

Washington University in St. Louis

## Washington University Open Scholarship

---

Mechanical Engineering Design Project Class

Mechanical Engineering & Materials Science

---

Spring 2021

### **MEMS 411: ARLISS Canister Vehicle**

Patrick Smith

*Washington University in St. Louis*

Brannon Boyd

*Washington University in St. Louis*

Matthew Donaldson

*Washington University in St. Louis*

Kieran Natarajan

*Washington University in St. Louis*

Follow this and additional works at: <https://openscholarship.wustl.edu/mems411>



Part of the [Mechanical Engineering Commons](#)

---

#### **Recommended Citation**

Smith, Patrick; Boyd, Brannon; Donaldson, Matthew; and Natarajan, Kieran, "MEMS 411: ARLISS Canister Vehicle" (2021). *Mechanical Engineering Design Project Class*. 144.

<https://openscholarship.wustl.edu/mems411/144>

This Final Report is brought to you for free and open access by the Mechanical Engineering & Materials Science at Washington University Open Scholarship. It has been accepted for inclusion in Mechanical Engineering Design Project Class by an authorized administrator of Washington University Open Scholarship. For more information, please contact [digital@wumail.wustl.edu](mailto:digital@wumail.wustl.edu).



# Washington University in St. Louis

## JAMES MCKELVEY SCHOOL OF ENGINEERING

### SP21 MEMS 411 Mechanical Engineering Design Project

#### ARLISS Canister Vehicle

The ARLISS canister vehicle is a part of the ARLISS competition, a collaboration between students to build, launch, test, and recover prototype satellites. This project mainly focuses on the vehicular aspects, designing a vehicle to meet the criterion to pass as an ARLISS vehicle. One of the largest design constraints from this competition is the ability to fit within a cylinder and not weigh too much such that it significantly changes the amount of thrust the rocket requires. The main outcome goals for this device were: being able to survive a drop and still function, operate and navigate autonomously to a given set of GPS coordinates, and travel through or across rough terrain. This reports serves to document the design process. Throughout this process, multiple design facets were optimized to allow the vehicle to effectively meet the outcome goals, including battery life and motor-control algorithm. The optimal design to maximize usable design space given the canister size constraint was a cylindrical, two-wheeled device. To ensure all the criteria were met, several engineering models were used to help determine how the vehicle will function, such as stall torque calculations and an FEM stress analysis. Finally, design considerations were made for safety, usability, and manufacturability. Through these methods, a successful prototype was constructed that was able to achieve all the outcome goals.

BOYD, Brannon  
DONALDSON, Matthew  
NATARAJAN, Kieran  
SMITH, Patrick

# Contents

<b>List of Figures</b>	<b>2</b>
<b>List of Tables</b>	<b>2</b>
<b>1 Introduction</b>	<b>3</b>
<b>2 Problem Understanding</b>	<b>3</b>
2.1 Existing Devices . . . . .	3
2.2 Patents . . . . .	5
2.3 Codes & Standards . . . . .	7
2.4 User Needs . . . . .	7
2.5 Design Metrics . . . . .	9
<b>3 Concept Generation</b>	<b>11</b>
3.1 Mockup Prototype . . . . .	11
3.2 Functional Decomposition . . . . .	12
3.3 Morphological Chart . . . . .	13
3.4 Alternative Design Concepts . . . . .	14
<b>4 Concept Selection</b>	<b>18</b>
4.1 Selection Criteria . . . . .	18
4.2 Concept Evaluation . . . . .	18
4.3 Evaluation Results . . . . .	19
4.4 Engineering Models/Relationships . . . . .	19
<b>5 Concept Embodiment</b>	<b>21</b>
5.1 Initial Embodiment . . . . .	21
5.2 Proofs-of-Concept . . . . .	21
<b>6 Design Refinement</b>	<b>25</b>
6.1 Model-Based Design Decisions . . . . .	25
6.2 Design for Safety . . . . .	29
6.3 Design for Manufacturing . . . . .	30
6.4 Design for Usability . . . . .	32
<b>7 Final Prototype</b>	<b>33</b>
7.1 Overview . . . . .	33
7.2 Documentation . . . . .	33
<b>Bibliography</b>	<b>35</b>
<b>Appendix A Software Code - MATLAB</b>	<b>36</b>

## List of Figures

1	iRobot Roomba s9+ [1]. . . . .	3
2	RC Tank showing maneuverability [2]. . . . .	4
3	Motorized MOC Hailfire droid (LEGO) [3]. . . . .	5
4	Patent images for expansible shaft [4]. . . . .	6
5	Patent images for Solar Car Toy [5]. . . . .	6
6	Four photographs of the initial mockup that was created using a PVC pipe, dowel rod, pencils, toilet paper roll, and cardboard. . . . .	11
7	Function tree for ARLISS Vehicle, screen shot from a Word document. . . . .	12
8	Morphological Chart for ARLISS Vehicle. . . . .	13
9	Preliminary and final sketches of the tank concept. . . . .	14
10	Preliminary and final sketches of the Tri-Wheel concept. . . . .	15
11	Preliminary and final sketches of the Swept-Wheel Cylinder concept. . . . .	16
12	Preliminary and final sketches of The Minimalist concept design. . . . .	17
13	Analytic Hierarchy Process (AHP) to determine scoring matrix weights . . . . .	18
14	Weighted Scoring Matrix (WSM) for choosing between alternative concepts . . . . .	19
15	Assembled projected views with overall dimensions. . . . .	22
16	Assembled isometric view with bill of materials (BOM). . . . .	23
17	Exploded view with callout to BOM. . . . .	24
18	Initial prototype design. . . . .	25
19	SolidWorks simulation displacement results of the bracket . . . . .	26
20	SolidWorks simulation stress results of the bracket . . . . .	27
21	SolidWorks simulation displacement results of the platform (25mm thickness) . . . . .	27
22	SolidWorks simulation stress results of the platform (25mm thickness) . . . . .	28
23	Risk assessment heat map. . . . .	30
24	Photographs of final prototype. . . . .	33
25	Final prototype: CAD exploded view. . . . .	34

## List of Tables

1	Interpreted Customer Needs . . . . .	8
2	Target Specifications . . . . .	10
3	List of main parts used in construction of the final prototype. . . . .	31

# 1 Introduction

Over the course of this report, we will be laying out our methods and design for a canister vehicle that could compete in the competition, A Rocket Launch for International Student Satellites (ARLISS). The objective of this competition is to design and build a vehicle which can be launched by a rocket to simulate a satellite in orbit. The overall design is constrained to fit within a cylinder and should be capable of auto-navigating to a given set of GPS coordinates upon returning to the planet's surface. Furthermore, the vehicle must be able to traverse an off-road terrain, traveling distances up to several miles. The entirety of this project will be designed from scratch, starting from a base schematic before being prototyped and refined until a final design is reached. The entire construction process will be overseen by Dr. James Potter and other Washington University faculty. Finally, once constructed, the device will be rigorously tested against a set of design specifications to determine its level of success in reaching the stated objectives.

## 2 Problem Understanding

### 2.1 Existing Devices

To inform our initial concept generation, we began by researching several existing devices whose stated purposes are similar to that of our own device.

#### 2.1.1 Existing Device #1: iRobot Roomba s9+



Figure 1: iRobot Roomba s9+ [1].

Link: <https://store.irobot.com/default/roomba-vacuuming-robot-vacuum-irobot-roomba-s9-s955020.html>

Description: The iRobot Roomba s9+ is a robotic vacuum cleaner. It is capable of autonomously cleaning a floor until its battery runs low or its dust tank is full. Once either of these conditions are met, it is capable of autonomously returning to its battery charging dock, which also empties the Roomba's dust tank. The Roomba moves via two, independently-driven, tractor-style wheels, allowing the unit to maintain its grip on dusty floors, pass over non-flush flooring, and turn on a dime. It is also fitted with optical sensors for obstacle avoidance, and can "learn" the layout of your home by generating a map of these obstacles.

### 2.1.2 Existing Device #2: RC Tank



Figure 2: RC Tank showing maneuverability [2].

Link: <https://www.aliexpress.com/item/32805317760.html>

Description: The Smart RC Tank is a DIY customizable device which allows for more maneuverability over a traditional RC car. The Smart RC Tank does so by incorporating 4 separate motors, one for each tread and a suspension system to keep the body balanced. This system allows the RC Tank to climb over obstacles that would stop most traditional vehicles.

### 2.1.3 Existing Device #3: Hailfire Droid



Figure 3: Motorized MOC Hailfire droid (LEGO) [3].

Link: <https://rebrickable.com/mocs/MOC-4355/Aniomylone/ucs-hailfire-droid/#details> Description: Shown in Fig. 3 is a My Own Creation (MOC) of a Lego Hailfire droid. This design is constructed of 1,000 pieces with working drive trains. There are two cylindrical components on the side that rotate with treads able to navigate even the roughest of terrains. The Hailfire droid can turn with independent motion of the trains and with the main chassis being raised up due to the nature of the wheels. This design allows for the vehicle to deal with an off-road environment, while maximizing the efficiency of the cylindrical constraints. While the design is not motorized, a motor can be added allowing for autonomous motion.

## 2.2 Patents

Listed below are a few U.S. patents that could restrict the commercial use of our design.

### 2.2.1 Miniature Robotic Vehicles And Methods Of Controlling Same (6548982 B1)

This patent looks at miniature robotic vehicles and how to overcome diverse terrain. The vehicle used in the patent is a compact robotic vehicle with two wheels, an outer body, sensors and a spring. The vehicle in the patent utilizes a spring that would launch a flap off of the vehicle in order for the vehicle to maneuver obstacles on the ground (on the ground "propel across a surface"). Another prototype implements wheels that can rotate to maneuver around the terrain and get across the ground. Overall, the robotic vehicles in this patent provide ways to overcome the problem of maneuvering a small vehicle across terrain [4].

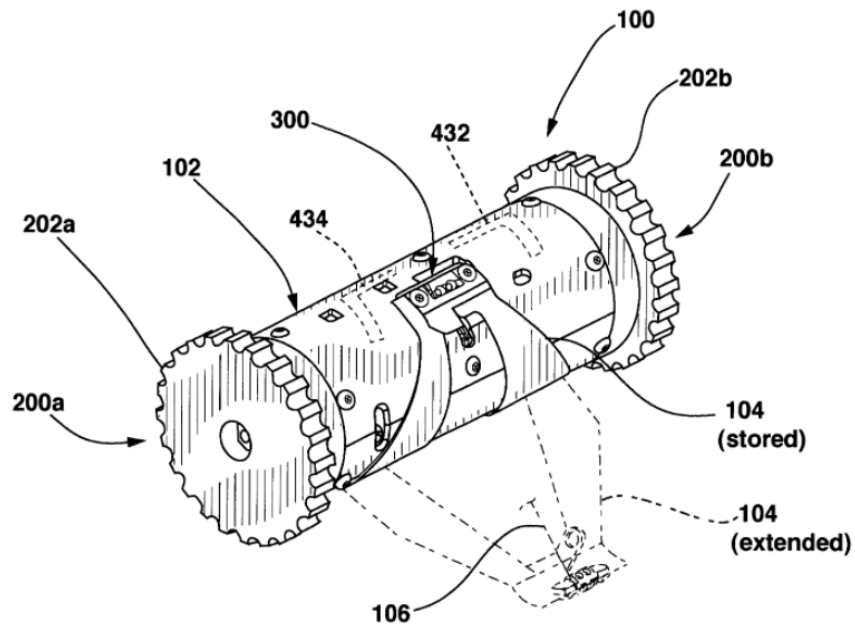


Figure 4: Patent images for expansible shaft [4].

### 2.2.2 Solar Car Toy KR100683271B1

This patents implements a solar cell unit into a toy car. The frame of the car is fiberglass, keeping the body light weight, with four wheels (two on each side). The solar cell sits in the center top of the body of the vehicle where it converts the solar heat into electricity for the the motor to be operated. It should also be noted that the solar cell is curved to cause lift as the vehicle moves forward [5].

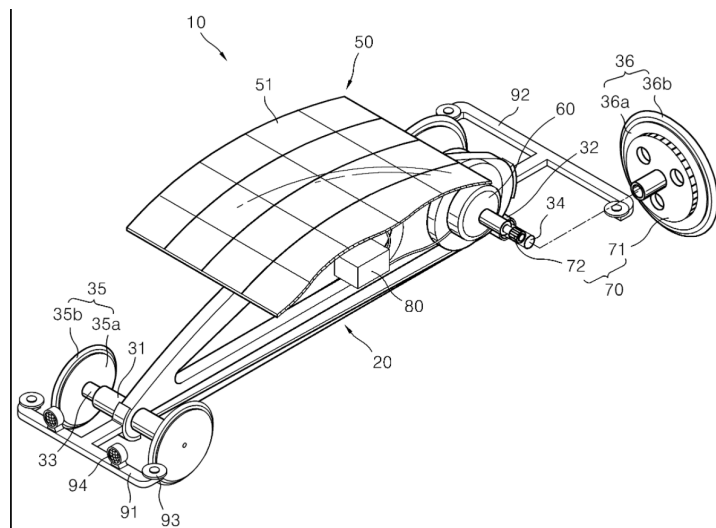


Figure 5: Patent images for Solar Car Toy [5].



## 2.3 Codes & Standards

Listed below are a few engineering standards that could guide our design and design specifications. We do not currently have full access to these standards, so we have simply listed their topics and potential uses.

### 2.3.1 Geometrical product specifications (GPS) – ISO code system for tolerances on linear sizes – Part 1: Basis of tolerances, deviations and fits (ISO 286-1:2010)

This standard provides size and tolerance constraints for various shaft/hole fit designations. We will use this standard to determine the fit type and dimensions of a possible wheel/axle interface in the drive system of our design.

### 2.3.2 Lithium-Ion Battery-Safety (IEC 62133-2:2017)

This standard specifies requirements and tests for the safe operation of portable sealed secondary lithium cells and batteries containing non-acid electrolyte, under intended use and reasonably foreseeable misuse. We will use this standard to determine if the battery we choose for our device will stay intact during operations.

## 2.4 User Needs

In this section we discuss with the customer what they want in the device.

### 2.4.1 Customer Interview

Interviewee: Dr. James Jackson Potter

Location: Zoom Meeting Room, Washington University in St. Louis, Danforth Campus

Date: February 2<sup>nd</sup>, 2021

Setting: We were showed a video of an ARLISS launch by the customer before he described what the competition entailed and what he was looking for in a canister vehicle. The whole interview was conducted in his office, and took ~40 min.

Interview Notes:

*What are the size constraints of this device?*

- The vehicle must fit in in a cylinder with a 146 mm diameter and length of 240 mm.

*What are the weight constraints of this device?*

- The vehicle must weigh less than 1050 g.

*Does the vehicle need to record GPS data while driving?*

- Yes, it should be able to somehow record the data and save it to some kind of a memory card.

*What are the objectives of this vehicle?*

- It should be able to navigate to a given set of GPS coordinates. You don't need to worry about launching or parachutes.

*Do you care if the vehicle is aesthetically-pleasing?*

- I don't care at all. It just needs to work.

*Does the vehicle need to be water-resistant?*

- No, the vehicle will only be operated on dry days in the desert.

*Does the vehicle need to make tight turns?*

- No, it will only be operated in the desert where there will be plenty of space to turn.

*How far will the vehicle need to travel?*

- It should be able to travel distances of around 3 miles on a single charge.

## 2.4.2 Interpreted User Needs

We assign values to the various needs we discerned from our interview with the customer. The scale goes from 1 (not important) to 5 (extremely important).

**Table 1: Interpreted Customer Needs**

Need Number	Need	Importance
1	Vehicle is compact	5
2	Vehicle is lightweight	2
3	Vehicle navigates to given GPS coordinates	5
4	Vehicle is autonomous	5
5	Vehicle has a long battery life	4
6	Vehicle moves quickly	1
7	Vehicle will function normally in a sandstorm	3
8	Vehicle is self-righting	4
9	Vehicle operates normally after low-impact drop	4
10	Vehicle has renewable power source other than battery	3
11	Vehicle is user-safe	2
12	Vehicle can record and store GPS data	4
13	Vehicle can navigate in desert terrain	5
14	Vehicle is capable of making tight turns	2
15	Vehicle is aesthetic	1

Based on Table 1 above, the most important needs in this design are the size and weight constraints as well as having the car be fully autonomous and the vehicle having a long battery life. Some needs that are less important are speed, turn radius and aesthetic appeal.

## **2.5 Design Metrics**

In the design metrics we take our customer needs table and apply values to it. These values are in two groups: acceptable (the lowest/highest we will accept a metric) and ideal (where we would like our value to be).

**Table 2: Target Specifications**

Metric Number	Associated Needs	Metric	Units	Acceptable	Ideal
1	1,2	Total weight	g	< 1050	< 80
2	1	Total diameter	mm	< 146	< 110
3	1	Total height	mm	< 240	< 200
4	5	Battery Life	km	> 3	> 5
5	6	Speed of vehicle	mph	> 0.5	> 2.5
6	9	Number of drops from hip height	amount	3	10
7	3,4	Vehicle navigates to given GPS coordinates	hours	it gets there	3 – 4
8	7,13	Vehicle will operate in the desert conditions	Boolean	True	True
9	14	Turn radius	m	< 1	< 0.5
10	11	Overall Safety	touch test	no bleed	no injury
11	12	Capacity to store data	GB	> 8	> 1000
12	15	Vehicle looks cool	n/a	It functions	Store-ready
13	8	Vehicle is self righting	s	< 10	< 3
14	10	Vehicle used an alternative, renewable energy source	type 10	Charged battery	Fully renewable

### 3 Concept Generation

#### 3.1 Mockup Prototype

Shown in Fig. 6 is a rough initial design that we created in order to flush out some initial concepts and determine some of the troublesome points within our design. We found that for this mock-up the most difficult part was getting the wheels to stay on and function properly.

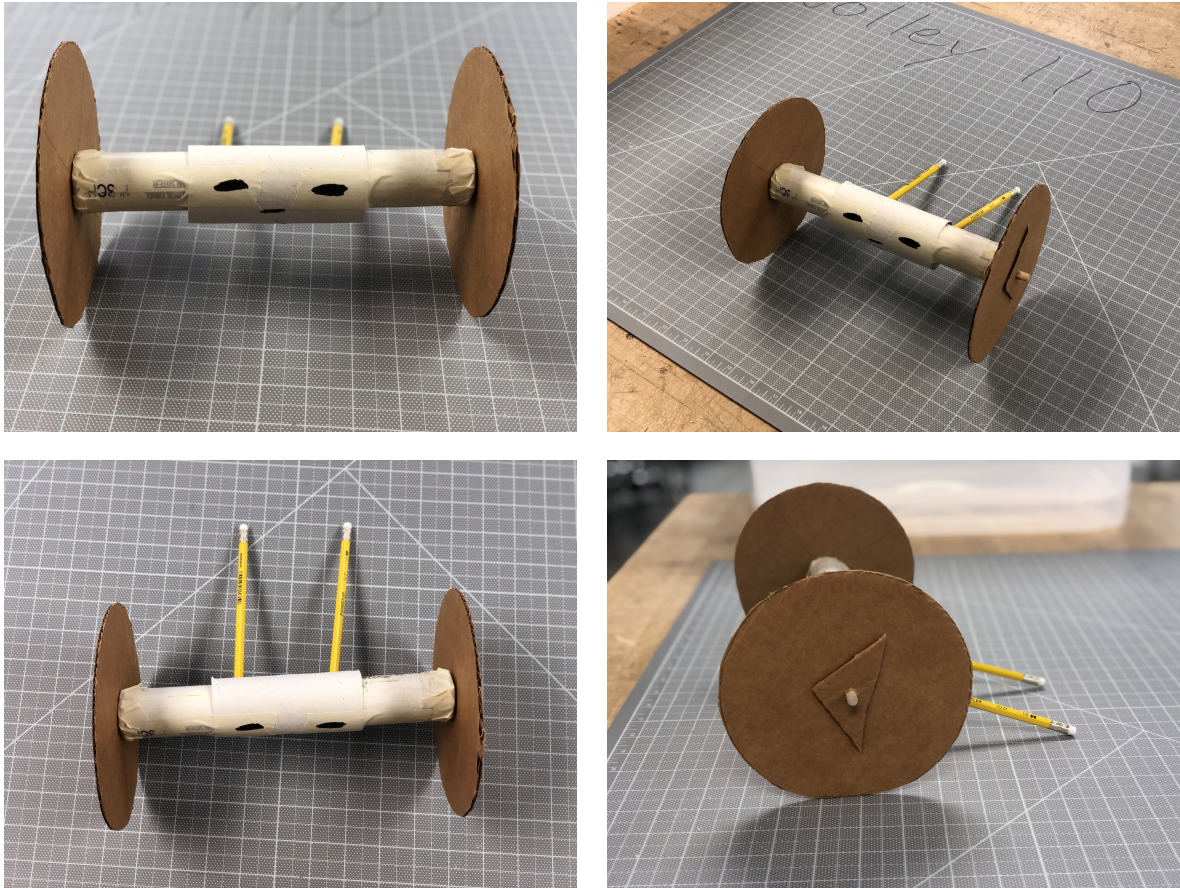


Figure 6: Four photographs of the initial mockup that was created using a PVC pipe, dowel rod, pencils, toilet paper roll, and cardboard.

## 3.2 Functional Decomposition

Our function tree (Fig. 7) breaks down our goal of driving to a goal into the necessary components to get there within the constraints of the ARLISS competition.

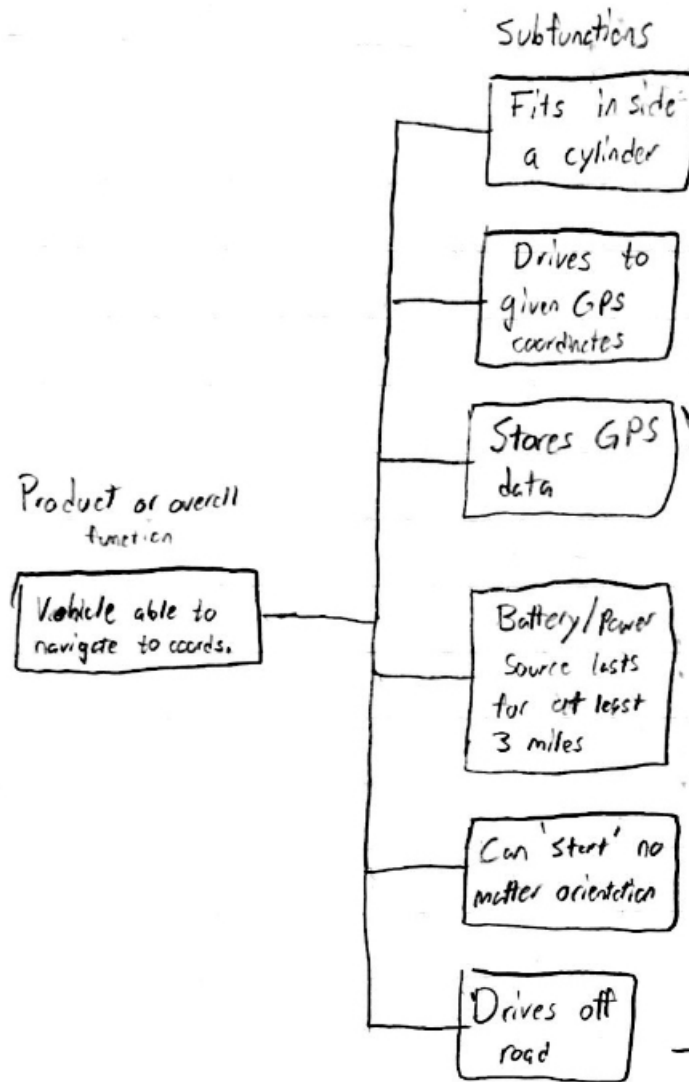


Figure 7: Function tree for ARLISS Vehicle, screen shot from a Word document.

### 3.3 Morphological Chart

Our morph chart (Fig. 8) shows some possible solutions to our function tree.






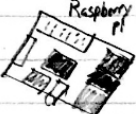



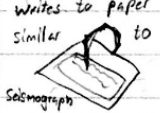

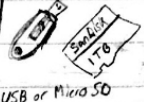
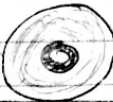

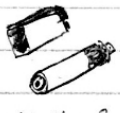
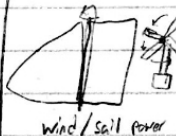

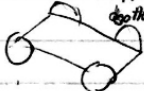




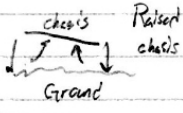
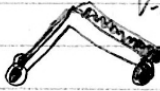

Fits inside a cylinder	Long, short, narrow 	Vehicle is cylindrical 	Fold over wheels 	Hovers, but small 	Spherical design 
Drives to given GPS coordinates	Raspberry pi 	Arduino 	R/C 	Infinite driving until it eventually reaches destination 	
Stores GPS data	writes to paper similar to Seismograph 	Uploads instantly to a cloud 	USB or Micro SD 	Burns to CD 	
Battery/Power source lasts for at least 3 miles	Solar 	Alkaline Powered 	Wind/sail power 	Steam/coal/gas combustion powered 	
Can 'start' no matter orientation	No distinct top/bottom 	Cylindrical with no sides 	Round 	Spring mechanism to self-right 	
Drives off road	Tractor-like treads 	chess's Raised chasis Ground 	V-Shape 	chain of wheels or chains that spin on axles 	

Figure 8: Morphological Chart for ARLISS Vehicle.

## 3.4 Alternative Design Concepts

### 3.4.1 Tank

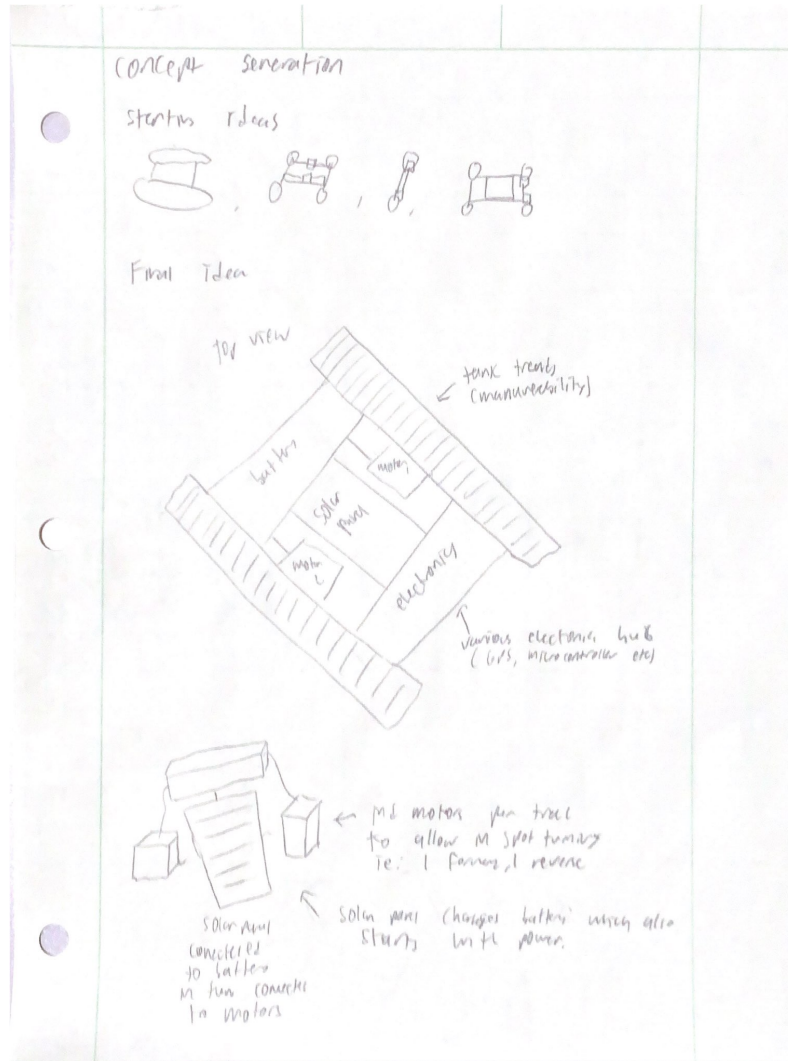


Figure 9: Preliminary and final sketches of the tank concept.

Description: Tank treads allow for more maneuverability over uneven terrain with a motor for each so turning can be done in place (one forward one backward). A solar panel and backup battery provide enough power to operate well within the desired range. A shell covering the top components to protect during a fall test with additional protection for the data storage within.



### 3.4.2 Tri-Wheel Vehicle

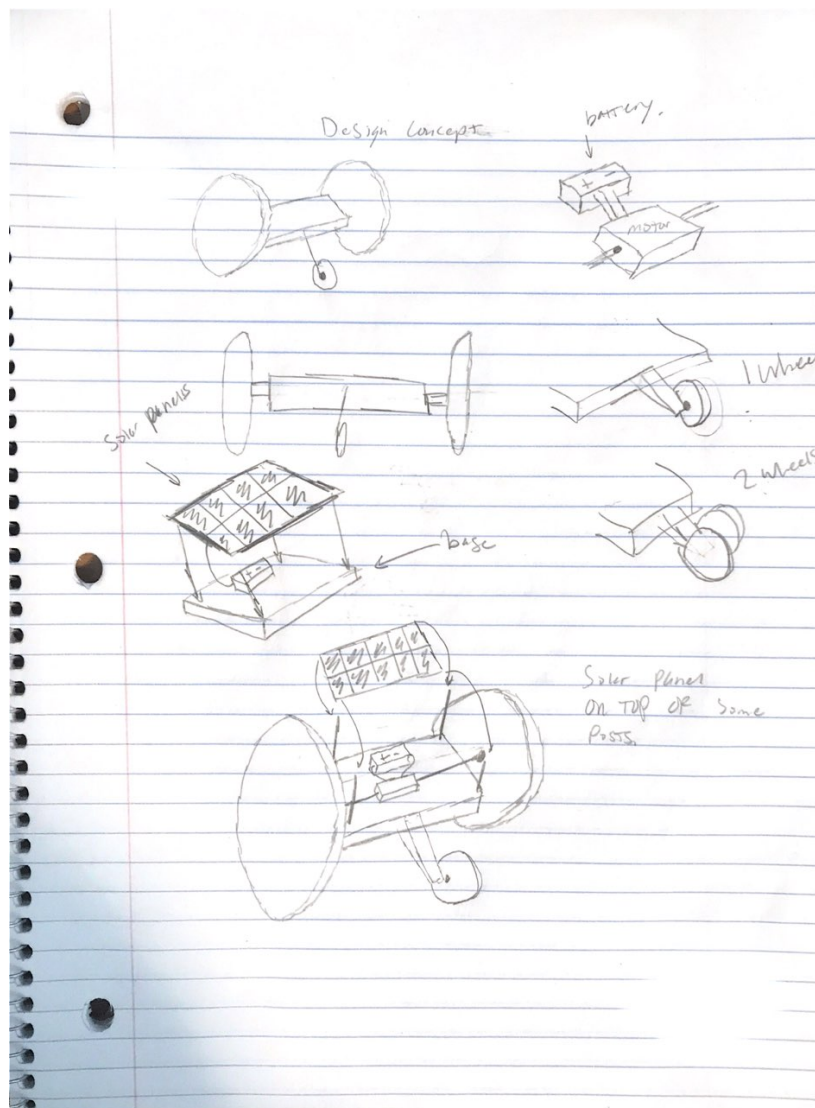


Figure 10: Preliminary and final sketches of the Tri-Wheel concept.

Description: The mini wheel allows for the vehicle to move and not have the body turn in place also keeps the tail from dragging on the ground. The big wheels are used to keep the body of the ground and get over the small obstacles that may be encountered in the desert terrain. Also, a small bearing on a wheel to allow the vehicle to rotate. The springs hopefully would cushion the impact of a fall for the vehicle. Lastly, solar panels would be mounted on top to provide extra battery life to the vehicle.

### 3.4.3 Swept-Wheel Cylinder

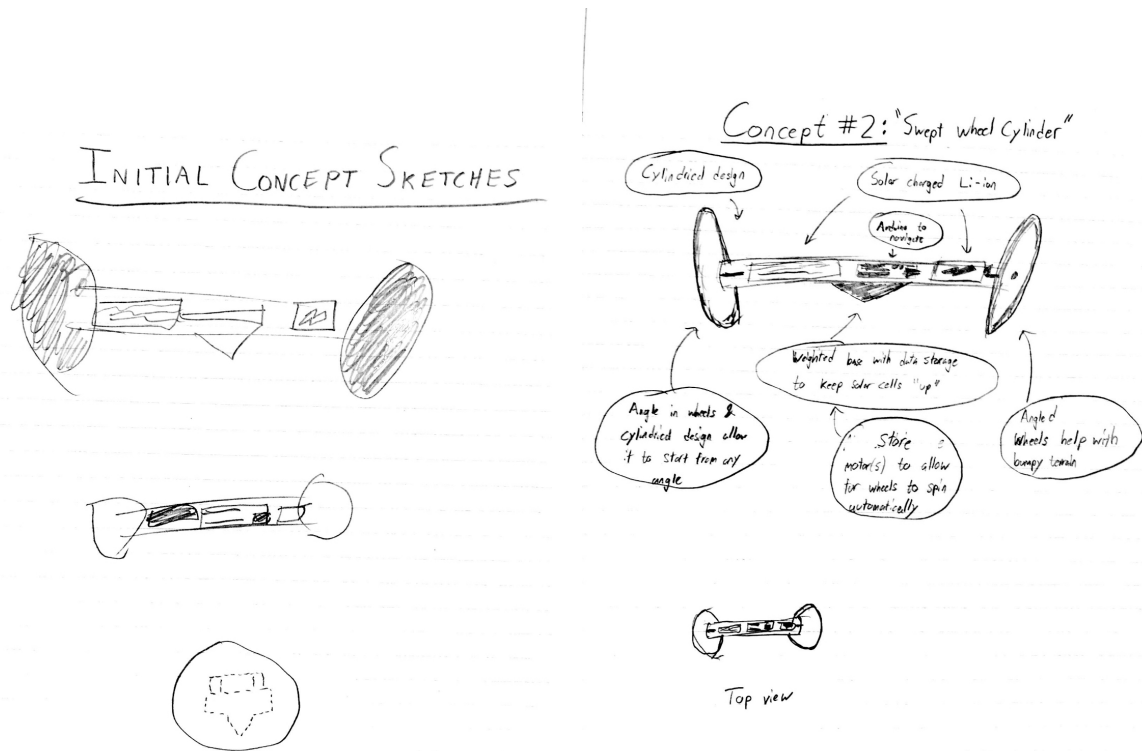


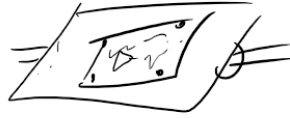
Figure 11: Preliminary and final sketches of the Swept-Wheel Cylinder concept.

Description: The Swept-Wheel Cylinder design features two independently powered wheels allowing for easy navigation. The wheels are attached to a main body that contains an Arduino UNO unit as well as solar panel, Li-ion battery, and data storage MicroSD card. These features will allow the vehicle to successfully trek to a given set of GPS coordinates no matter the distance. It will also enable storage of the route through the MicroSD card for visualization of the vehicle's trajectory. Further, the weighted undercarriage along with angled wheels and cylindrical design allow for the vehicle to be started from almost any initial position even after being dropped. Most importantly, the design is cylindrical in nature and will thus satisfy the condition that it should be able to fit within a cylinder to potentially be stored on a rocket.

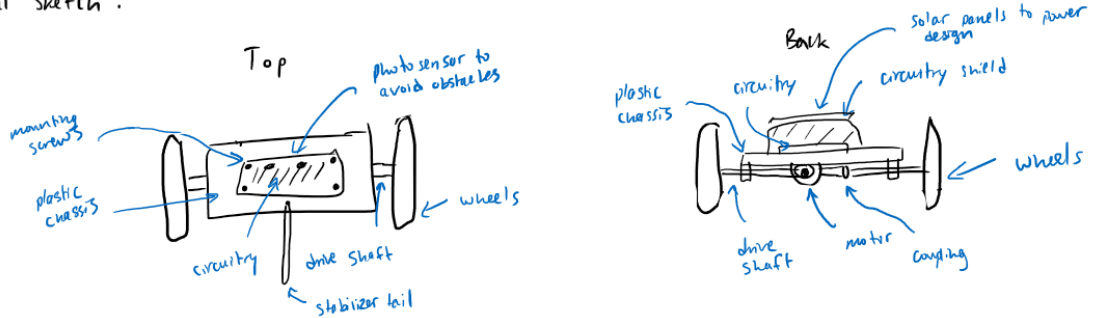
### 3.4.4 The Minimalist

#### Concept 1: "The Minimalist"

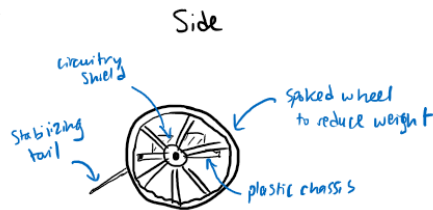
Preliminary ideas:



Final sketch:



Designed to minimize weight and avoid obstacles. Fast and maneuverable.



One wheel fitted with reverse bushing so when the bot reverses, that side won't engage, causing bot to turn.

Figure 12: Preliminary and final sketches of The Minimalist concept design.

Description: The Minimalist concept is all about lightweight maneuverability and obstacle avoidance. Rather than include shock absorbers or a heavy-duty chassis to protect the bot, this design features a photosensor to detect and subsequently veer around obstructions. The wheels are spoked rather than solid to further reduce weight, and the chassis consists solely of a plastic rectangle on which the components are mounted. The bot is completely powered by a solar panel to remove the need for a weighty onboard battery. Finally, since the wheels are the largest part of the side view, the stabilizing tail can collapse to ensure the bot has no problem fitting inside the launch cylinder.

## 4 Concept Selection

### 4.1 Selection Criteria

Shown in Fig. 13 is an Analytic Hierarchy Process (AHP) analyzing key components for our design. An Analytic Hierarchy Process is essentially a structural technique for making a complex decision based on mathematics and psychology.

	Battery Life	Manuverability	Ease of Engineering	Cost	Durability		Row Total	Weight Value	Weight (%)
Battery Life	1.00	7.00	7.00	3.00	1.00		19.00	0.34	33.95
Manuverability	0.14	1.00	0.20	0.33	0.14		1.82	0.03	3.25
Ease of Engineering	0.14	5.00	1.00	3.00	0.20		9.34	0.17	16.69
Cost	0.33	3.00	0.33	1.00	0.14		4.81	0.09	8.59
Durability	1.00	7.00	5.00	7.00	1.00		21.00	0.38	37.52
						<b>Column Total:</b>	<b>55.97</b>	<b>1.00</b>	<b>100.00</b>

Figure 13: Analytic Hierarchy Process (AHP) to determine scoring matrix weights

This type of process allowed us to develop numerical weights for five of our most important tenants for the design. This was done through a comparative analysis between each tenant and then the final results were made into percentages to be used in a concept selection through a Weighted Scoring Matrix (WSM).

### 4.2 Concept Evaluation

Seen in Fig. 14 is our completed Weighted Scoring Matrix (WSM). This matrix takes the criteria and weighted averages of the Analytic Hierarchy Process, seen in Fig. 13, and applied them to our designs. Each design is rated on a scale of 1-5 for the specific criteria. Once rated the weight % and rating value are used to generate a weighted score for the criteria. All weighted criteria scores are added together to produce the design that best suits our selection criteria.




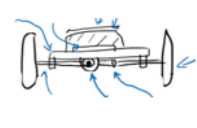
Alternative Design Concepts		Brannon Concept		Kieran Concept		Matt Concept		Patrick Concept	
									
Selection Criterion	Weight (%)	Rating	Weighted	Rating	Weighted	Rating	Weighted	Rating	Weighted
Battery Life	33.95	5	1.70	5	1.70	5	1.70	5	1.70
Manuverability	3.25	5	0.16	3	0.10	4	0.13	5	0.16
Ease of Engineering	16.69	2	0.33	3	0.50	4	0.67	4	0.67
Cost	8.59	3	0.26	3	0.26	3	0.26	4	0.34
Durability	37.52	4	1.50	4	1.50	3	1.13	2	0.75
	<b>Total score</b>	<b>3.952</b>		<b>4.054</b>		<b>3.878</b>		<b>3.622</b>	
	<b>Rank</b>	<b>2</b>		<b>1</b>		<b>3</b>		<b>4</b>	

Figure 14: Weighted Scoring Matrix (WSM) for choosing between alternative concepts

### 4.3 Evaluation Results

From the weighted score matrix, the Tank design scored the highest for our 5 criteria. All of the concepts used batteries with solar panels, making all the concepts score high for that criteria. Since the Swept Wheel Cylinder had tilted wheels, the ease of engineering scored low in that regard, while the Minimalist scored a 4 for the ease of engineering because of its simplistic layout. The Tank and Tri-Wheel concepts also had similar scores as the Minimalist for ease of engineering. The Tank and Tri-Wheel designs scored the lowest for maneuverability because one used tank treads while the other had three wheels, putting more restriction on turning. The Swept Wheel Cylinder and Minimalist concepts score high for maneuverability because they only used two wheels, allowing the vehicle to make sharper turns. For cost and durability, the scoring in all the concepts was about the same, except the Minimalist design was given a 2 for durability because it may break if dropped. Overall, the Tank concept scored the highest. However, ideas from all the sketches will be used in the final design.

### 4.4 Engineering Models/Relationships

#### STALL TORQUE MODEL

Force-momentum model to determine minimum stall torque. Begin with relation for torque:

$$T = F_m \times r \tag{1}$$

where  $T$  is the stall torque [N·m],  $r$  is the radius [m], and  $F_m$  is the force required to move [N]. From this equation, we know that we need the force required to move to be greater than the maximum gross weight of the vehicle. We then find the following inequality:

$$\begin{aligned} W &< F_m \\ \implies mg &< \frac{T}{r} \end{aligned}$$

where the max gross weight,  $W$  [N], can be expressed as the mass,  $m$  [kg], times the relative acceleration due to gravity,  $g$  [ $m/s^2$ ]. Rewriting this expression explicitly for the stall torque we find:

$$T > mgr \quad (2)$$

Then using conservative estimates for these two variables ( $m$  and  $r$ ) of the maximum possible mass of 1050 g, and 0.0735 m wheel radius, we can find a minimum stall torque required of:

$$T > (1.05kg)(9,81m/s^2)(0.0735m) \quad (3)$$

$$T > 0.757N \cdot m = \boxed{7.72kg \cdot cm} \quad (4)$$

We will therefore need to pick a motor, or set of motors, that will combined produce a larger stall torque than the result found above to ensure that our vehicle can move.

### PEUKERT'S LAW ESTIMATES THE AMOUNT OF BATTERY LIFE

Amount of time taken for a battery to fully discharge:

$$t = H \left( \frac{C}{IH} \right)^k \quad (5)$$

where  $t$  is the amount of time of battery life [hrs.],  $H$  is the rated discharge time [hrs],  $C$  is the rated capacity on the battery [A-hrs.],  $I$  is the actual discharge current [A] and  $k$  is the Peukerts constant. This equation will give insight to what battery to choose. We know that the vehicle needs to travel approximately 3-5 [km]. If assuming that distance takes 6 hours to complete, the batter has a Peukerts constant of 1 in ideal scenarios and a actual discharge current of maybe 10 [A] then the rated capacity can be found. This then will help when choosing a battery to use.

### MINIMUM DRIVE SHAFT DIAMETER MODEL

Maximum shear stress in a rod due to torsion:

$$\tau = \frac{Tr}{J} = \frac{2T}{\pi r^3} \quad (6)$$

where  $T_{max}$  [N-m] is the supplied motor torque,  $r = d/2$  [m] is the shaft radius, and  $J = \frac{\pi}{2}r^4$  [ $m^4$ ] is the polar moment of inertia of the shaft cross section. Solving this relation for shaft diameter yields:

$$d \geq 2 \left( \frac{2T}{\pi\tau_{allow}} \right)^{1/3} \quad (7)$$

Choosing 6061 aluminum as the drive shaft material due to its good strength-weight ratio and relatively low cost,  $\tau_{max} = 30$  ksi. Using a safety factor of 1.2,  $\tau_{allow} = 25$  ksi. Using the specifications for a 12 Volt, 100:1 gear ratio DC motor found online,

$T_{max} = 4.375$  lb-in. Using these numbers,  $d \geq 0.0962$  in., which we round up for safety and part availability to:

$$\boxed{d = 1/8''} \tag{8}$$

Thus, we should choose a 1/8 inch-diameter, 6061 aluminum steel shaft to prevent the shaft from failing under torsion for the motor selected above.

## 5 Concept Embodiment

### 5.1 Initial Embodiment

Shown in Figs. 15-17 (next page) are SolidWorks models of the initial prototype design of the ARLISS canister vehicle. Parts found on GrabCad, etc. were used when possible. Otherwise, rudimentary designs as representations of the components were used. For instance, the wheels (Dagu Wild Thumper Wheels) required a basic design as a simple representation in the model. Additionally, a bill of materials and exploded view can be seen in Figs. 16 and 17, respectively.

### 5.2 Proofs-of-Concept

For our initial prototype, the main focus was initializing our concepts to create a cylindrical vehicle capable of moving autonomously. We were more relaxed on the size constraints, durability, and power source, allowing us to test a more basic design. In the second design iteration, the the base will be 3D printed and parts will be assembled properly. Figure 18 (next page) shows our first prototype. The motors used (100:1 Metal Gearmotor 20D x 44L mm 12V CB with Extended Motor Shaft) possessed stall torques (6.1 kg-cm each) greater than the required value from in Eq. 3. The Arduino, motors, drive controller, and battery were wired using circuitry knowledge.

The Arduino code used a closest-path-style algorithm to navigate to the given GPS coordinates. The vehicle begins its journey following a circular path, constantly evaluating the magnitude of the vector between it and the given GPS coordinates using the distance formula:

$$d = \sqrt{(x_{goal} - x_{curr})^2 + (y_{goal} - y_{curr})^2} \tag{9}$$

where the subscript “goal” refers to the desired coordinates, “curr” refers to the current coordinates, latitude is assumed to be  $x$ , and longitude is the  $y$ . As soon as this distance begins to increase (following an initial decrease to ensure the vehicle is pointed generally toward the target), the vehicle stops, turns 90° to point directly toward the target (normal to the circle), and begins traveling in that direction. This process is then repeated every so often to ensure the proper direction is maintained throughout the trip.

Our initial prototype did not function quite as we had hoped. However it allowed us to determine the main problems needing to be addressed. Therefore, the design

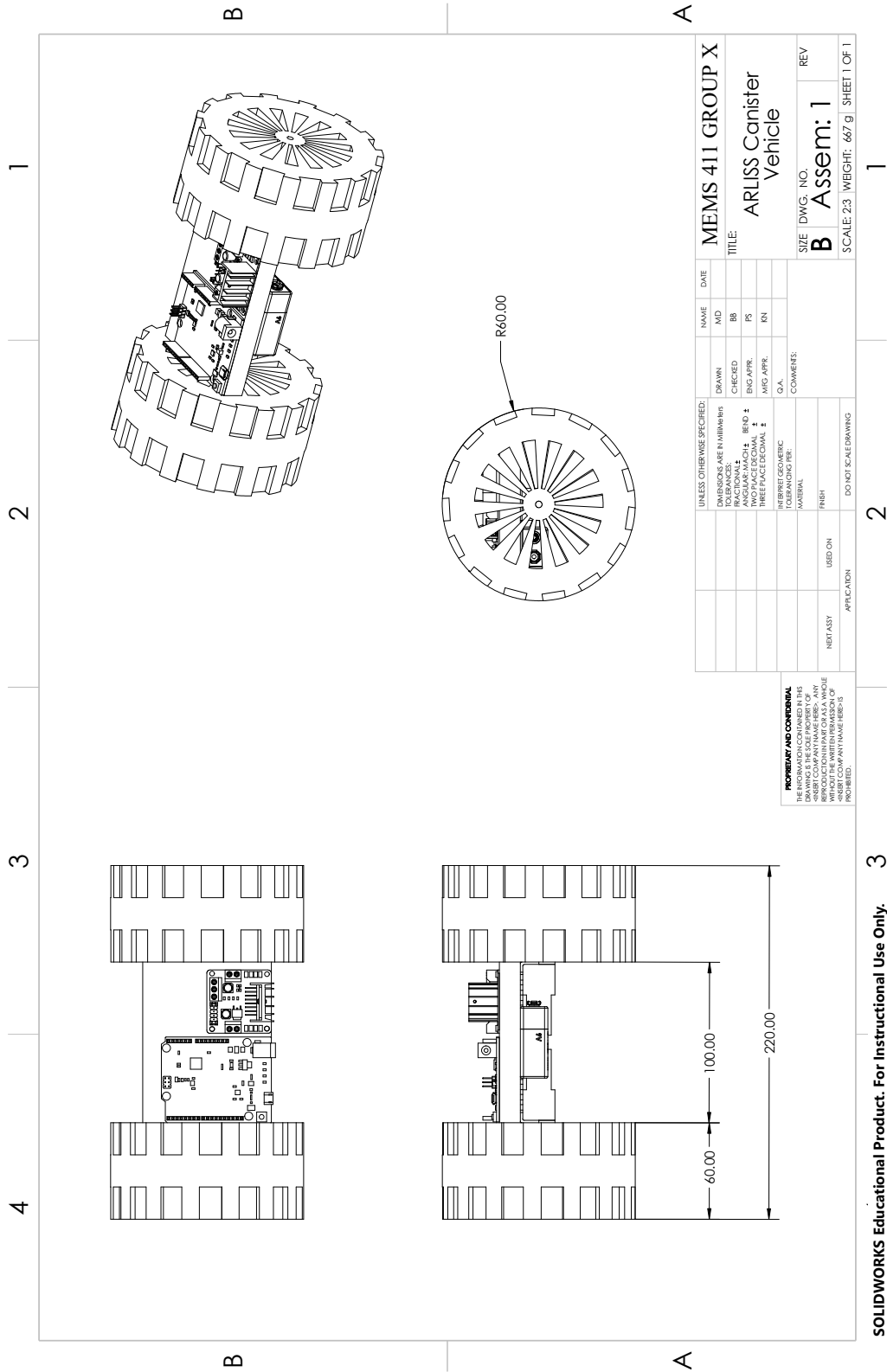


Figure 15: Assembled projected views with overall dimensions.





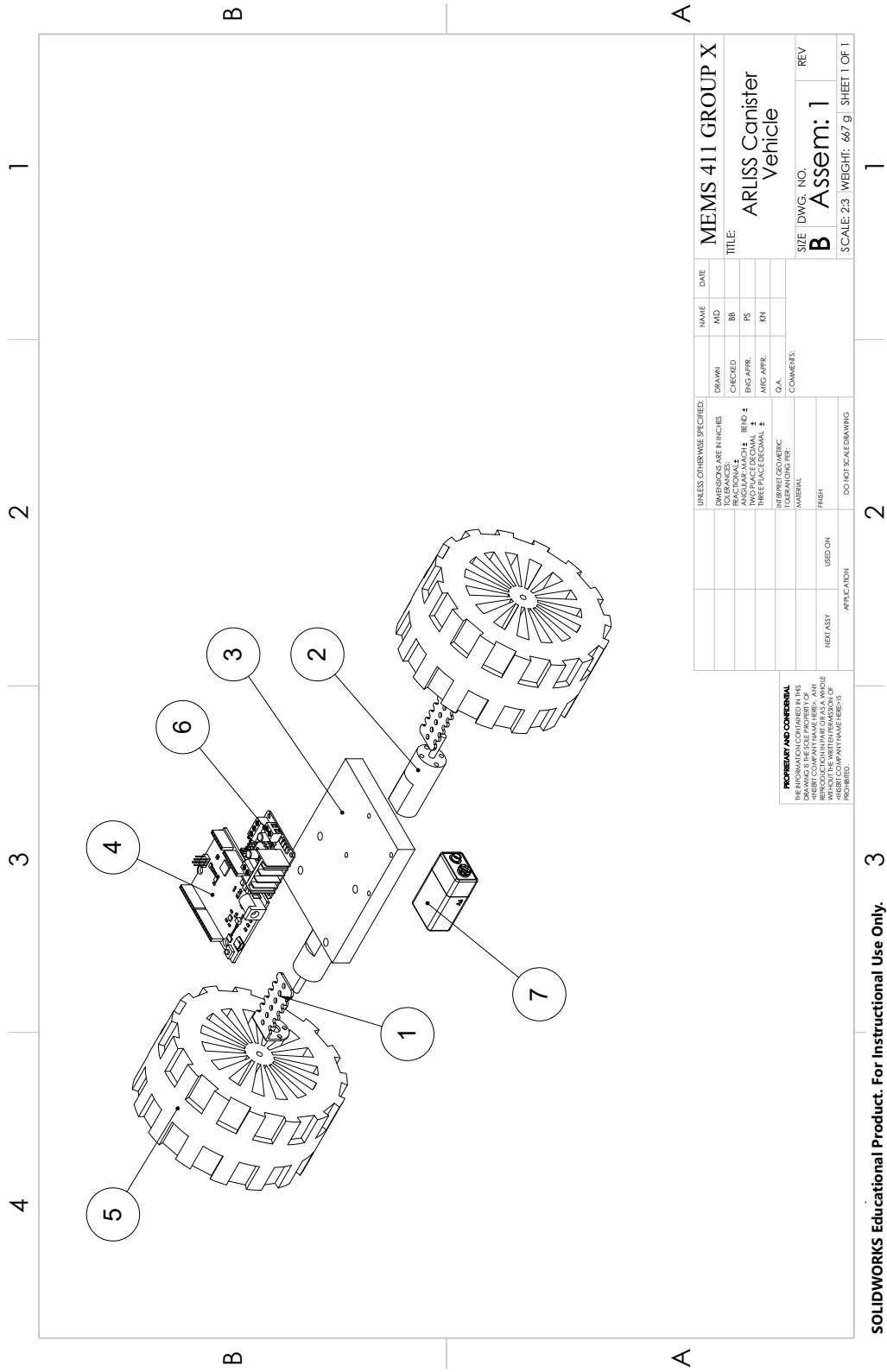


Figure 17: Exploded view with callout to BOM.

served its purpose. One of these problems we identified from this prototype was that the 9V battery used was not capable of supplying enough power to the Arduino while operating the motors at full speed. The wiring was done in a temporary fashion since this prototype is not the final iteration, so the connections were likely not completely secure. Both factors may have contributed to one of the motors being unable to operate in both directions through Arduino control. Additionally, the vehicle tended to flip over when braking due to the rotational momentum of the body. Finally, the vehicle did not travel in perfectly straight lines.

Moving forward, we will implement a permanent, anti-rotation tail that can fold to fit in the launch cylinder. To solve this, we have considered a snap-lock system similar to a spring-activated knife, or something similar to the snap bracelets that lock around your wrist. To fix the issue of the body flipping during braking, we will implement a more gradual braking subroutine. We have also ordered 12V batteries and better jumper wires to ensure adequate power for the electronics. Next, we have ordered narrower wheels so our second design iteration will meet the width constraint of 240 mm. Finally, we will adjust motor speeds in the code so the vehicle can travel in straight lines.

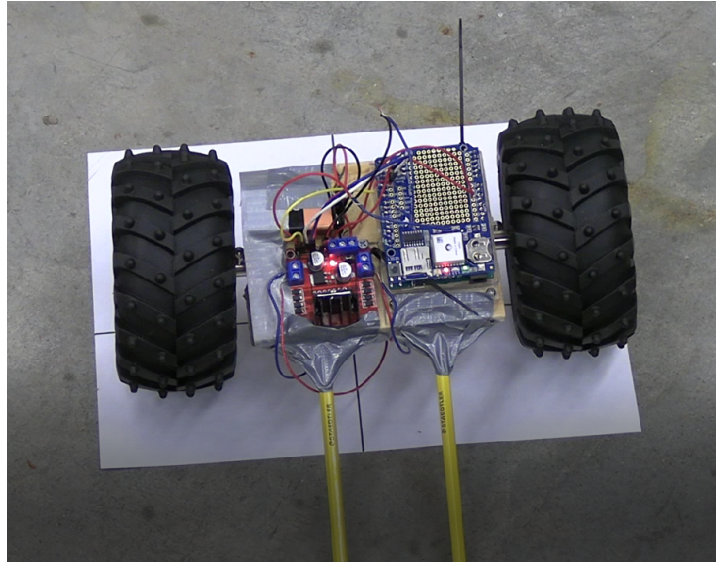


Figure 18: Initial prototype design.

## 6 Design Refinement

### 6.1 Model-Based Design Decisions

#### 6.1.1 Motor Bracket Impact Strength Verification

A model of the stress and displacement of the motor brackets was used to determine if the aluminum brackets were sufficient in holding the motors and wheels during the drop test. In order to get an estimation of how much force the brackets would experience,

conservation of energy and impact force equations were used, as shown in Eq. 10 below:

$$U = mgh \quad (10)$$

where  $U$  is the potential energy of an object [J],  $m$  is the mass of the object [kg],  $g$  is the gravitational constant [ $\text{m/s}^2$ ] and  $h$  is the height at which the object will fall [m]. Assuming conservation of energy, we can say that all the energy from the fall would be converted into an average force felt by the brackets, as shown in in Eq. 11:

$$U = \bar{F}\Delta D \quad (11)$$

where  $\bar{F}$  is the average force [N] and  $\Delta D$  is the impact distance of the whole object after the collision [m]. With Eq. 10 and Eq.11, the average force can be estimated.

If it is assumed that the mass of the vehicle is 1 kg (max weight allowed), the height if the drop is 0.8 m (roughly 3 ft) and a total impact distance of 15 mm, the average force can then be calculated to be 520 N. So between two brackets, one for each motor and wheel the average force experience by the bracket would be 260 N. When creating the brackets in SolidWorks, the estimated average impact force can be used to estimate a displacement and stress of the bracket. The displacement was found to be 0.8 mm and a stress of 430 MPa. With the displacement being less than a millimeter but the stress being slightly above the yield stress of our aluminum bracket, we know some permanent deformation would occur. However, this is a one-time impact and the bracket can also be bent back if needed. For multiple drops, it is recommended to add an extra fastener to help support the bracket. See Figs. 19 and 20 for SolidWorks results plots.

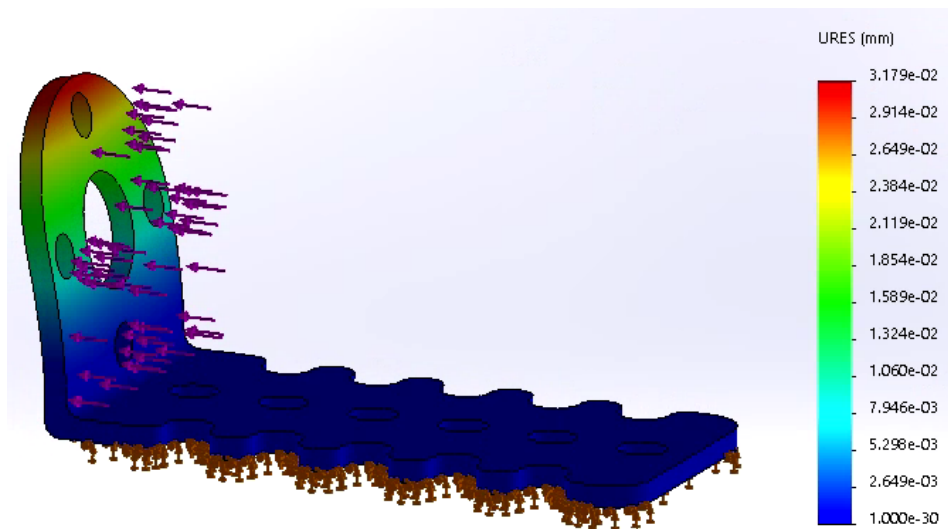


Figure 19: SolidWorks simulation displacement results of the bracket

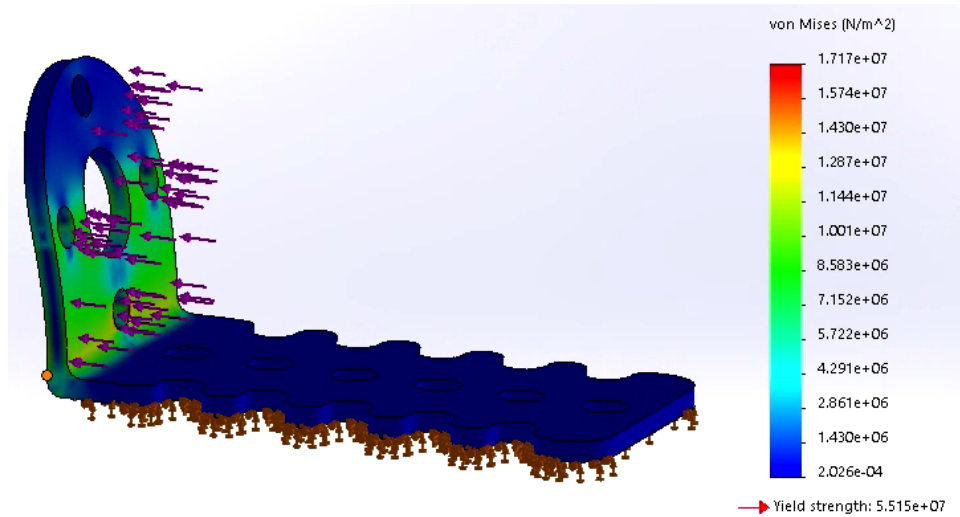


Figure 20: SolidWorks simulation stress results of the bracket

### 6.1.2 Base Thickness Selection

A 2D simulation of the deformation of the platform was also done in SolidWorks to approximate the thickness the platform. The thickness of the platform was set to 25 mm. The displacement and stress results are seen in Figs. 21 and 22 below.

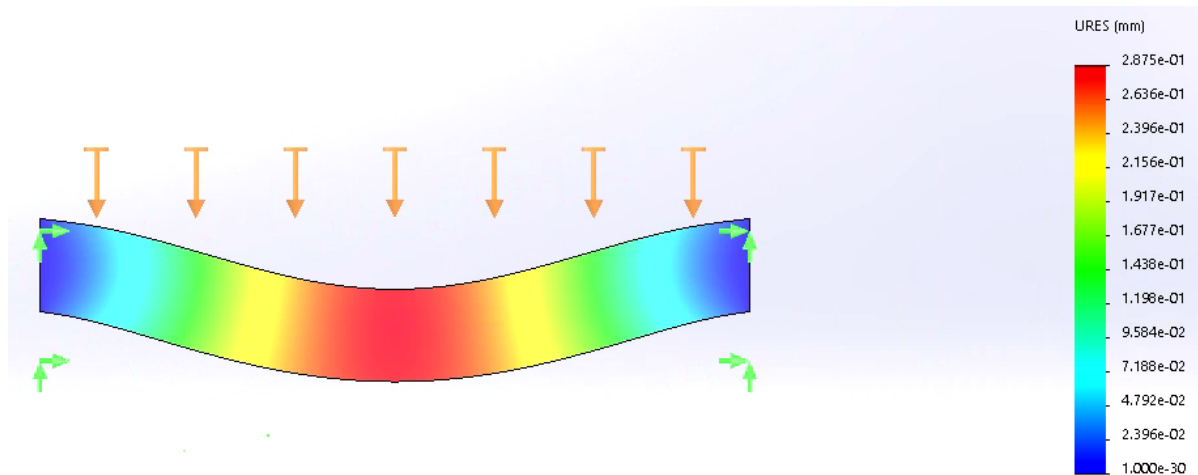
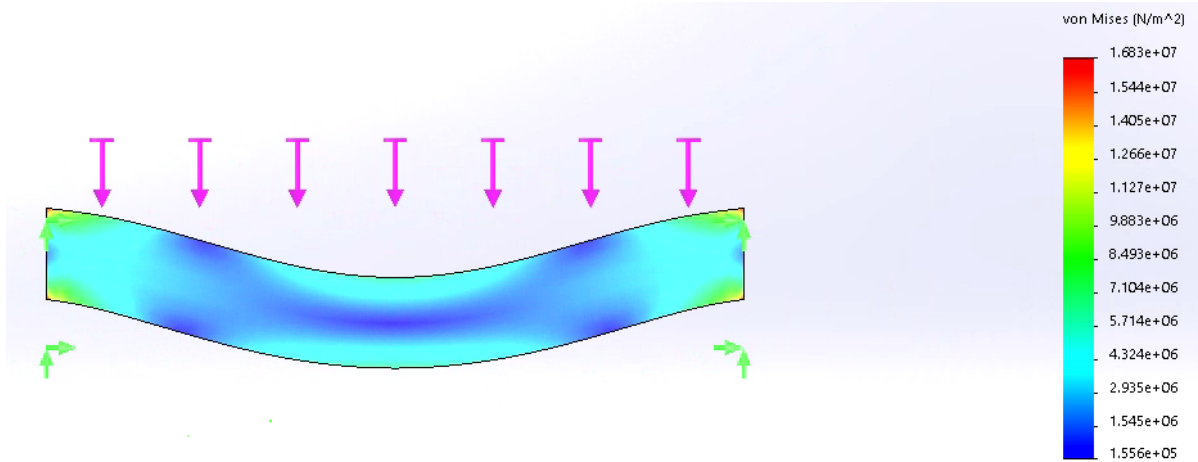


Figure 21: SolidWorks simulation displacement results of the platform (25mm thickness)



**Figure 22: SolidWorks simulation stress results of the platform (25mm thickness)**

From these two figures it can be seen that the the platform has a maximum displacement of 0.29 mm and a stress of 16.8 MPa. The displacement was determined to be acceptable and the stress found in the simulation was under the yield strength of PETG, which has a minimum yield strength of 28.3 MPa [6]. It should be noted that the simulation assumed a solid platform of PETG. With this in mind, we did not want to print with a smaller thickness because the platform was printed with 15% infill. With the results from the 2D simulation and the fact that the print was not solid, a 25 mm thickness for the platform was determined to be appropriate.

### 6.1.3 Battery Selection

The total power consumption [W] of our device is given in Eq. 12 below:

$$P_{tot} = P_{Arduino} + P_{bridge} + P_{GPS} + 2P_{motor} \quad (12)$$

where  $P_{Arduino}$ ,  $P_{bridge}$ ,  $P_{GPS}$ , and  $P_{motor}$  denote the respective power consumptions of the Arduino Leonardo, motor driver, GPS module, and the motors themselves [W]. Maximum power values (for a conservative calculation) for these components are 0.3, 0.18, 0.1, and  $2(2.6) = 5.2$  W, respectively [7, 8, 9, 10]. These values yield a total power consumption of 5.78 W. Note that this calculation neglects the internal resistance of the jumper cables and the power switch. Thus, we use a conservative value of 6 W for further calculations.

Equation 13 below describes the relationship between power and battery capacity:

$$C = It = \frac{Pt}{V} \quad (13)$$

where  $C$  is the battery capacity [Ah],  $I$  is the operating current [A],  $V$  is the operating voltage [V], and  $t$  is the operating time [hr]. An A23-size 12-Volt battery has a capacity of 0.06 mAh, meaning it can only supply the necessary 6 W of power for 0.12 hours = 7.2 minutes [11]. Even two of these batteries in parallel can only power the vehicle

for 14.4 minutes, which is not nearly long enough for the vehicle to reach its destination, especially considering the fact that the motors will slow down considerably as the batteries drain. Instead, two parallel 9-Volt batteries will be used. A 9-Volt battery has a capacity of 490 mAh, allowing these batteries to power the circuit for 1.47 hours, over 12 times longer than the 12-Volt batteries [12].

## 6.2 Design for Safety

This subsection explains what we consider the most common safety issues associated with our design will be, how they will affect our device and the operator, their severity, their likelihood of occurring, and possible ways to mitigate these issues.

### 6.2.1 Risk #1: Pinched Fingers

**Description:** Body parts being trapped and injured while the device is operating.

**Severity:** Negligible

**Probability:** Likely

**Mitigating Steps:** Add soft surfaces around potential pinch points on the vehicle.

### 6.2.2 Risk #2: Chassis Damage

**Description:** The device is damaged in such a way that it becomes inoperable.

**Severity:** Marginal

**Probability:** Occasional

**Mitigating Steps:** Holding the robot by the base with two hands; placing away from edge drops.

### 6.2.3 Risk #3: Shocks

**Description:** Shocks due to the wires and batteries being exposed on the device.

**Severity:** Marginal

**Probability:** Occasional

**Mitigating Steps:** Covering bare wires with electrical tape; adding a cover over electronics.

### 6.2.4 Risk #4: Tripping Hazard

**Description:** The device can serve as a hazard when operating on the ground.

**Severity:** Marginal

**Probability:** Seldom

**Mitigating Steps:** Take care to avoid the robot when it is operating on the ground.

### 6.2.5 Risk #5: Battery Overheats

**Description:** Components drawing too much power from the battery.

**Severity:** Critical

**Probability:** Seldom

**Mitigating Steps:** Using a sufficiently high-capacity battery (2 or more if needed).

### 6.2.6 Risk Assessment Heat Map

Fig. 23 below shows a heat map generated from the above risk assessment.

		Probability that something will go wrong				
		<b>Frequent</b> Likely to occur immediately or in a short period of time; expected to occur frequently	<b>Likely</b> Quite likely to occur in time	<b>Occasional</b> May occur in time	<b>Seldom</b> Not likely to occur but possible	<b>Unlikely</b> Unlikely to occur
Severity of risk	<b>Catastrophic</b>					
	<b>Critical</b>				Battery Overheats	
	<b>Marginal</b>			Chassis Damage Shocks	Tripping Hazard	
	<b>Negligible</b> hazard presents a minimal threat to safety, health, and well-being of participants; trivial		Pinched Fingers			

Figure 23: Risk assessment heat map.

Based on this heat map, pinched fingers and tripping are not very particularly important safety risks for our design due to their low severity and probability. More important safety risks to consider are chassis damage, shock, and the battery overheating because they are relatively more severe and more likely to occur. The most important of these risks are the hazards posed by the electronics, because these risks are dangerous to the user as well as to the device. Providing some kind of covering on top of all electronics and wiring would completely remove these dangers to the user, as they would no longer be able to come into contact with bar wire or an overly hot battery. This covering would also help mitigate the risk of drop damage, as it would provide more mechanical protection for the electronics.

### 6.3 Design for Manufacturing

The total parts required for the design that we constructed are listed in Table 3.



**Table 3: List of main parts used in construction of the final prototype.**

Part	Quantity	Description
Arduino Leonardo	1	Main controller for device.
Ultimate GPS Logger Shield	1	Device to retrieve GPS location data for positioning.
Brushed 12 V DC Motor	2	100:1 Metal Gearmotor 20Dx44L mm 12V CB with Extended Motor Shaft.
Motor Brackets	2	Pololu 20D mm Metal Gearmotor Bracket Pair.
Motor Drive Controller Board	1	L298N Dual H Bridge DC Stepper For Arduino.
Breadboard Jumper Wires	12	Assorted colors, sizes, and connection types.
Traxxas Slash 4x4 2WD 1/10 Short Course Truck Tires	2	120 mm outer diameter 12 mm hex.
MicroSD Card	1	Samsung 32 GB 95 MB/s microSDHC EVO Select Memory Card.
Tape Measure	1	Komelon SL2825 Self Lock 25-Foot Power Tape.
Battery Holders	3	9V battery Holder With ON/OFF Switch.
Batteries	3	Duracell 9V DC batteries.
Main Body	1	3D Printed PETG Custom Base.

It can therefore be seen from Table 3 that there are a total of 30 individual components required for the construction of our device, with the exception of fasteners. A total of 21 screws were used to attach all the parts together, including attaching components to the 3D printed base and components to each other, such as the motors and their brackets.

Of the 30 components used, we count approximately 10 Theoretically Necessary Components (TNC) without considering fasteners. These 10 components include: two motors, two wheels, a microSD card, three batteries, the base, and the control piece. It should be noted that the “control piece” includes the Leonardo Arduino, as well as the GPS Logger Shield and Drive Controller, which can all be combined into one component for manufacturing. The batteries need to be separate components for easy replacement, instead of the entire design having to be thrown out when the power supply dies (or cannot be recharged). The wheels need to be separate components to allow for free rotation, and to allow for them to be swapped for different wheels, depending on the terrain. The microSD card must be a separate component so that it can be removed and inserted into a computer in order to extract the information for analysis. The motors need to be separate components as well to allow for them to be easily replaced and maintained after and during use. Finally, the control system on the device—while multiple parts of it can be combined—is necessary to be a separate component since it can not be manufactured in the same way that the main base can be. This separation is due to the many delicate electronic components that are held within the control system.

As described above, many of the components listed in Table 3 can be paired down

through combination. For instance the wires, motor brackets, battery holders, and tail, can all be combined into the base, allowing it to be produced as a single part. This would allow for further increased structural integrity, as these parts would no longer be reliant on the fasteners that were used to attach them. The Arduino, GPS Logger, and Drive Controller can also all be combined into one main component that is able to perform all three functions, thereby eliminating the need to attach multiple control pieces to the base.

## 6.4 Design for Usability

1. The design may be difficult for someone with less-than-optimal vision due to the many small, intricate parts required to fit the vehicle within the canister and meet the weight requirement. There could also be difficulty with operation, since the entirety of the device's mechanics will function through Arduino code. A potential solution for this would be to have the small parts incased with only a few clearly labeled buttons visible, allowing for easier use (i.e. the user does not struggle to find on/off switch).
2. With a hearing impairment, the user may not be able to hear that the device is on if the DC motors make a high frequency noise, but the wheels are not spinning (vehicle stuck). This could be a potential problem if they go to pick it up thinking it is off, and it starts to move when the overall normal forces are lessened. A solution to this issue would be to have an LED indicator that alerts the user when the device is on by being illuminated.
3. A physical impairment would only slightly impede the operation of this device. Someone who is disabled in such a way that they are missing hands, fingers, etc. would have trouble carrying the device and even switching it on. Once on however, due to the intended autonomous nature, there should no longer be restrictions due to this impairment. We do not necessarily know of an alternate method of picking up the device; however, the power switch on could be made a large, visually-apparent button away from other components that could be damaged. This alteration would allow for the user to power on the vehicle without need of a delicate switch flip, but rather through a—perhaps violent—press.
4. Someone with a control impairment would potentially struggle for similar reasons as someone with trouble seeing. The components of the design are exposed and not within a shell, and therefore, someone without full motor control might struggle operating the device (i.e. turning it on or replacing a battery). The rest of the components may be impacted by mistake, leading to potential breakage or injury. The solution to this issue would follow in the same way as some of the others through a protective casing. if the user is rough with the device or impacts it accidentally, no components are damaged enough to cause malfunction or inoperability.

## 7 Final Prototype

### 7.1 Overview

Our design of the ARLISS canister vehicle was able to meet all three of our outcome goals. First, it was able to survive a drop of 2 feet off the ground and still function normally. Second, the vehicle was able to operate and navigate autonomously to a given set of GPS coordinates. Lastly, it was able to traverse rough and uneven terrain. Additionally, the size and weight constraints were met with the device being less than 1 kg and 120 mm  $\times$  220 mm. The design consisted of two wheels mounted to a 3D printed body, with two 12 V brushed DC motors. To account for the torque produced by the motors, a tape measure was attached to the back to prevent body rollover. This allowed for a rigid tail while in operation, with the ability to be folded up for storage within a cylinder. An Arduino Leonardo, GPS logger and motor driver controller were used to control the motion of the vehicle. Extra batteries were used to ensure battery life was long enough for the vehicle to reach its destination. For future development, it is recommended that our design is improved through the addition of renewable solar power, use of a Raspberry Pi instead of the Arduino Leonardo, and stronger wheel brackets to eliminate any concerns of deformation due to impact.

### 7.2 Documentation

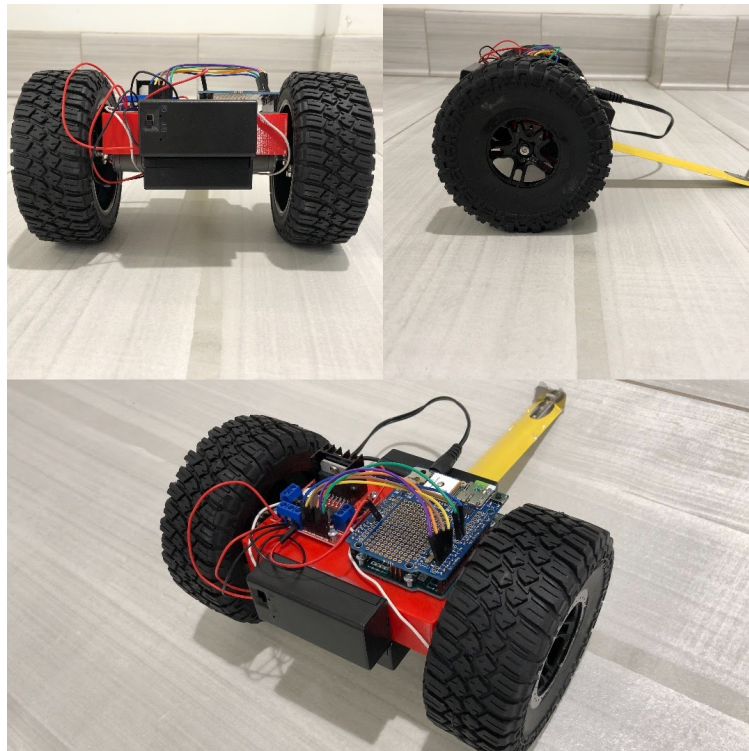


Figure 24: Photographs of final prototype.

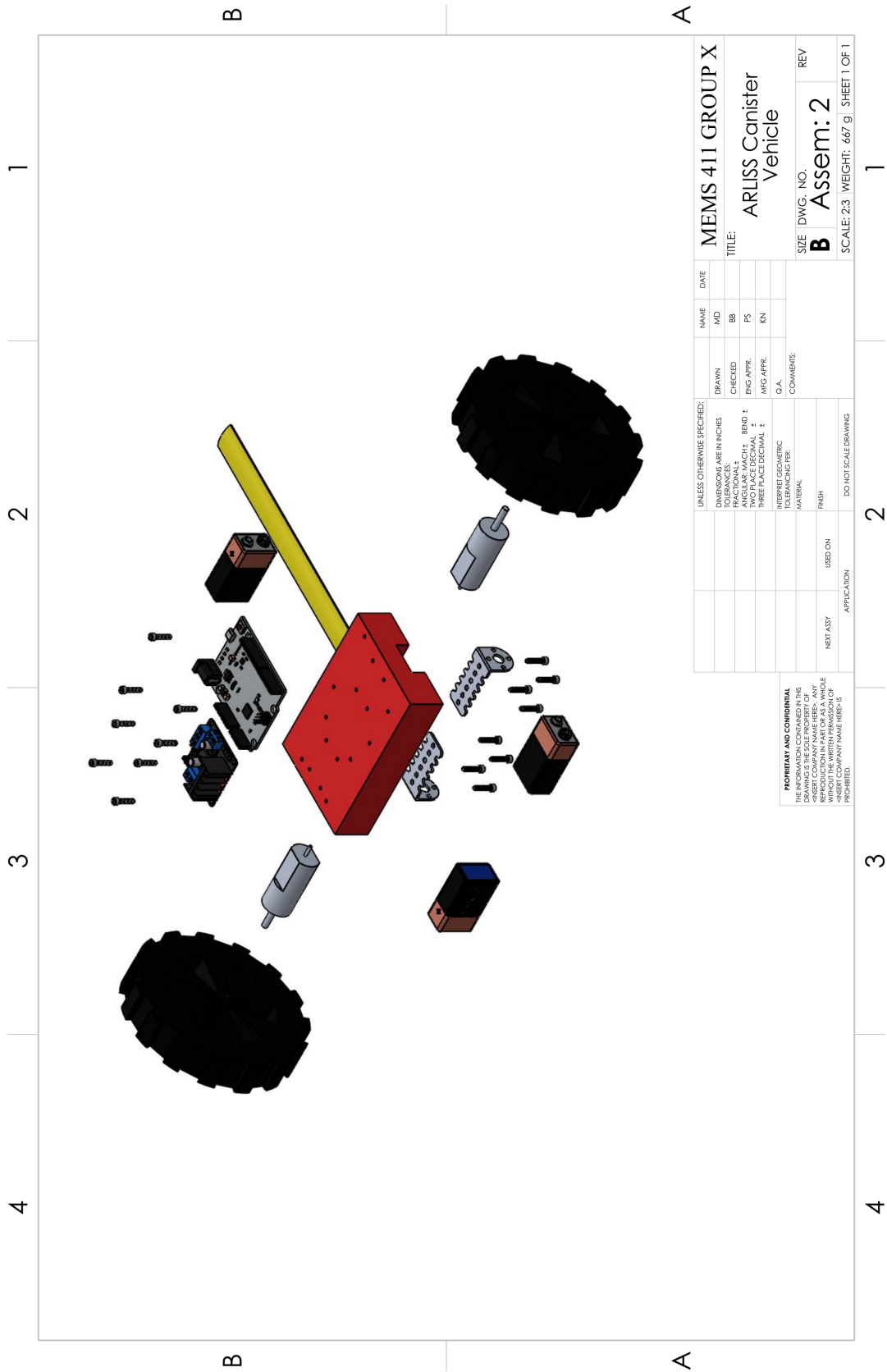


Figure 25: Final prototype: CAD exploded view.

UNLESS OTHERWISE SPECIFIED:		NAME	DATE	MEMS 411 GROUP X	
DIMENSIONS ARE IN INCHES	DRAWN	MD		TITLE:	
TOLERANCES:	CHECKED	BB		ARLUSS Canister Vehicle	
FRACTIONS: 1/16" > 1/8" > 1/4" > 3/8" > 1/2" > 5/8" > 3/4" > 1" > 1 1/2" > 2" > 3" > 4" > 6" > 8" > 12"	ENG. APPR.	PS		SIZE DWG. NO. REV	
DECIMALS: 0.0001" > 0.0005" > 0.001" > 0.002" > 0.005" > 0.010" > 0.015" > 0.030" > 0.060" > 0.125" > 0.250" > 0.500" > 1.000" > 1.500" > 2.000" > 3.000" > 4.000" > 6.000" > 8.000" > 12.000"	MFG. APPR.	KN		<b>B</b>	Assem: 2
FIT: H11, H9, H7, H6, H5, H4, H3, H2, H1, K6, K5, K4, K3, K2, K1, M6, M5, M4, M3, M2, M1, N6, N5, N4, N3, N2, N1, P6, P5, P4, P3, P2, P1, R6, R5, R4, R3, R2, R1, S6, S5, S4, S3, S2, S1, T6, T5, T4, T3, T2, T1, U6, U5, U4, U3, U2, U1, V6, V5, V4, V3, V2, V1, W6, W5, W4, W3, W2, W1, X6, X5, X4, X3, X2, X1, Y6, Y5, Y4, Y3, Y2, Y1, Z6, Z5, Z4, Z3, Z2, Z1	G.A.			SCALE: 2:3	WEIGHT: 667 g
INTERPRET GEOMETRIC TOLERANCING PER:	COMMENTS:				SHEET 1 OF 1
MATERIAL:					
FINISH:					
DO NOT SCALE DRAWING					
APPLICATION					
USED ON					
NEXT ASSY					

**PROPRIETARY AND CONFIDENTIAL**  
 THIS DRAWING IS THE SOLE PROPERTY OF  
 ARLUSS COMPANY. NO REPRODUCTION OR  
 DISSEMINATION OF THIS DRAWING IS  
 ALLOWED WITHOUT THE WRITTEN PERMISSION OF  
 ARLUSS COMPANY. NAME HEREIN IS  
 PROHIBITED.

## Bibliography

- [1] iRobot. *Roomba s9+ Robot Vacuum Cleaner*. 2021. URL: <https://store.irobot.com/default/roomba-vacuuming-robot-vacuum-irobot-roomba-s9-plus/s955020.html>.
- [2] Aliexpress. *WZY569 Smart RC Tank Car Truck Robot Platform Climbin Metal Tank Chassis DIY 350 RPM CNC Alloy body+4 Plastic Tracks + 4 Motors*. URL: <https://www.aliexpress.com/item/32805317760.html>.
- [3] Aniomylone. *LEGO MOC-4355 UCS Hailfire Droid (Star Wars Ultimate Collector Series 2015)*. 2015. URL: <https://rebrickable.com/mocs/MOC-4355/Aniomylone/ucs-hailfire-droid/#details>.
- [4] Nikolaos P. PapanikolopoulosDonald G. KrantzRichard M. VoylesJohn A. BusheyAlan N. JohnsonBradley J. NelsonPaul E. RybskiKathleen A. GriggsII Ellison C. Urban. “Miniature Robotic Vehicle And Methods Of Controlling Same”. US6548982B1. 2002. URL: <https://patents.google.com/patent/KR100683271B1/en?q=solar+powered+miniature+robotic+vehicle&oq=solar+powered+miniature+robotic+vehicle#title>.
- [5] Kim Kyungmin. “Solar Car Toy”. KR100683271B1. 2004. URL: <https://patentimages.storage.googleapis.com/41/04/60/b4d34703720996/US6548982.pdf>.
- [6] MatWeb. *Overview of materials for PETG Copolyester*. URL: [http://www.matweb.com/search/datasheet\\_print.aspx?matguid=4de1c85bb946406a86c52b688e3810d0](http://www.matweb.com/search/datasheet_print.aspx?matguid=4de1c85bb946406a86c52b688e3810d0).
- [7] Arduino. *ArduinoBoardLeonardo - Technical Specs*. URL: [https://www.arduino.cc/en/Main/Arduino\\_BoardLeonardo](https://www.arduino.cc/en/Main/Arduino_BoardLeonardo).
- [8] Qunqi. *L298N Motor Drive Controller Board - Specification*. URL: [https://www.amazon.com/gp/product/B014KMHSW6/ref=ppx\\_yo\\_dt\\_b\\_asin\\_image\\_o06\\_s00?ie=UTF8&psc=1](https://www.amazon.com/gp/product/B014KMHSW6/ref=ppx_yo_dt_b_asin_image_o06_s00?ie=UTF8&psc=1).
- [9] Adafruit. *Adafruit Ultimate GPS Logger Shield - Overview*. URL: <https://learn.adafruit.com/adafruit-ultimate-gps-logger-shield>.
- [10] Pololu Robotics & Electronics. *100:1 Metal Gearmotor 20Dx44L mm 12V CB with Extended Motor Shaft*. URL: <https://www.pololu.com/product/3490>.
- [11] MJKAA. *Pack of 5 A23 Battery 12V Alkaline 60 mAh [Ultra Power] - 12 Volt*. URL: [https://www.amazon.com/gp/product/B07TKQZ8VB/ref=ppx\\_yo\\_dt\\_b\\_asin\\_image\\_o05\\_s00?ie=UTF8&psc=1](https://www.amazon.com/gp/product/B07TKQZ8VB/ref=ppx_yo_dt_b_asin_image_o05_s00?ie=UTF8&psc=1).
- [12] PowerStream. *Discharge tests of 9 Volt transistor radio style batteries*. URL: <https://www.powerstream.com/9V-Alkaline-tests.htm>.

## A Software Code - MATLAB

```
1 % Patrick Smith
2 % 4/15/2021
3
4 clear all; close all; clc;
5
6 load("target.txt");
7 load("path.txt");
8
9 x = -4800*(path(:,2)-path(1,2));
10 y = 6068*(path(:,1)-path(1,1));
11 targ = [-4800*(target(2)-path(1,2)) 6068*(target(1)-path(1,1))];
12 d = path(:,3);
13 t = (0:0.5:(length(d)-1)/2)/60';
14
15 figure(1);
16 xlabel('Longitudinal Displacement [ft]','FontSize',12); axis equal; ...
17 hold on;
18 ylabel('Latitudinal Displacement [ft]','FontSize',12);
19 title('ARLISS Vehicle Trajectory','FontSize',14);
20 plot(0,0,'g.','MarkerSize',30);
21 plot(targ(1),targ(2),'r*','MarkerSize',10,'LineWidth',2);
22 plot(ksidedpoly(100,'Center',targ,'Radius',5),'FaceColor','r');
23 comet(x,y,0); plot(x,y,'b-'); hold off;
24 legend('Landing Site','Destination','Target Radius','Vehicle Path','...
25 Location','NW');
26
27 figure(2);
28 plot(t,d,'b-','LineWidth',2);
29 xlabel('Time [min]','FontSize',12); xlim([t(1) t(end)]);
30 ylabel('Distance From Target [ft]','FontSize',12);
31 title('ARLISS Vehicle Distance From Target vs Time','FontSize',14);
```

code/plot\_path.m

## B Software Code - Arduino

```
1 // Patrick Smith & Group X
2 // 4/28/2021
3
4 #include <Adafruit_GPS.h>
5 #include <SPI.h>
6 #include <SD.h>
7
8 SoftwareSerial mySerial(8,7);
9 Adafruit_GPS GPS(&mySerial);
10
11 // #define PMTK_SET_NMEA_UPDATE_1HZ "$PMTK220,1000*1F"
12 #define PMTK_SET_NMEA_UPDATE_5HZ "$PMTK220,200*2C"
13 // #define PMTK_SET_NMEA_UPDATE_10HZ "$PMTK220,100*2F"
14 #define PMTK_SET_NMEA_OUTPUT_RMONLY "$PMTK314...
15     ,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0*29"
16
17 #define highR 2
18 #define lowR 3
19 #define highL 5
20 #define lowL 4
21 #define pwmR 9
22 #define pwmL 6
23 #define SDpin 10
24
25 #define targX 3838.8782
26 #define targY 9018.8643
27 #define targR 5
28 #define sens 0.25
29
30 #define FLT_MAX 3.4028235E38
31
32 void setup() {
33     pinMode(highR,OUTPUT);
34     pinMode(lowR,OUTPUT);
35     pinMode(highL,OUTPUT);
36     pinMode(lowL,OUTPUT);
37     pinMode(pwmR,OUTPUT);
38     pinMode(pwmL,OUTPUT);
39     pinMode(SDpin,OUTPUT);
40     Serial.begin(115200);
41     //delay(2000);
42     //Serial.println("Connecting SD");
43     while (!SD.begin(SDpin, 11, 12, 13))
44         delay(1);
45     if (SD.exists("path.txt"))
46         SD.remove("path.txt");
47     GPS.begin(9600);
48     GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMONLY);
```

```

GPS.sendCommand(PMTK_SET_NMEA_UPDATE_5HZ);
49 //Serial.println("Connecting GPS");
while (!GPS.fix)
51   ping(false);
//Serial.println("Success");
53 delay(8000);
}
55
float x;
57 float y;
float prevDist = FLT_MAX;
59 float dist;
char state = 1;
61 uint32_t timer = millis();
uint32_t timer2 = millis();
63 bool fin = false;

65 void loop() {
  ping(true);
67   if (millis()-timer > 500) {
    findDist();
69    logPos();
    //Serial.println(String(dist,4)+"", "+String(prevDist,4));
71    if (dist < targR) {
      brake();
73      state = 1;
      if (!fin) {
75        fin = true;
        logPos();
77      }
      delay(1);
79      return;
    }
81    switch (state) {
      case 1: //circling
83        if (fin)
          fin = false;
85        circle();
        //Serial.println("circling");
87        if (dist-prevDist > sens) {
          //Serial.println("Stop Circle");
89          turn();
          drive();
91          //delay(1000);
          prevDist = FLT_MAX;
93          state = 2;
          timer2 = millis();
95        }
        break;
97      case 2: //straight
        drive();
99        //Serial.println("driving");

```



```

101     if ( millis()-timer2 > 10000) {
102         //Serial.println("Stop straight");
103         prevDist = FLT_MAX;
104         state = 1;
105     }
106     break;
107 }
108 if (dist < prevDist)
109     prevDist = dist;
110 timer = millis();
111 }
112 delay(1);
113 }
114
115 void brake(void) {
116     for (int i=255; i>=0; i--) {
117         analogWrite(pwmR, i);
118         analogWrite(pwmL, i);
119         delay(3);
120     }
121 }
122
123 void findDist(void) {
124     dist = sqrt(sq(6068.0*(x-targX))+sq(4800.0*(y-targY)));
125 }
126
127 void turn(void) {
128     brake();
129     digitalWrite(highR, LOW);
130     digitalWrite(lowR, HIGH);
131     digitalWrite(highL, HIGH);
132     digitalWrite(lowL, LOW);
133     analogWrite(pwmR, 255);
134     analogWrite(pwmL, 255);
135     delay(300);
136     brake();
137 }
138
139 void circle(void) {
140     digitalWrite(highR, HIGH);
141     digitalWrite(lowR, LOW);
142     digitalWrite(highL, HIGH);
143     digitalWrite(lowL, LOW);
144     analogWrite(pwmR, 255);
145     analogWrite(pwmL, 205);
146 }
147
148 void drive(void) {
149     digitalWrite(highR, HIGH);
150     digitalWrite(lowR, LOW);
151     digitalWrite(highL, HIGH);
152     digitalWrite(lowL, LOW);

```

```

153 analogWrite (pwmR, 255);
    analogWrite (pwmL, 225);
    }
155
157 void ping (bool b) {
    GPS.read ();
    if (GPS.newNMEAreceived ()) {
159     if (!GPS.parse (GPS.lastNMEA ())) {
        if (b)
161         return;
        else
163         delay (1);
    }
165 }
    x = GPS.latitude;
167 y = GPS.longitude;
    }
169
171 void logPos (void) {
    File path = SD.open ("path.txt", FILE_WRITE);
    path.println (String (x, 4) + '\t' + String (y, 4) + '\t' + String (dist, 4));
173 path.close ();
    }

```