It becomes apparent, therefore, that inspectors are in high demand and yet low in availability.

The Autonomous Subway Inspection Vehicle (subsequently referred to as the ASIV) looks to increase the productivity of subway inspections by uncovering and flagging warnings for track defects early, in order to concentrate a subway inspector's efforts into distinct geographic areas of focus. While the current regulations may restrict the creation of a fully autonomous system, the project, regardless, adds value by creating an autonomous and live asset management of the subway track. When regulations change in the future, the system can be used to provide stronger analysis for proactive operational investments and scheduling of site-technicians and inspectors for daily maintenance.

The current prototype design will incorporate several features to demonstrate ASIV's primary purpose. It will utilize two high accuracy laser distance sensors attached to the chassis which will measure the distance to each of the two rails and from there, the values can be converted to gage (the distance between the two rails) given the fixed distance between each sensor. Gage was selected as the primary characteristic to measure since variations in this value can at best cause undesirable ride characteristics, and at worst cause a derailment. Additionally, the vehicle will be self propelled using one electric motor for each of its four wheels to demonstrate its ability to autonomously travel across and measure for defects at any part of the subway system with onboard power for propulsion and electronics provided by a battery. In accordance with this goal of autonomous and self-propelled travel, control systems will be incorporated to allow the ASIV to travel at a constant monitoring speed via Arduino microcontrollers. A unique wheel-retraction and collision avoidance mechanism would also be implemented to allow the ASIV to fold itself into the rail bed, out of the path of any oncoming revenue trains. These features are further discussed in the following sections.

Mechanical Design

Revisions to Motor Selection and Analysis

The targeted performance values for ASIV's propulsion system were changed since the last phase to better represent a prototype that could meet the customer's needs. There were concerns that the electric motor selection and quantity would not be suitable for the inclines present on subway systems such as NYC's since the selected motors might perform poorly in terms of acceleration or in the worst case, stall going uphill. As such, the number of driven wheels was increased from 2 to 4, thus making all of the wheels contributors to the vehicle's traction (this will also reduce the chances of wheel slip when accelerating). Additionally, the specified motors were swapped out for motors with higher power ratings, and more importantly, higher rated currents. In most cases, it was found that the current draw would be the most significant limiter of vehicle propulsion since this value could not be exceeded without the possibility of causing significant damage to the motors.

Table 1: Changes to Motor Specifications

Motor Specification	**Phase IV - Original**[1]	**Phase V - Revised**[2]
Rated Power	150 W	250 W
Rated Voltage	24 V	24 V
No Load Speed	3500 rpm	3350 rpm
Rated Torque	0.52 N-m	0.9 N-m
No Load Current	0.8 A	1.6 A
Max Rated Current	9.6 A	13.4 A
Quantity	2	4

The following analysis parameters relevant to the vehicle design were also updated to incorporate the addition of the retractable legs which increased ASIV's overall weight and frontal surface area.

[1] Based on specifications for Type ZY6812 DC Motor.
[2] Adapted from specifications for Type MY1016 DC Motor.

Table 2: Changes to Relevant Vehicle Specifications

Vehicle Specification	**Phase IV - Original**	**Phase V - Revised**
Total Mass	23 kg	30.6 kg [3]
Wheel Diameter	32 cm	32 cm
Projected Frontal Area	0.3 m^2	0.375 m^2
Drag Coefficient	1.0	1.0
Rolling Resistance Coefficient	0.002	0.002

Additionally, in accordance with the improvements made to propulsion, more appropriate target values for operating speed and acceleration were adapted as well.

Table 3: Changes to Propulsion Performance Targets

Performance Target	**Phase IV - Original**	**Phase V - Revised**
Normal Operating Velocity	10 mph	10 mph
Normal Acceleration	0.2 m/s^2	0.4 m/s^2
Reduced Velocity	5 mph	10 mph (reduction not necessary)
Reduced Acceleration	0.0667 m/s^2	0.1 m/s^2
Gradient	20 mm/m	40 mm/m

The justification for removing the reduced velocity target for the new motor selection is shown in the following analysis. The next two figures demonstrate the current draw for the new motor characteristics while accelerating on level track and a 40 mm/m positive incline respectively. Please refer to the appendix for an explanation of how the values for current, torque, tractive force, etc. were calculated.

[3] This new value also accommodates the change from 2 7 Ah batteries to a single 18 Ah battery discussed in the next paragraphs.

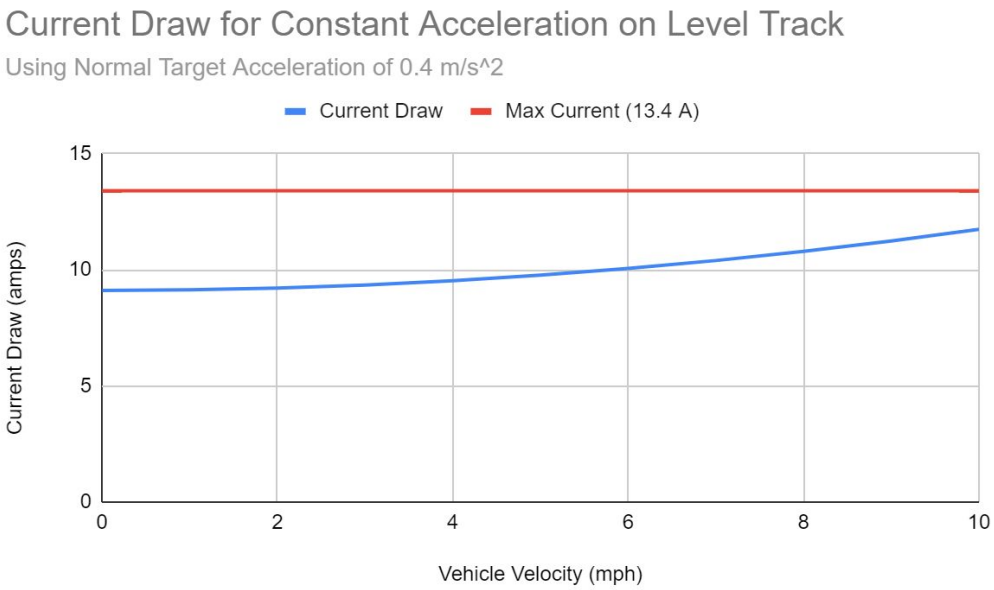


Figure 1. Current Draw vs. Velocity

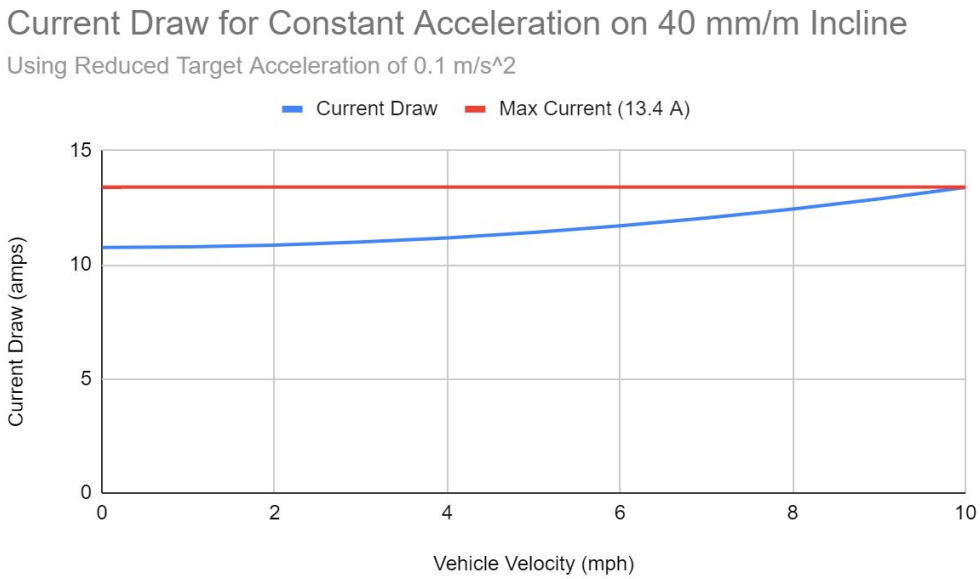


Figure 2. Current Draw vs. Velocity Moving Uphill

Notice that the blue line represents the required current draw to maintain the indicated target acceleration from 0 to 10 mph considering all of the resistance forces covered in the appendix, while the red line shows the limiting value of the specified motor at 13.4 amps. This value is not exceeded in either case, and thus, the motors will suffice in this range of speed. This contrasts from the original setup whereby the normal acceleration target was half of the new value and 67% of the reduced acceleration value for inclines in order to stay below the previous rated

current threshold of 9.6 amps. Thus, the new motor design can accelerate twice as fast on level track and climb grades twice as steep as the old case.

Another aspect of the propulsion that was improved was that of ASIV's range. The pair 7 Ah batteries (also used in the alpha prototype) would be replaced by a single 18 Ah, thereby doubling the capacity to store energy for longer trips across different routes[4]. The combined effects of the electric motor and battery refinements would yield the following improvements to ASIV's expected range before depletion and need for recharge.

Table 4: Improvements to Range

Performance at Target Speed (10 mph)	**Phase IV - Original**	**Phase V - Revised**
Required Tractive Effort	4.05 N	5.10 N
Required Torque	0.3239 Nm	0.2039 Nm
Current Draw	5.746 A	4.580 A
Battery Life	1.218 hours	3.930 hours
Equivalent Distance Traveled	12.18 miles	39.30 miles

Notice that the while the tractive effort was increased due to the increased weight and resistance, the torque at each wheel (and resulting current draw) was decreased due to the distribution of power across all four wheels. This resulted in a three-fold increase to equivalent distance traveled on level track at the target speed of 10 mph. While the change in components would likely necessitate an increase to the overall cost of the prototype, the benefits in increased performance as well as the current favorable budget indicate that such expenses are worth the investment. The remaining funds in the budget for said parts are discussed in the project plan section.

[4] Proposed battery characteristics from Okoman Model 6S5P.

Vehicle Design

The use of SolidWorks and Motion Analysis (animation) has been employed thoroughly for the project since Phase I. Now that the project must shift to simulation and analysis, the team was ready and experienced in this regard. Motion analysis was used to simulate all parts of the retraction mechanism with realistic specifications and conditions. The retraction mechanism uses many linear actuators to accomplish four main movements. The first movement is to rotate the four legs from their retracted position around the hinge to their extended, vertical position where they will support the weight of the vehicle. The second movement is to extend the telescoping portion of the legs until each leg contacts the ground, and then slightly lift the vehicle off of the tracks. The third movement is to retract the wheels inward using their respective actuators. Finally, the fourth movement involves retracting the telescoping portion of the leg to lower the vehicle down closer to the ground. The animation can be viewed via the following YouTube link: https://youtu.be/NIcN_vaSdo8 . A before and after snapshot of the sequence is shown below.

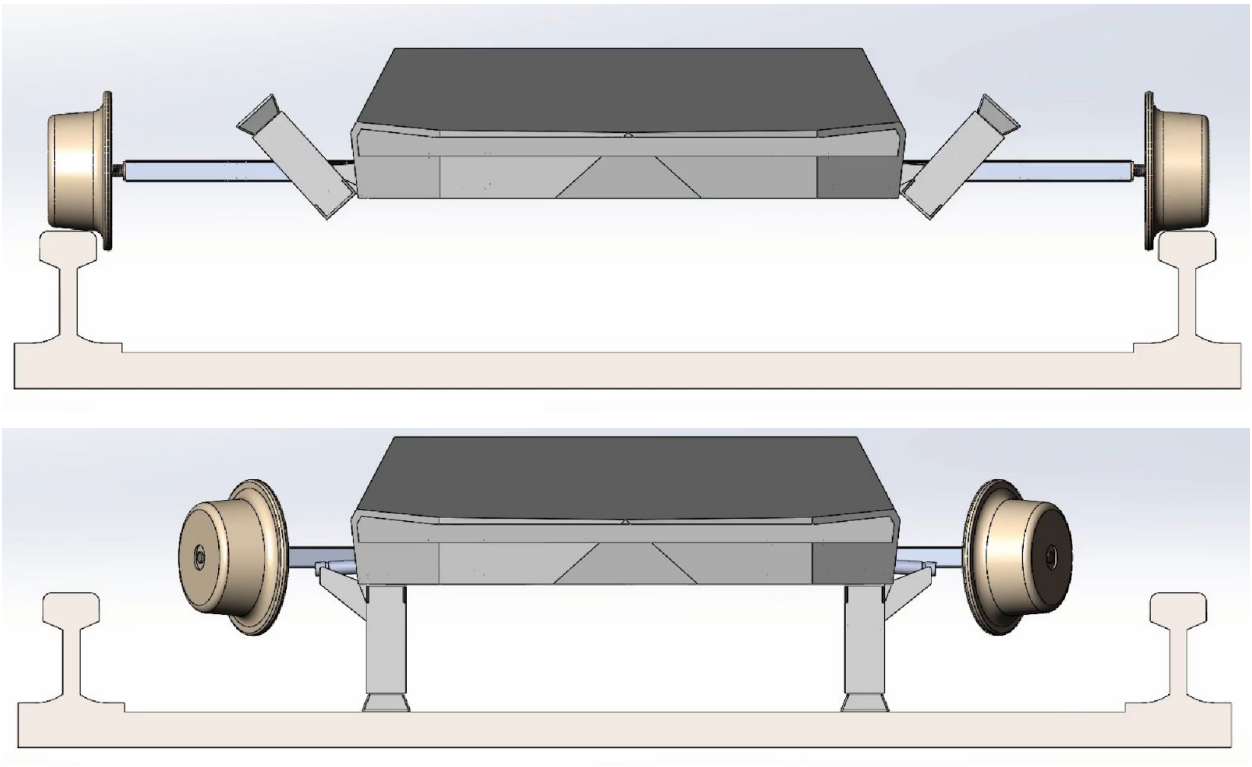


Figure **3** - Before (top) retraction and after (bottom) retraction

Previous iterations of the leg design featured a fixed actuator that was inside the vehicle and angled relative to the ground to provide more stability. However, changing the length of angled legs will cause the point of contact between the legs and the ground to shift, as changing the length now changes an ‘x’ and ‘y’ component of the leg length. For this reason, vertical legs were needed. The current design places the legs at the corners of the vehicle, ensuring the maximum possible stability for a vertical leg configuration.

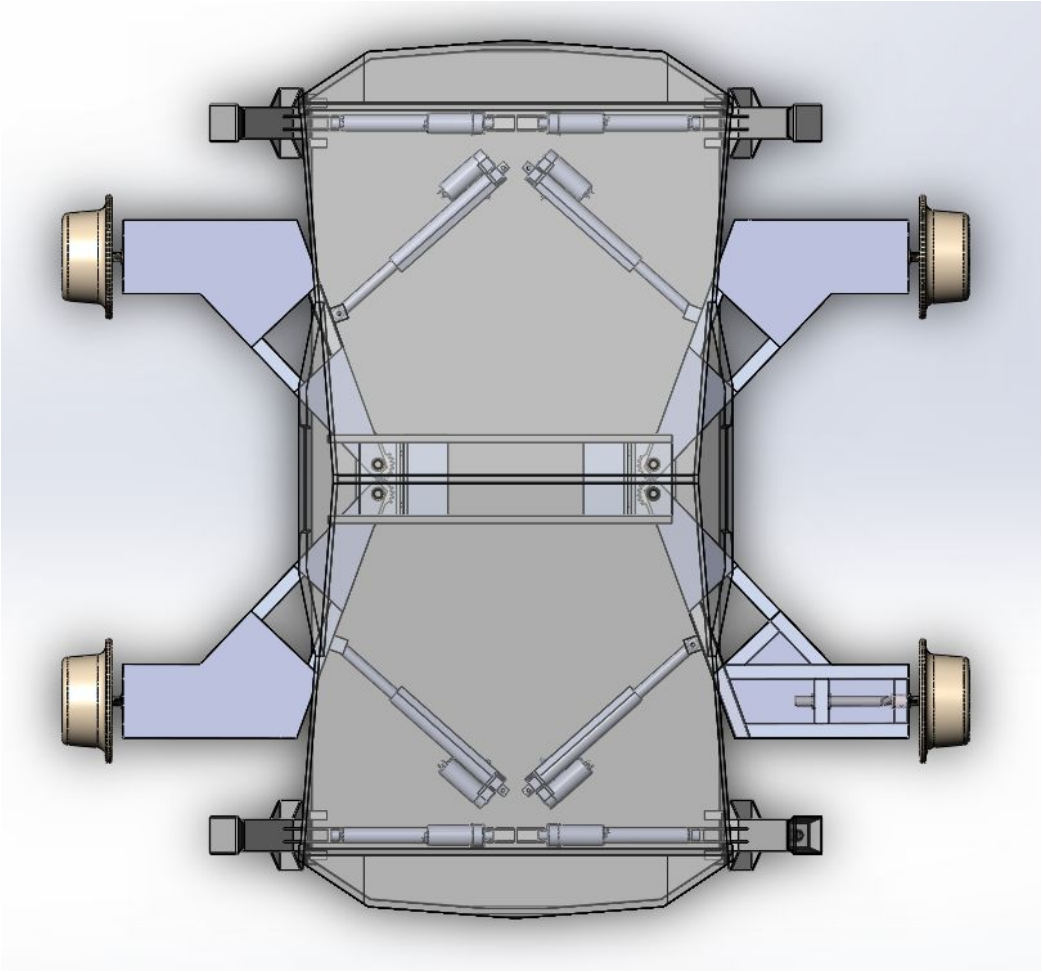


Figure **4** - Leg placement, top view, shown in the retracted (pointed up) position

Additionally, as seen above, two more linear actuators were added to mirror the original two that were controlling the wheel base retraction. This was due to a comment regarding the backlash induced by the original method using a gear to actuate the rear half of the wheel base. While the team was well aware of this issue beforehand, the budget was more of a valid concern at the time, since the backlash would be very low and it would only affect the non driven axle. However, since the prototype will not actually be built, the extra actuators were added to facilitate the retraction process and stabilize the vehicle even further. This will also help with the new simulation plan of four motors instead of two, making all the wheels driven.

Furthermore, concerns directed at the project regarding the tilt on curves and potential instability of the vehicle were found to be insignificant. The wheel flange and its distance from the track greatly limit the amount it can tilt, shift, or twist on the track. This ensures that the sensors reading the measurements do not measure very different portions of the track. Shown below are two worst-case scenarios where the vehicle is shifted to its physical limits in two different ways.

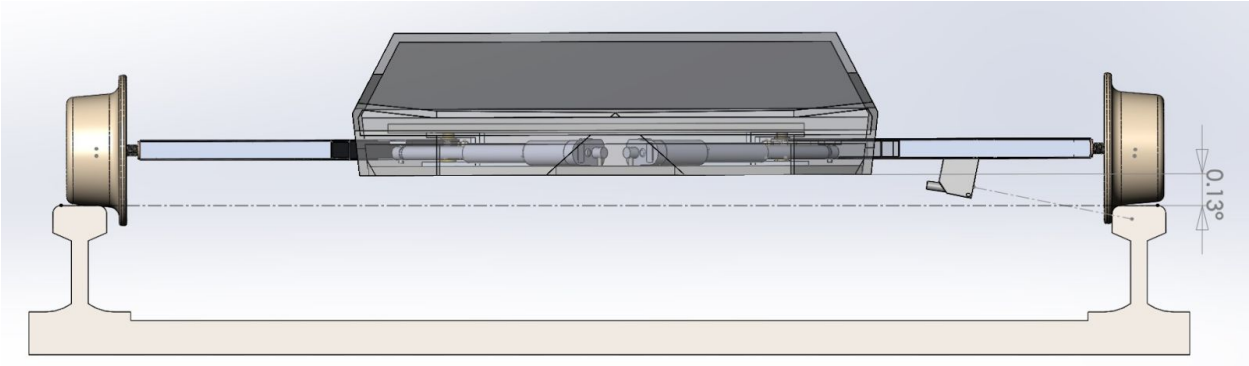


Figure **5** - Maximum tilt from horizontal plane (~0.13 degrees)

The first case is the vehicle shifted all the way to the right, which could happen when taking a curve at high speeds. The flange stops the motion of the vehicle horizontally before it could move far enough to affect the sensor's measurement position. As shown above, the maximum tilt the vehicle could encounter is approximately 0.13 degrees. This moves the measurement area of the sensor by less than a quarter of an inch. Prior to this, the sensor was measuring at the midpoint of the vertical face of the rail. After tilting the maximum amount, it is still measuring that same face with a large margin above it. This shows that the motion of the measurement point is minimal and the concerns regarding it moving off of the rail are invalid.

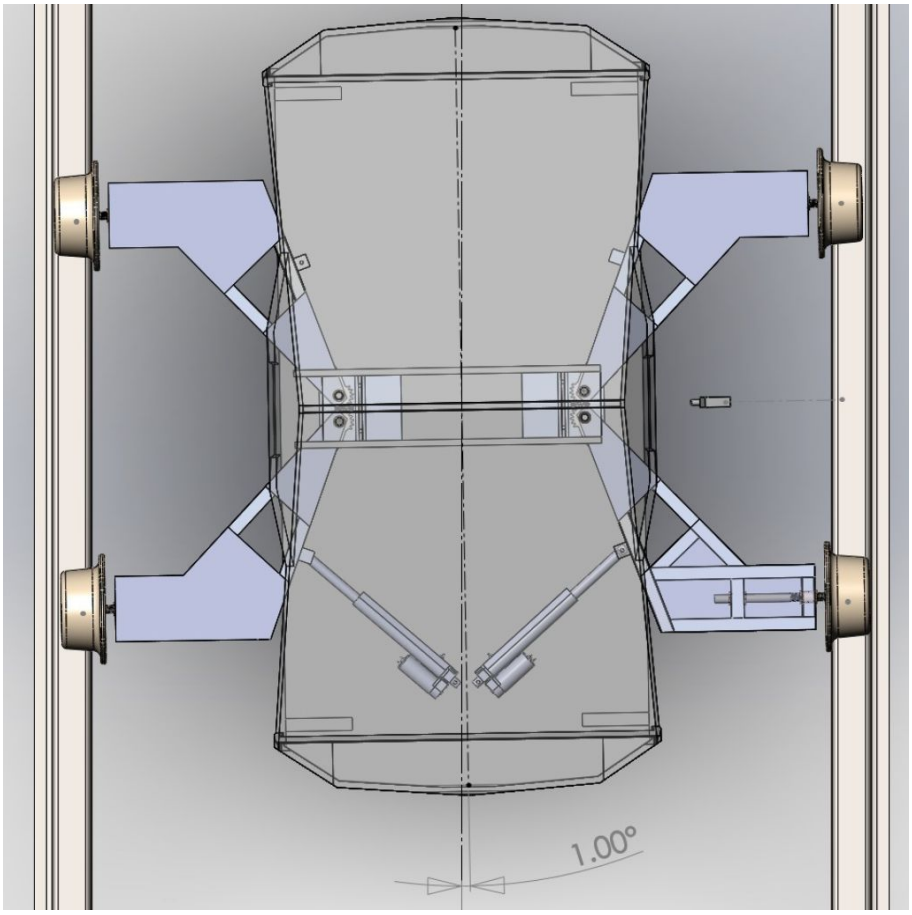


Figure **6** - Maximum skew/twist from the vertical midplane (~1.0 degrees)

The second case shows how the flanges stop twisting motion as well. Shown above is the maximum twist, which comes out to be around 1 degree. This is very small, and both the issues are self correcting due to the shape of the wheels. One point to note is that the vehicle is meant to be a cheaper replacement to the current track geometry cars. Those vehicles can be very precise and accurate because they are multi-million dollar vehicles. The ASIV is meant to cost orders of magnitude less - around $15,000. Measurement data will be affected by tilt and twist, but it will not be nearly enough to invalidate the data. The vehicle is supposed to be deployed as part of a fleet that will inspect the track repeatedly. Over a few measurement passes, a reliable average measurement can be obtained. The current track geometry cars can then be used to supplement the ASIV and measure areas of high defects in more detail. This cuts down operational costs for the MTA/subway systems, as they no longer need to keep the expensive, manned vehicle running 24/7 and will only use it when needed.

The next steps regarding the mechanical design are to create motion simulations with the vehicle moving at its target velocity and use the Sensors feature in SolidWorks to measure the distance between the laser sensor and the track to emulate the function of the sensors. This will show how the readings vary on straight and curved tracks. Worn down models of the track will be made with random variations in gauge and profile to see if the sensors can detect areas that are not within specification and need repair. Braking distances will be analyzed (within the

limitations of the program) to see if additional physical braking is needed to supplement the current motor braking. A sensor mount needs to be finalized, but the necessary positioning of the sensor is known. This mount will ideally be retractable to protect the sensor when the vehicle retracts, but it is not necessary.

Finally, some improvements need to be made to the leg design. A longer extension is preferred for the telescoping portion. Other options for actuating this that can be more compact but offer a longer stroke will be explored.

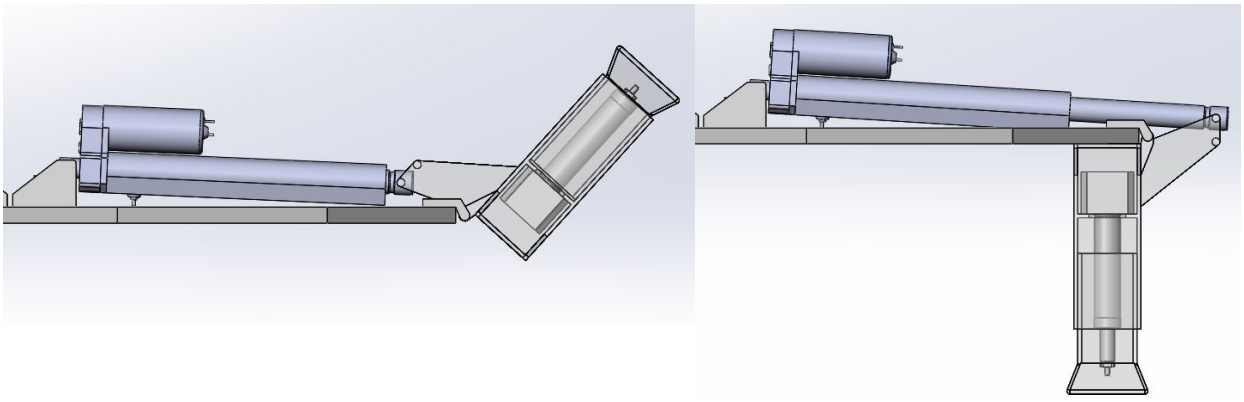


Figure **7** - Leg subsystem retracted (left) and partially extended (right)

Electrical Design

Part 1 - Old and New Objectives and Design:

Old Objectives and Design:

The goal of the electrical team is to “animate” the chassis design and learn about mechatronic and robotic electronics. If brought to market, ASIV could be fully autonomous and be built as a “hive mind” system, etc. However, this is beyond the capabilities of our team, since it consists of five Mechanical Engineering students, and one Electrical Engineering student. A more realistic goal is to learn how to take basic measurements, control the robot remotely, and have some semi-autonomous functionality. Our original objectives would have been as follows:

1. Collect Measurements
    a. Measure
        i. Gauge
        ii. Crosslevel
        iii. Distance from start
    b. Collect the data to be analyzed later

2. Animate the robot
    a. stop/go
        i. varying speed
        ii. Direction
    b. Trigger Evasion + Return to Tracks
    c. Basic obstacle detection

Operation would consist of the following controls:

Measurement:

- Turn measurement collection on/off
    - Gauge on/off
    - Crosslevel on/off
    - Two modes - continuous and distance based
- Move or not move (not autonomous)
    - Direction
    - Speed
- Engage Evasion + Return to Rails

A schematic of the control panel (wireless control) would look as follows:

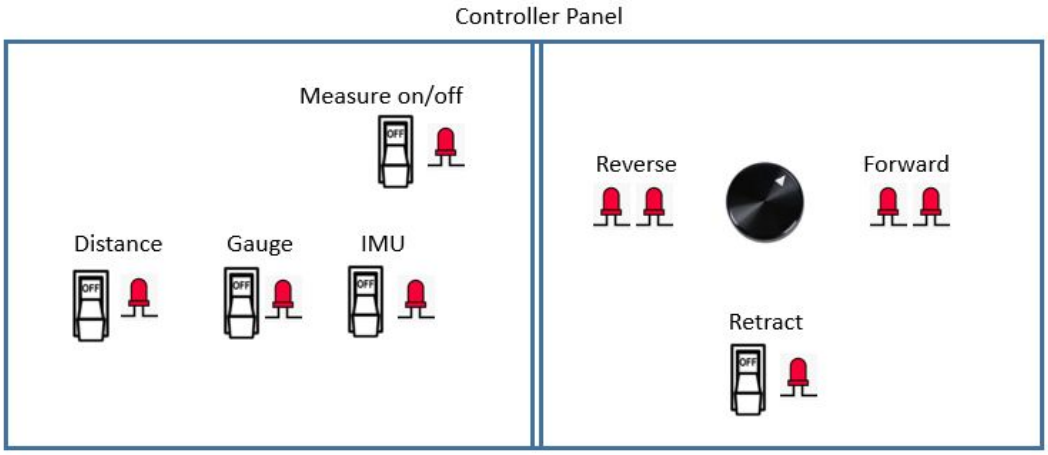


Figure **8** - Controller Panel

The robot would have a similar bank of LEDs for troubleshooting.

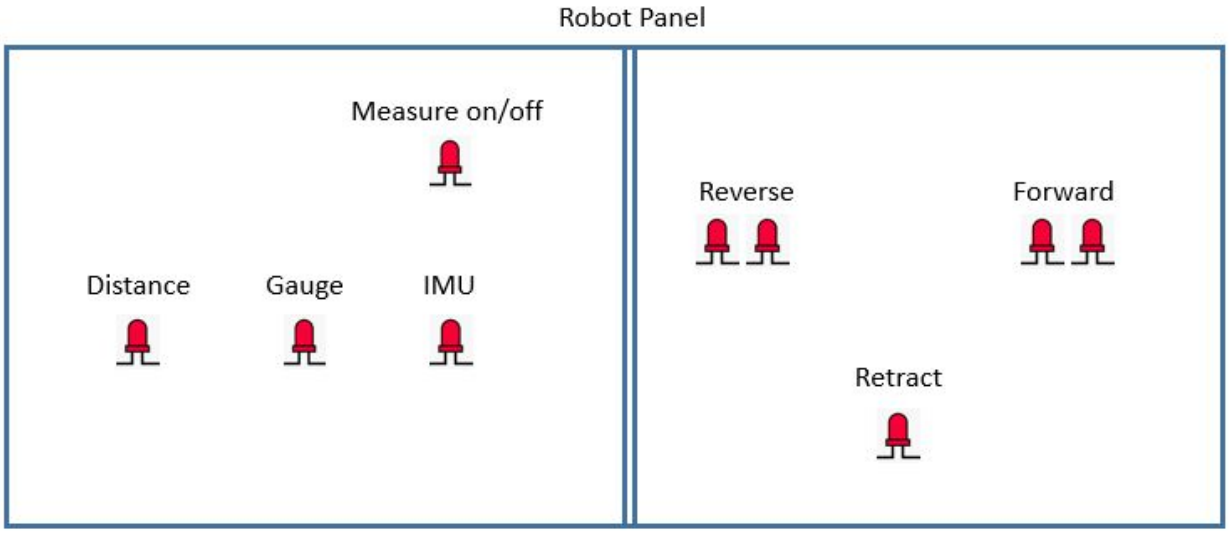


Figure **9** - Robot Panel

General Testing Procedures:

The overall approach to creating the electrical system involved operating a component in isolation, and then adding components together. By working with a component separately, it is easier to see how it functions and troubleshoot problems. From a coding perspective, it helps us think about how this component would best fit into the overall system code and control architecture. This control architecture is another testing outcome, and it includes questions such as, “What microcontrollers should be used?”, “How many?, “How should they be arranged?”, etc. Testing also would involve determining the accuracy of current measurement equipment. We would want to see first if tested components were suitable for a prototype, and second if they are suitable for a production level model.

Bonus Capability:

If a working prototype was built which could be manually driven, collect reliable measurements, and prevent itself from hitting things, we considered adding another feature: a 3D track map. It came to our attention that with the track gauge, angle of the chassis, and driven distance, we could calculate the track as a series of 3D Cartesian points and graph the track in Matlab. This would be an interesting feature and would be a great way for us to practice using collected data for a more complex task.

Old Deliverables:

1. Code
2. Circuit Diagrams
3. Built Circuit

New Objectives and Design:

Our new objective is similar to the original, minus any physical testing. Some physical testing began, and that is presented. In some ways, we are free to design the prototype with more parts, since we cannot meet in person to build it. In essence, we will have the following deliverables:

1. Code
2. Circuit Diagram
3. Suggestions for testing

The goal is to present the electrical design as it would have been, and provide a "cookbook" so that could be used to create a physical prototype.

Part 2 - Detailed Description of Robot Design and Testing:

In this section, we present our tests and system design since Phase 4, as well as what we expect the final circuitry, code, and control architecture to be.

Measurement:

Gauge:

Gauge is the width between the rails. This is shown in Figure 10. To measure gauge, we have two laser distance sensors donated to us from Micro-Epsilon. These are placed on either side of the robot chassis and the distance each sensor measures together with the distance between the two sensors is the gauge. The gauge is defined in Figure 10. d1 and d2 begin at the face of the sensor where the laser exits.

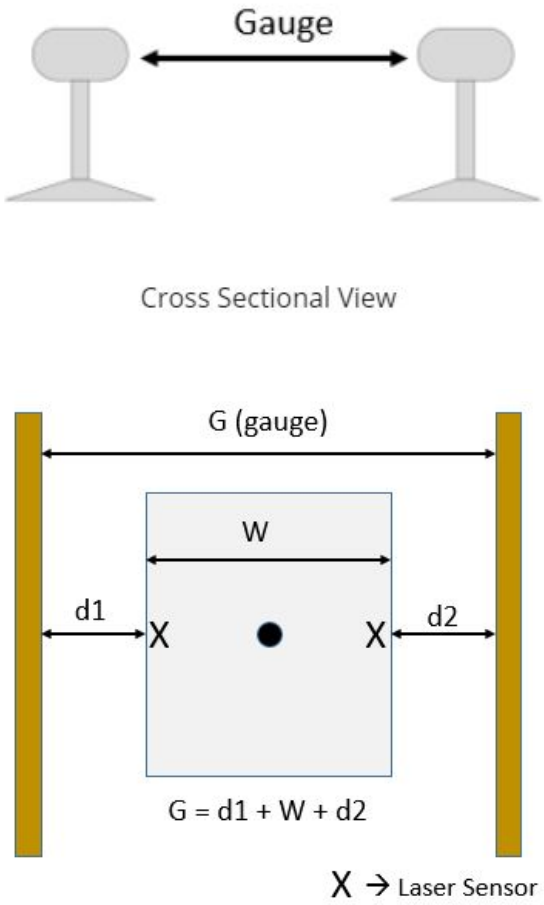


Figure **10** - Gauge definition

The circuit configuration of the laser sensor is shown in Figure 11.

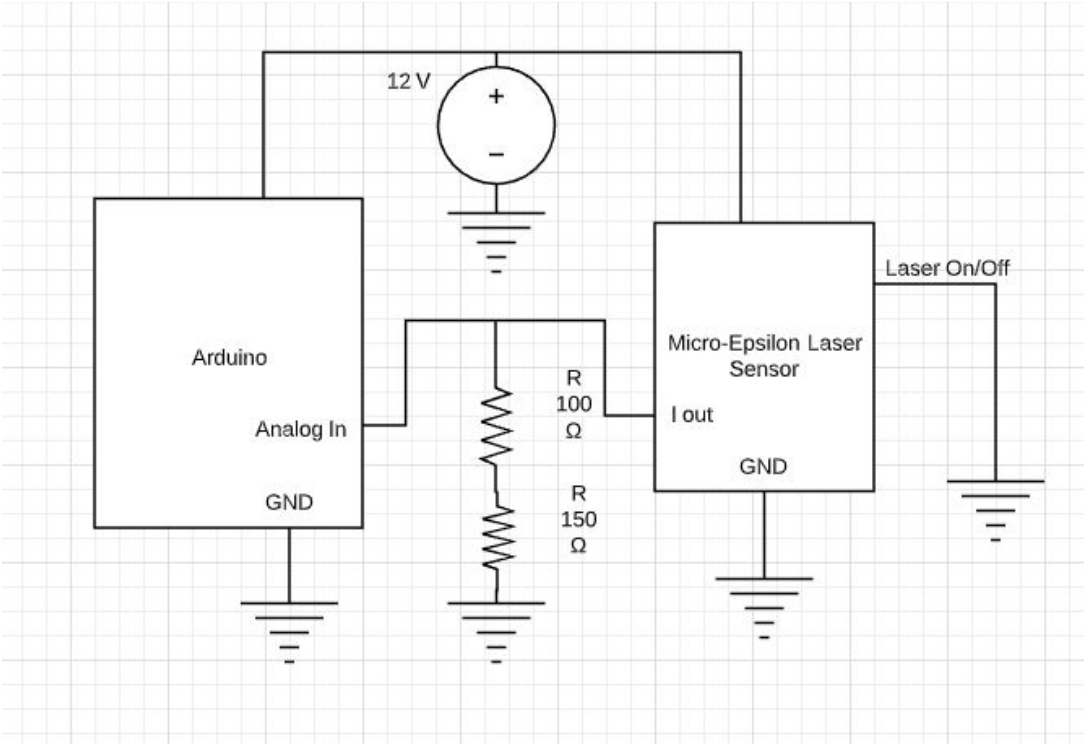


Figure **11** - Laser Sensor Schematic

Using the Laser Sensor:

I out varies from 1-5V when placed over a 250 ohm load (as shown). By default, it measures from 60mm to 260mm and 60mm corresponds to low voltage. Using a teach button on the body of the laser, high voltage can be adjusted to correspond to 60mm, and the measuring range can be shortened. When a target is out of range, a specific value is given which is outside of the normal range of voltage out, so an out of range target is easily identified. Resistors with a ±1% variation are used.

To determine the quality of this set up, the following experiment is proposed to test steadiness of output, and consistency of output:

1. Use default settings
2. Monitor the fluctuation in Analog In readings from the Arduino for 60mm, midrange, 260mm, and out of range over the span of 30 sec
3. To be repeated five times:
    a. Place ruler in front of sensor
    b. Choose 5 distances
    c. Place object at each distance for 10 sec
    d. Record average Analog In reading for each distance
4. Compare each of five averages for all five distances

This system we were able to put together, but not test. Quick inspection shows that values are largely steady.

Crosslevel:

Crosslevel is the height of one rail above the other (Figure 12). We will measure it using the angle determined from an IMU, the measured gauge at that location, and trigonometry.

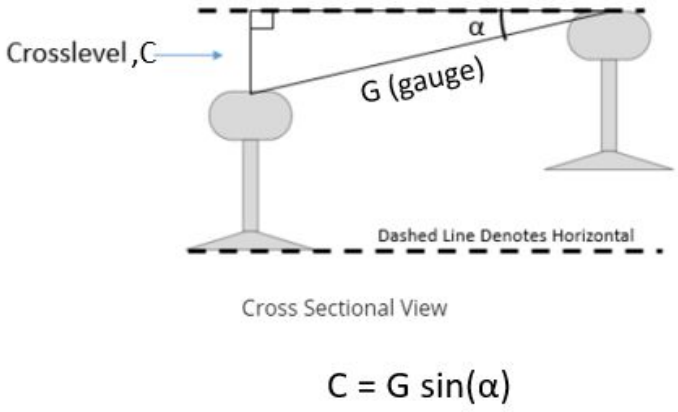


Figure **12** - Crosslevel definition

The angle is measured using the Pololu minIMU v5. This is a board which contains a magnetometer and a gyroscope/accelerometer combination. We still need to learn about the process, but there should be algorithms and pre-built libraries to determine absolute angle relative to gravitational pull using the acceleration and angular velocity readings. The board gives an output of acceleration in three axes and angular velocity about three axes. Specifics of our findings are in the Appendix A2. The board communicates with the Arduino using I2C as shown in Figure 13.

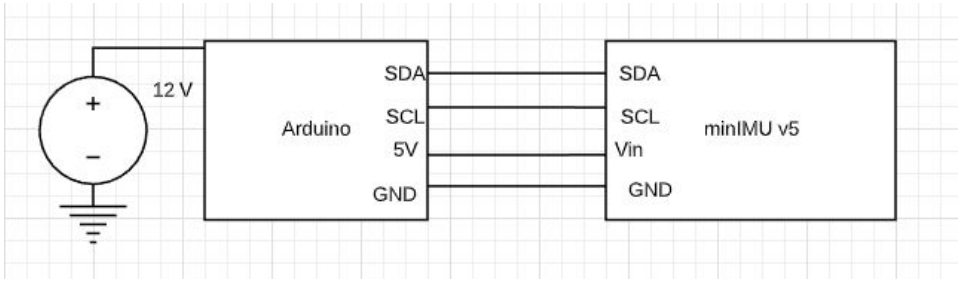


Figure **13** - IMU schematic

A proposed test for the IMU repeatability and measurement fluctuation is outlined as follows.
Accelerometer:

1. For each axis
    a. Hold steady for 15 seconds and record reading fluctuation
    b. Flip over and repeat
    c. Record average value for each direction

2. Repeat above three times and compare averages to determine repeatability

Gyroscope:

1. For each axis
    a. Hold steady for 15 seconds and record reading fluctuation
    b. Record average value for each direction
2. Repeat above three times and compare averages to determine repeatability

We were able to test IMU briefly, but not formally. Results are found in Appendix A2. At a glance, measurements seem steady enough for prototyping.

Distance:

To track distance, we were considering a magnetic encoder from Pololu. It is meant to be placed on a small hobby-grade DC motor, so the intention was to connect the robot shaft of one wheel to a small motor or dummy axle via gears. These quadrature encoders can be found here: https://www.pololu.com/category/157/encoders-for-micro-metal-gearmotors.

More research must be done to see how these would be read in the microcontroller architecture.

Data Collection:

To collect our data for future use, we have purchased a DataLogger Shield from Adafruit (Figure 14). This connects to an Arduino's ICSP pins and Digital Pin 10. It can read to and write from an SD card and has a battery operated Real Time Clock (RTC) for timestamping information. We use it to write measurement values as a CSV file which can be opened in Excel. Our code is written to create a new file each time the Reset button is pressed or the Arduino is booted up.

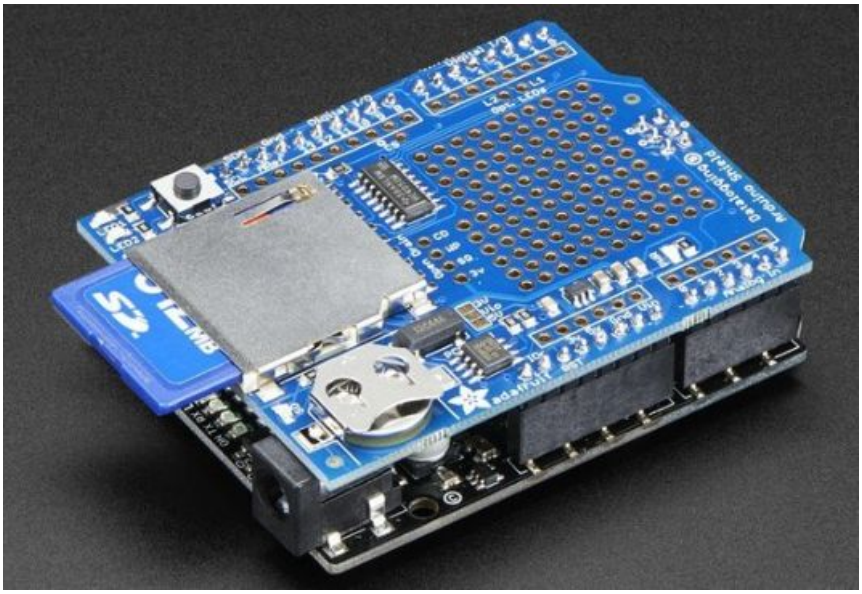


Figure **14** - Datalogger shield

While using the board, we have found the RTC can drift, and so this is not a reliable piece of equipment for a production model. However, it seems the timestamps are accurate in short time spans during tests on a particular day. It is when we return to the unit after a few day when we notice an error. To formally test short term synchronization between timestamps and our measurements, the unit can be run as follows:

1. Sync time with external computer's clock
2. Create obvious disturbance in respective sensor and note external computer time
3. Check later to see if the timestamp at the disturbance corresponds to the time recorded from the external computer

To test time accuracy over time, check the RTC time over the course of several days.
Animating the Robot:

Motor Control:

To control the motors, we have purchased two BTS7960 High Current 43A H-Bridge Motor Drivers. They are on a board which allows one to choose a motor direction and speed via a PWM signal. (Figure 15)

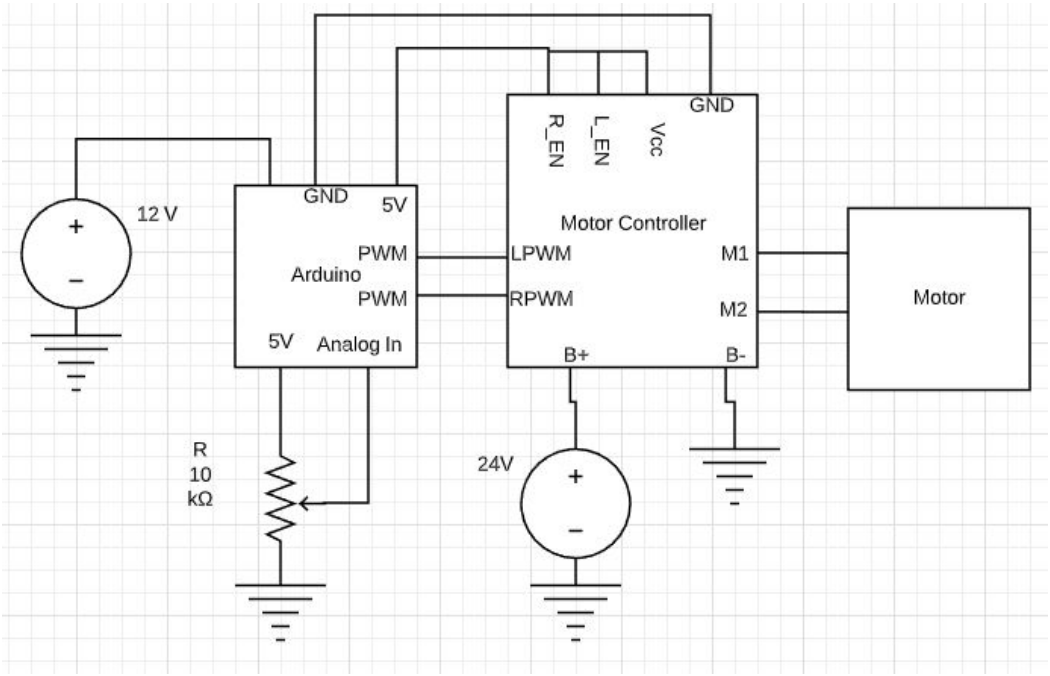


Figure **15** - Motor Schematic

Wireless Control:

To wirelessly control the robot, an HC-12 Radio module is used. This can be used as a serial output for Arduino. The schematic for the module is shown in Figure 16. We have found range and communication to be reliable.

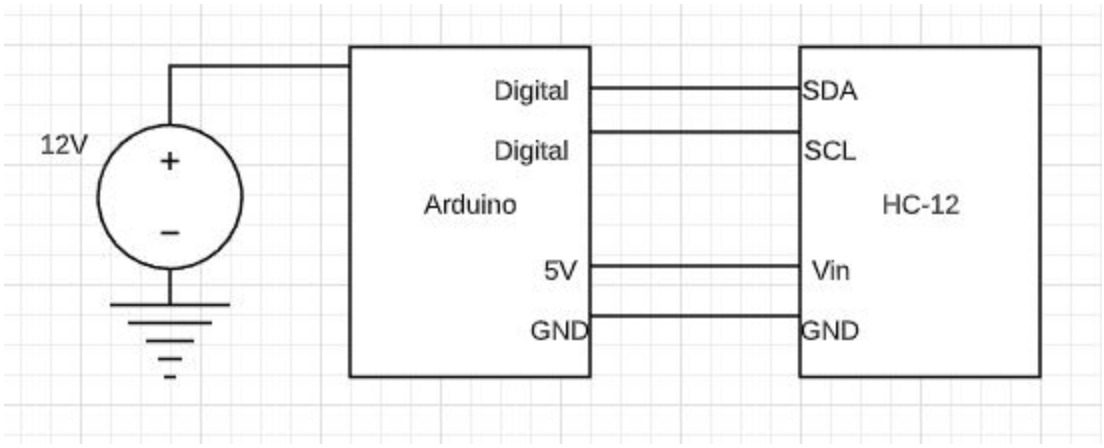


Figure **16** - Wireless Transceiver Schematic

Evasion:

Details of evasion circuitry are still to come, but it will involve linear actuators.

Obstacle Detection:

Obstacles can be avoided using a long range IR distance sensor. We have purchased the VL6180x, but there is a long range version which would work better. The schematic is as shown in Figure 17.

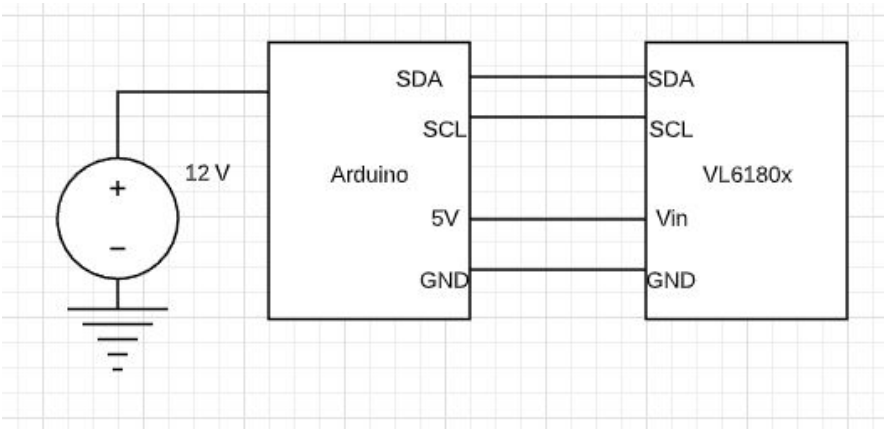


Figure **17** - IR ToF Schematic

The overall control architecture would likely have the following characteristics:

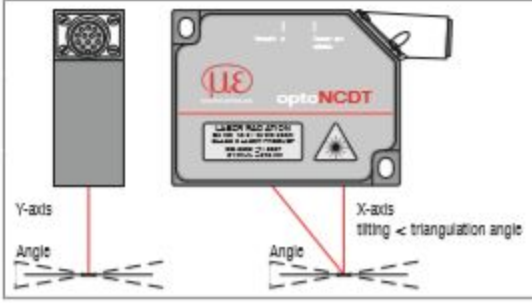
1. One Arduino manages measurement and data collection
2. One Arduino resides in user control panel to send commands
3. One Arduino handles locomotion, evasion, and activating/deactivating measurement Arduino (this Arduino would also receive user commands)

Available Code:

The available code we have written is in the Appendix. It includes overall code for our Alpha prototype, which could write two laser distance measurements to an SD card.

Mathematical Kinematic Proofs:

The Micro-Epsilon laser sensors are affected adversely by degrees of tilt according to Figure 18 from the equipment manual.

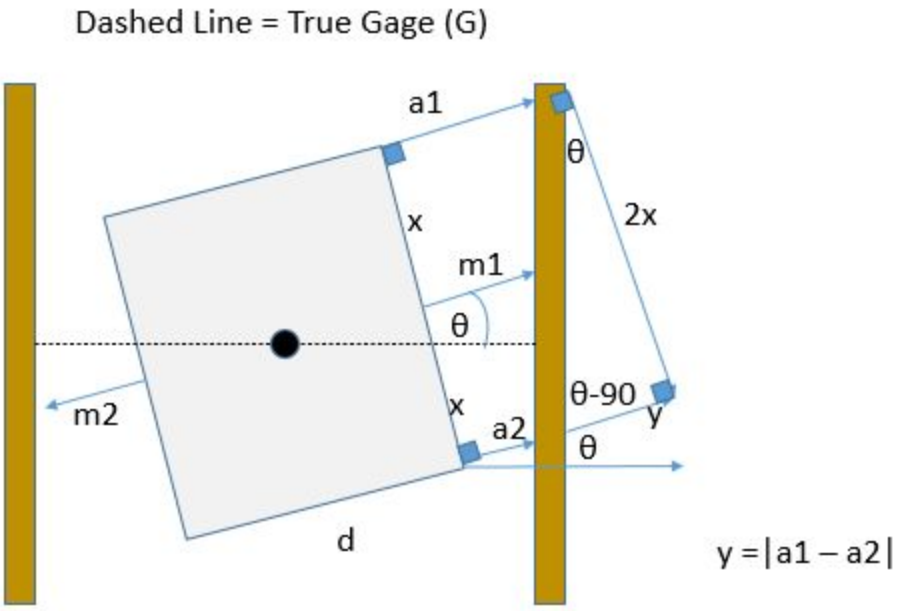


Angle	X-axis %	Y-axis %
±5 °	typ. 0.12	typ. 0.12
±15 °	typ. 0.2	typ. 0.2
±30 °	typ. 0.5	typ. 0.5

Figure **18 -** Laser Sensor Measurement Error

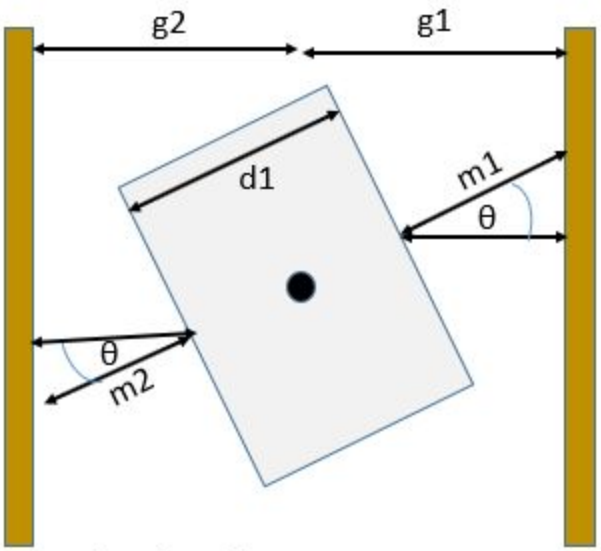
Two proofs are presented to give basic predictions as to vehicle tilt during motion. The first is for change in gauge due to tilt looking overhead. The second is for tilt around the axis in the direction of motion. Discussion of results follows.

Proof 1: Rotation around z-axis



$$\text{Tan}(\theta) = y / 2x = |a1 - a2| / 2x$$

$$G = (m1+m2+d)*\cos\theta$$



$$g1 = (m1 + d1/2)*\cos(\theta)$$

$$g2 = (m2 + d1/2)*\cos(\theta)$$

$$G = m1\cos(\theta) + m2\cos(\theta) + d1$$

$$\boxed{G = (m1+m2 +d1)\cos(\theta)}$$

$$G = g1 + g2$$

$$g1 / (m1 + d1/2) = \cos(\theta)$$

Figure **19** - Calculating twist of vehicle around Z

Qualitative Error Analysis:

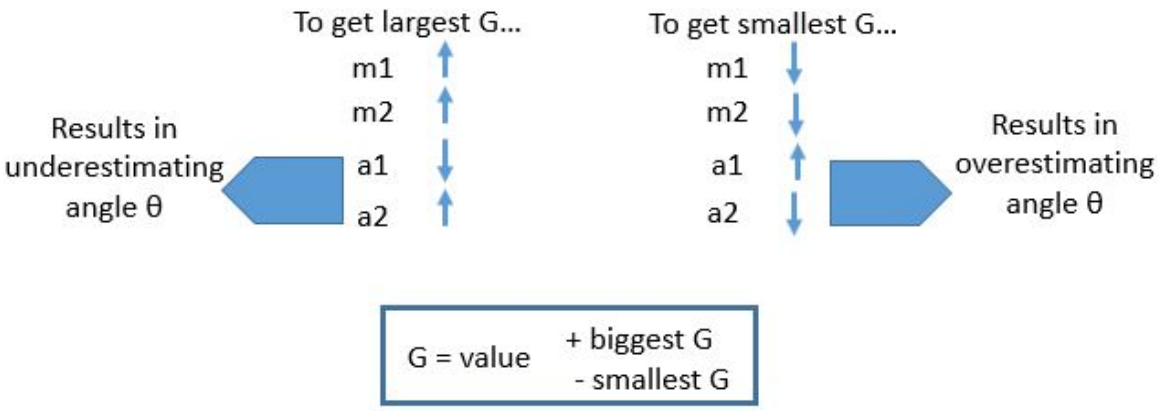


Figure **20** - Error Analysis

Proof 2: Rotation about x-Axis:

Axes change as vehicle tilts

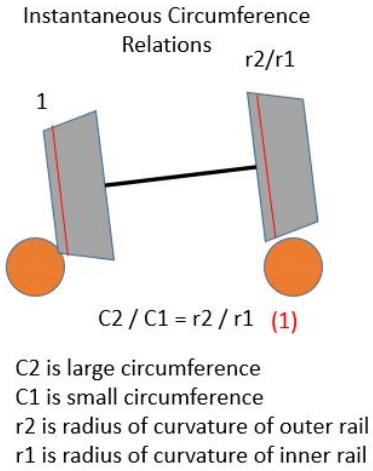


Figure **21** - Tilt Calculation

Six laser sensors can be used to determine radius of curvatures.

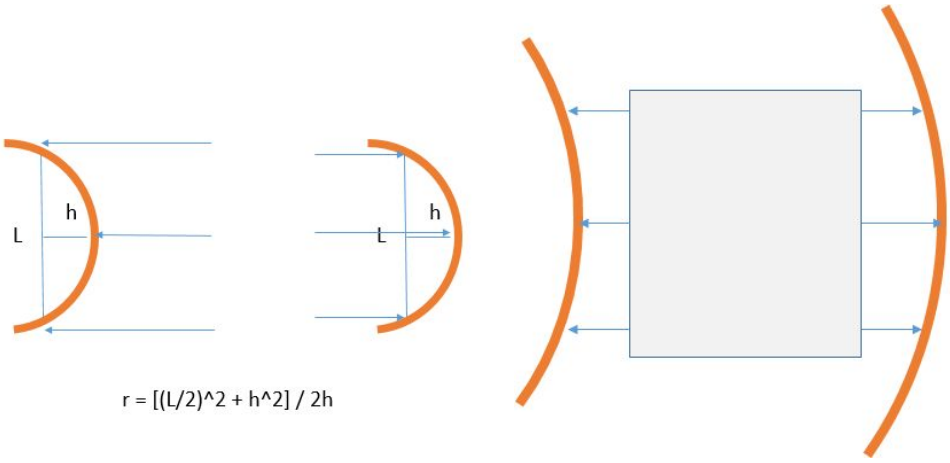
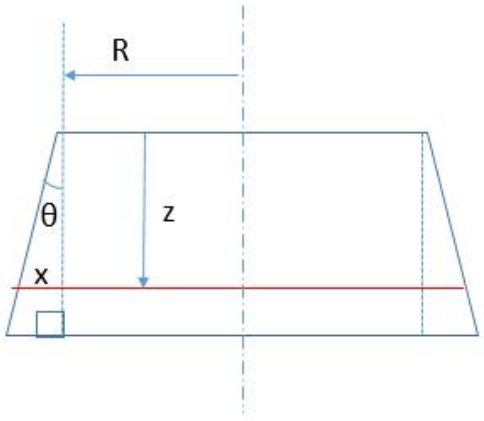


Figure **22** - Laser Placement

Wheel Geometry:



$C(z, \theta) = 2\pi(R + z^{*}\tan\theta)$

Red Line is Instantaneous Circumference, C

Figure **23** - Wheel Geometry

Angle of Tilt:

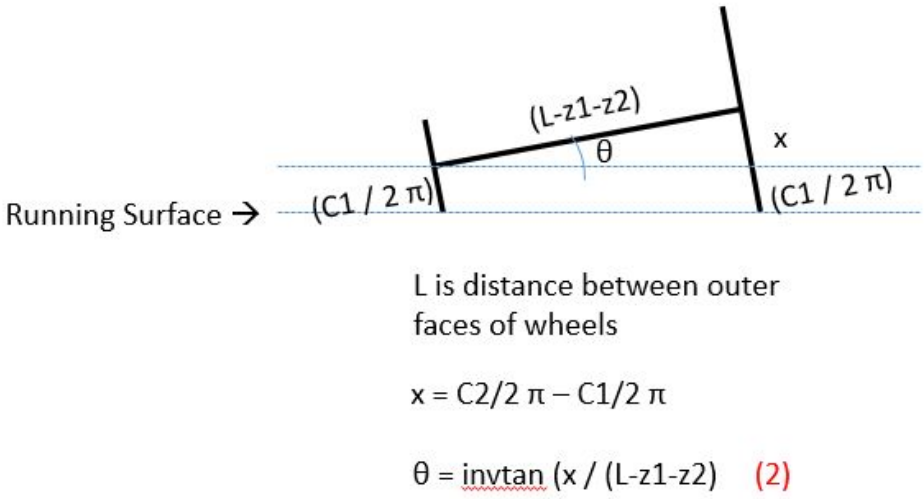


Figure **24** - Tilt angle calculation

Discussion:

In the end, the rotation around the z-axis can be estimated and the gauge corrected. However, the rotation around the x-axis analysis leads to two equations (marked in red) and three unknowns (the two instantaneous circumferences and the angle of tilt). The maximum theoretical tilts for both axes are shown in Figures 5 and 6 using our CAD model, and show that the expected tilts are quite small.

Project Plan

Deliverables

Deliverable	**Description**
Transition to Virtual Prototype Development	With the transition to a virtual expo due to Covid-19, the project will resume as a theoretical model of the final prototype. All work will be continued online and made to be presented virtually.
Mechanical Design Components	The mechanical aspects of the Beta Prototype such as the CAD model, layout and drive mechanism code will be created.
Electrical Design Components	The electrical code and circuit diagram will be finalized for the beta prototype. These systems will be presented in a “cookbook” way to replace the physical model.
Innovation Expo Preparation	The project will be constructed to present virtually for the innovation expo. Suggestions will be created to show the theoretical creation of the beta prototype.

With the virus putting a complete pause on physical prototype creation, the project will continue virtually where the prototype will showcase the design of both the electrical and mechanical components. The Gantt Chart below details the new course of action for Phase 5 with more focus on the design aspect of the vehicle. Since a physical prototype can not be created, the focus will go into the CAD model to detail where each component will go and to display the different mechanisms within the system. All testing for motion will be examined in Solidworks while the electrical components will be tested separately to show how each sensor works. The individual sensors will receive test data to show functionality and a theoretical guide to create the physical prototype will be made in place of the original plan.

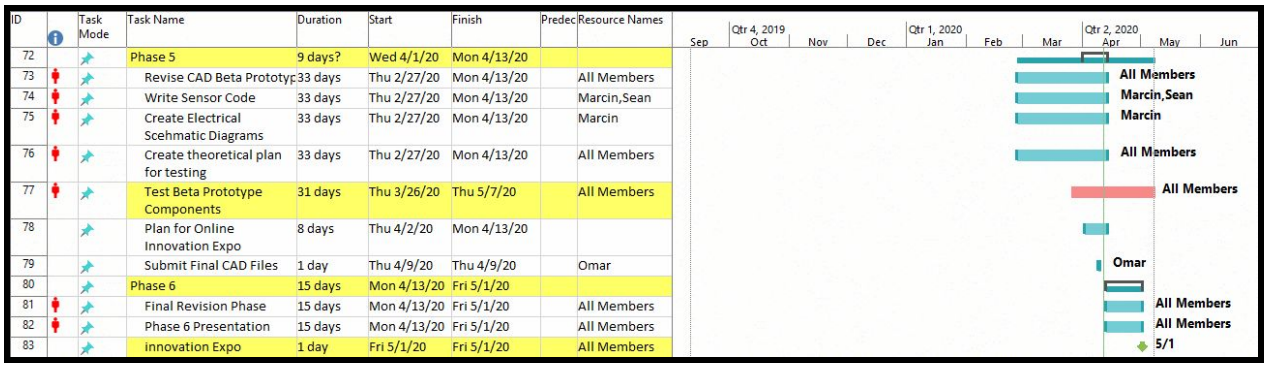


Figure 25: Phase Five to Phase Six Gantt Chart

The table below shows the current up to date budget prior to the project transitioning to a virtual platform. This table shows the items that were already purchased as well as the remaining budget left. Since a physical prototype is no longer required, there is no need to purchase the remaining components. These components will be implemented in the CAD model to further show the complete functionality of the vehicle. However, the design of the prototype can now be maximized since there is no budget constraint. The beta prototype will now represent an enhanced theoretical model with the ideal specifications for the components such as the motors and batteries. This change will allow for a complete representation for the ideal prototype.

Table 5: Up-to-date Budget

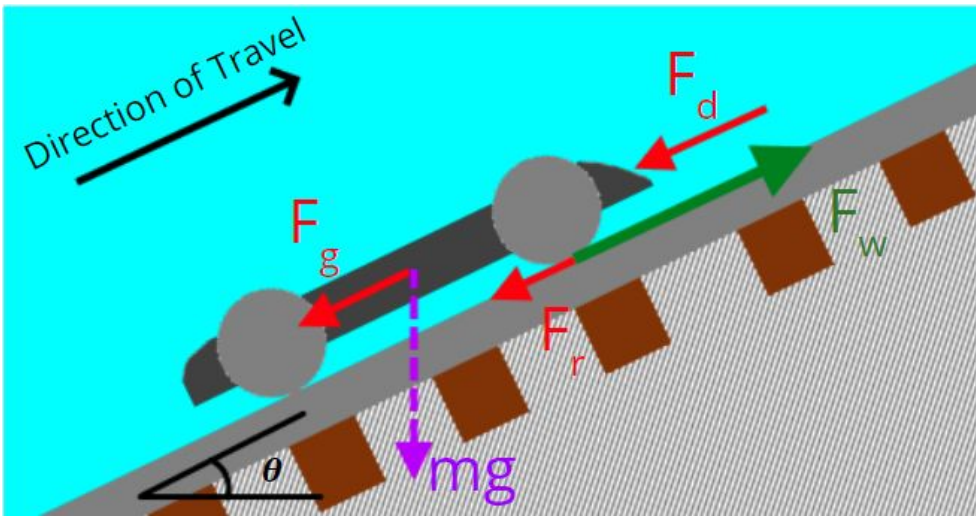
Up-to-Date Budget	
Purchased Sensors and Parts	
Item	**Price**
Laser Sensors	Donation from Micro-Epsilon
Gyro & Accelerometer Unit	$15.95
IR Time-of-Flight Sensor (x3)	$25.47
M12 12-Pole Cables (x2)	$72.00
Adafruit DataLogger Arduino Shield + Button Battery + Stacking Headers	$27.49
Arduino Mega 2560 Microcontroller Rev3	$40.10
SS 304 ⅝” Rod (x2)	$26.50
Square Tube (x5)	$56.25

24V Electric Motor (x2)	$45.99
Pillow Block Bearings (x4)	$22.58
12 V 7AH SLA Battery (x2)	$34.40
12 V Battery Charger	$9.99
Total Cost w/ Sales Tax	**$426.05**
Remaining Budget:	**$273.95**

Appendix

A1 - Motor-Battery Analysis

Motor/Battery Analysis and Formulas



The forces associated with travel up an incline for the vehicle can be summarized as follows:

$$\sum F_x = ma = F_w + F_g + F_d + F_r$$

$$F_w = Force\ at\ Wheel = \frac{n\tau_w}{D/2}$$

$$F_g = Component\ of\ Gravitational\ Force\ on\ Slope = mg\sin\theta$$

$$F_d = Drag\ Force = \frac{1}{2}c_d\rho v^2 A$$

$$F_r = Rolling\ Resistance = C_R mg$$

The terms can be rearranged to solve for the input torque at each wheel.

$$\tau_w = \frac{D}{2n}\left(ma - mg\sin\theta - \frac{1}{2}c_d\rho v^2 A - C_R mg\right)$$

When using the SI units, the following relationship is true for an electric motor's torque constant and voltage constant respectively. The rated and no-load values are taken directly from the specified motor's data sheets when provided.

$$k_T = \frac{\tau_{rated}}{I_{rated}} = k_e = \frac{V_{rated}}{\omega_{no\ load}}$$

The motor voltage constant can be substituted into the following relationship to solve for each motor's current draw at a given torque.

$$I_m = \tau_w \left(\frac{V_{rated}}{\omega_{no\ load}} \right)^{-1} + I_{no\ load}$$

This value of current can then be divided from a DC power source's capacity (in amp-hours) to find the battery life in hours if said current were to be held constant for that duration.

A2 - IMU Guide

IMU Guide
Device Name:
Pololu MinIMU-9 v5
Purchase Location and Helpful Information:
https://www.pololu.com/product/2738
Description:

- Contains accelerometer/gyroscope combo (LSM6DS33)
- Contains magnetometer (LIS3MDL)

Wiring:
Code:

Library Location:

https://github.com/pololu/lsm6-arduino

Operation:

- Measurements Used (6):
  - Linear Acceleration x,y,z
  - Angular Velocity x,y,z
  - Magnetic Field (not used)
- For acceleration and angular velocity, output is a signed 16bit number (-32,768 to 32,767)
- Sensitivities mapped over the signed 16bit number
  - Acceleration +- 2,4,6,8 g (default 2g)
  - Angular Velocity +- 125,245,500,1000,2000 degrees per sec (dps) (default unknown)

Testing:

- At rest, acceleration (or angular velocity? Can't remember) raw values fluctuate by 200 – 400 on default setting (2g or 125 dps)
- Change of sign indicates opposite direction

A3 - Alpha Prototype Code

```
//Writes a count to an SD card with timestamp and seconds since startup
//adjusts RTC to compiler time (if that line is uncommented)
//writes to serial current file

//Outside loop setup for RTC
#include "RTClib.h"
RTC_PCF8523 rtc;

//outside loop setup for sd
#include <SD.h>
const int chipSelect = 10; //for comm with new Adafruit datalogger
File distDataFile; //create file class to allow for repeated opening
char filename[] = "dist00.CSV"; //first name to try

//outside loop variables for Micro_Ep 1
byte distPin1 = A0;
int distRaw1;

//outside loop variables for Micro_Ep 2
byte distPin2 = A1;
int distRaw2;

void setup() {
  // put your setup code here, to run once:
Serial.begin(9600);

//start RTC
rtc.begin();
/*if(!rtc.initialized()){
rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));//give it compiler time (time of computer)
}*/

//initializing the SD
pinMode(chipSelect, OUTPUT);
SD.begin(chipSelect);

//create a new file
for(int i = 0; i < 100; i++){ //keep searching until name not taken
  filename[4] = i/10 + '0';//Are '0's neccessary ?
  filename[5] = i%10 + '0';
  if (!SD.exists(filename)){
    distDataFile = SD.open(filename, FILE_WRITE);
    distDataFile.close();
```

```
    Serial.print("Current File: ");
    Serial.println(filename);
    break;
  }
}
}

void loop() {
  // put your main code here, to run repeatedly:
DateTime now = rtc.now(); //process for capturing readable timestamp

//Read dist from Micro_Ep1 and 2. Write to serial
distRaw1= analogRead(distPin1);
distRaw2 = analogRead(distPin2);
Serial.print(distRaw1);
Serial.print(", ");
Serial.println(distRaw2);


distDataFile = SD.open(filename, FILE_WRITE);
//TimeStamp
distDataFile.print(now.year());
distDataFile.print('/');
distDataFile.print(now.month());
distDataFile.print('/');
distDataFile.print(now.day());
distDataFile.print(' ');
distDataFile.print(now.hour());
distDataFile.print(':');
distDataFile.print(now.minute());
distDataFile.print(':');
distDataFile.print(now.second());
//RunTime
distDataFile.print(",");
distDataFile.print(millis()/1000);
//Data
distDataFile.print(',');
distDataFile.print(distRaw1);
distDataFile.print(',');
distDataFile.println(distRaw2);
distDataFile.close();
//delay(100);
}
```

A4 - HC-12 Code

Receiver Code

```
#include <SoftwareSerial.h>
SoftwareSerial hc12(9, 3);

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  hc12.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  if (hc12.available()) {
    //capture
    int c = hc12.read();
    //write to serial
    Serial.print("You received: ");
    Serial.println(c);
  }
}
```

Transmitter Code

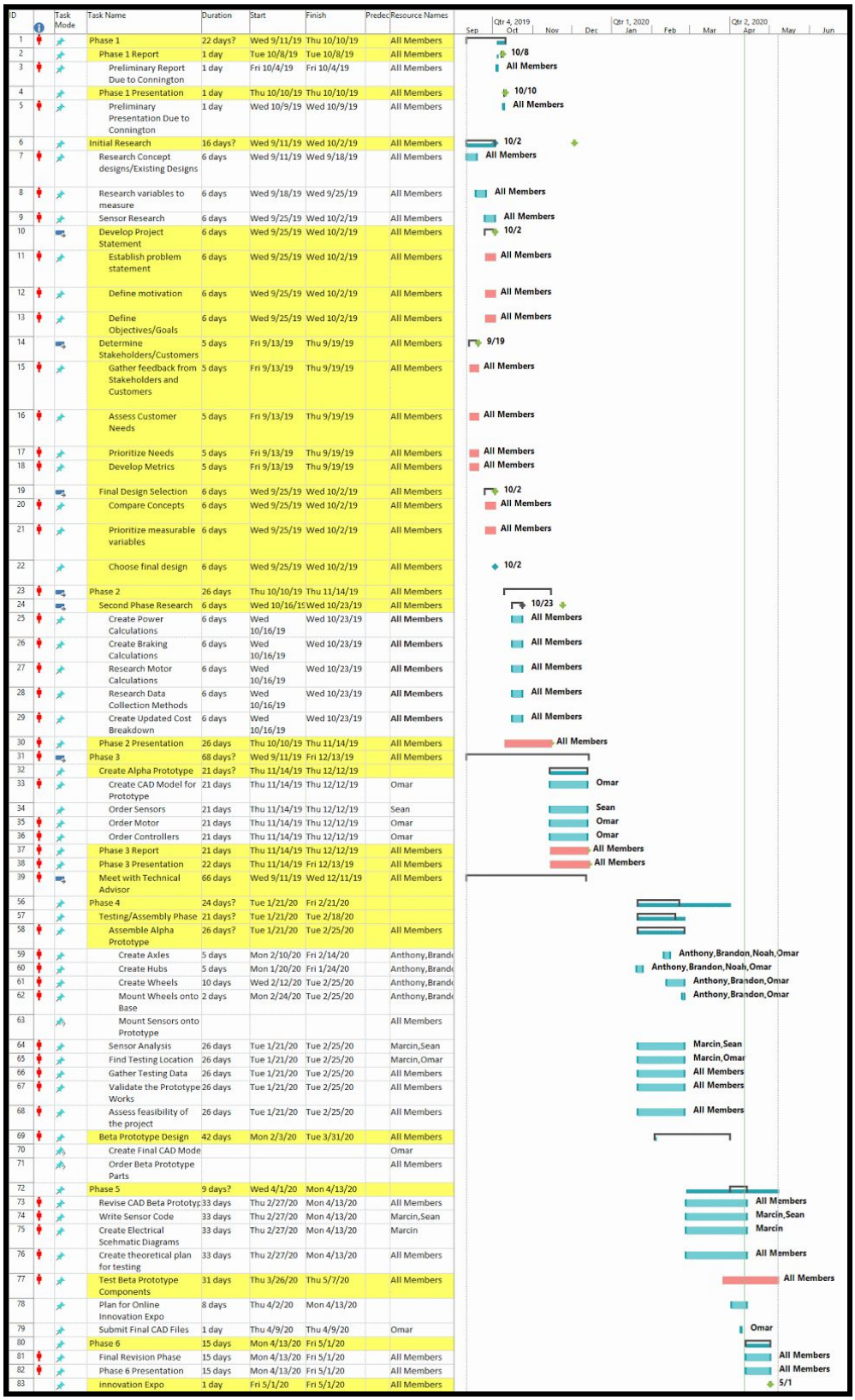
```
#include <SoftwareSerial.h>
SoftwareSerial hc12(10, 11);

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  hc12.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  if (Serial.available() > 0) {
    //capture
    int c = Serial.read();
    //write to serial
    Serial.print("You sent: ");
    Serial.println(c,BIN);
    //write to hc12
```

```
    hc12.write(c);
  }
}
```

A5 - Full Gantt Chart



ID	Task Mode	Task Name	Duration	Start	Finish	Predec	Resource Names
1		Phase 1	22 days?	Wed 9/11/19	Thu 10/10/19		All Members
2		Phase 1 Report	1 day	Tue 10/8/19	Tue 10/8/19		All Members
3		Preliminary Report Due to Connington	1 day	Fri 10/4/19	Fri 10/4/19		All Members
4		Phase 1 Presentation	1 day	Thu 10/10/19	Thu 10/10/19		All Members
5		Preliminary Presentation Due to Connington	1 day	Wed 10/9/19	Wed 10/9/19		All Members
6		Initial Research	16 days?	Wed 9/11/19	Wed 10/2/19		All Members
7		Research Concept designs/Existing Designs	6 days	Wed 9/11/19	Wed 9/18/19		All Members
8		Research variables to measure	6 days	Wed 9/18/19	Wed 9/25/19		All Members
9		Sensor Research	6 days	Wed 9/25/19	Wed 10/2/19		All Members
10		Develop Project Statement	6 days	Wed 9/25/19	Wed 10/2/19		All Members
11		Establish problem statement	6 days	Wed 9/25/19	Wed 10/2/19		All Members
12		Define motivation	6 days	Wed 9/25/19	Wed 10/2/19		All Members
13		Define Objectives/Goals	6 days	Wed 9/25/19	Wed 10/2/19		All Members
14		Determine Stakeholders/Customers	5 days	Fri 9/13/19	Thu 9/19/19		All Members
15		Gather feedback from Stakeholders and Customers	5 days	Fri 9/13/19	Thu 9/19/19		All Members
16		Assess Customer Needs	5 days	Fri 9/13/19	Thu 9/19/19		All Members
17		Prioritize Needs	5 days	Fri 9/13/19	Thu 9/19/19		All Members
18		Develop Metrics	5 days	Fri 9/13/19	Thu 9/19/19		All Members
19		Final Design Selection	6 days	Wed 9/25/19	Wed 10/2/19		All Members
20		Compare Concepts	6 days	Wed 9/25/19	Wed 10/2/19		All Members
21		Prioritize measurable variables	6 days	Wed 9/25/19	Wed 10/2/19		All Members
22		Choose final design	6 days	Wed 9/25/19	Wed 10/2/19		All Members
23		Phase 2	26 days	Thu 10/10/19	Thu 11/14/19		All Members
24		Second Phase Research	6 days	Wed 10/16/19	Wed 10/23/19		All Members
25		Create Power Calculations	6 days	Wed 10/16/19	Wed 10/23/19		All Members
26		Create Braking Calculations	6 days	Wed 10/16/19	Wed 10/23/19		All Members
27		Research Motor Calculations	6 days	Wed 10/16/19	Wed 10/23/19		All Members
28		Research Data Collection Methods	6 days	Wed 10/16/19	Wed 10/23/19		All Members
29		Create Updated Cost Breakdown	6 days	Wed 10/16/19	Wed 10/23/19		All Members
30		Phase 2 Presentation	26 days	Thu 10/10/19	Thu 11/14/19		All Members
31		Phase 3	68 days?	Wed 9/11/19	Fri 12/13/19		All Members
32		Create Alpha Prototype	21 days?	Thu 11/14/19	Thu 12/12/19		
33		Create CAD Model for Prototype	21 days	Thu 11/14/19	Thu 12/12/19		Omar
34		Order Sensors	21 days	Thu 11/14/19	Thu 12/12/19		Sean
35		Order Motor	21 days	Thu 11/14/19	Thu 12/12/19		Omar
36		Order Controllers	21 days	Thu 11/14/19	Thu 12/12/19		Omar
37		Phase 3 Report	21 days	Thu 11/14/19	Thu 12/12/19		All Members
38		Phase 3 Presentation	22 days	Thu 11/14/19	Fri 12/13/19		All Members
39		Meet with Technical Advisor	66 days	Wed 9/11/19	Wed 12/11/19		All Members
56		Phase 4	24 days?	Tue 1/21/20	Fri 2/21/20		
57		Testing/Assembly Phase	21 days?	Tue 1/21/20	Tue 2/18/20		
58		Assemble Alpha Prototype	26 days?	Tue 1/21/20	Tue 2/25/20		All Members
59		Create Axles	5 days	Mon 2/10/20	Fri 2/14/20		Anthony,Brandc
60		Create Hubs	5 days	Mon 1/20/20	Fri 1/24/20		Anthony,Brandc
61		Create Wheels	10 days	Wed 2/12/20	Tue 2/25/20		Anthony,Brandc
62		Mount Wheels onto Base	2 days	Mon 2/24/20	Tue 2/25/20		Anthony,Brandc
63		Mount Sensors onto Prototype					All Members
64		Sensor Analysis	26 days	Tue 1/21/20	Tue 2/25/20		Marcin,Sean
65		Find Testing Location	26 days	Tue 1/21/20	Tue 2/25/20		Marcin,Omar
66		Gather Testing Data	26 days	Tue 1/21/20	Tue 2/25/20		All Members
67		Validate the Prototype Works	26 days	Tue 1/21/20	Tue 2/25/20		All Members
68		Assess feasibility of the project	26 days	Tue 1/21/20	Tue 2/25/20		All Members
69		Beta Prototype Design	42 days	Mon 2/3/20	Tue 3/31/20		All Members
70		Create Final CAD Mode					Omar
71		Order Beta Prototype Parts					All Members
72		Phase 5	9 days?	Wed 4/1/20	Mon 4/13/20		
73		Revise CAD Beta Prototyp	33 days	Thu 2/27/20	Mon 4/13/20		All Members
74		Write Sensor Code	33 days	Thu 2/27/20	Mon 4/13/20		Marcin,Sean
75		Create Electrical Scehmatic Diagrams	33 days	Thu 2/27/20	Mon 4/13/20		Marcin
76		Create theoretical plan for testing	33 days	Thu 2/27/20	Mon 4/13/20		All Members
77		Test Beta Prototype Components	31 days	Thu 3/26/20	Thu 5/7/20		All Members
78		Plan for Online Innovation Expo	8 days	Thu 4/2/20	Mon 4/13/20		
79		Submit Final CAD Files	1 day	Thu 4/9/20	Thu 4/9/20		Omar
80		Phase 6	15 days	Mon 4/13/20	Fri 5/1/20		
81		Final Revision Phase	15 days	Mon 4/13/20	Fri 5/1/20		All Members
82		Phase 6 Presentation	15 days	Mon 4/13/20	Fri 5/1/20		All Members
83		innovation Expo	1 day	Fri 5/1/20	Fri 5/1/20		All Members