

Semi-autonomous All-terrain Vehicle

Group 12

Kris Choudhury - Computer Engineer

Sergio Gonzalez - Computer Engineer

Devon Wilkerson - Computer Engineer

Benjamin Goerdts - Electrical Engineer

August 3, 2021

Contents

1	Project Narrative	1
1.1	Background	1
1.2	Project Description	1
1.3	Motivation	1
2	Requirement Specifications	2
2.1	Design Requirements	2
2.2	Definition of All-terrain	3
2.2.1	Silt	3
2.2.2	Sand	3
2.2.3	Gravel	4
2.2.4	Rock	4
2.2.5	Soil	5
2.2.6	Grass	5
2.2.7	Asphalt Concrete	5
2.2.8	Concrete	5
2.2.9	Wood	5
2.2.10	Water	6
2.2.11	Colloid	6
2.3	Definition of Autonomy	6
2.3.1	Geofencing	7
2.3.2	Obstacle Detection and Classification	7
2.3.3	Attention	8
2.3.4	Waypoints	8
2.3.5	Automation Level	9
2.4	House of Quality Diagram	10
2.4.1	Long Runtime	11
2.4.2	Quick Recharge Time	12
2.4.3	Accessible and Easy To Use GUI	12
2.4.4	Reliability	12
2.4.5	Safety	13
2.4.6	High Accuracy	13
3	Block Diagram	14
4	Budget	16
4.1	Bill of Materials	18
5	Project Milestones	20
6	Related Research	22
6.1	Similar Devices	22
6.1.1	Curiosity Rover	22
6.1.2	Infosys' Driverless Golf Cart	23
6.1.3	Yamaha's Driverless Buggy	23
6.1.4	University of Central Florida's Autonomous Shuttle	23
6.1.5	Military UAVs	24

7	Vehicle	24
7.1	Retrofitting	25
7.2	Hand-built	26
7.3	Power Wheels	27
7.4	Comparison	28
7.4.1	Chance of Failure	28
7.4.2	Time investment	28
7.4.3	Reliability	29
7.4.4	Possible Repair Cost	30
7.4.5	Cost	30
7.5	Final Decision	31
8	Serial Communication	31
8.1	UART	32
8.2	SPI	33
8.3	I2C	34
9	Batteries	36
9.1	Power Wheels	36
9.1.1	12 Volt OEM	36
9.1.2	18V Milwaukee	37
9.1.3	Comparison	38
9.2	Golf Cart & Hand-Built	38
9.2.1	12V Marine Deep Cycle	39
9.2.2	6V Golf Cart	39
9.2.3	Comparison	40
9.3	All Applications	41
9.3.1	18650 Cells	41
10	Solar Panels	43
10.1	100W Flexible Panel	43
10.2	300W Fixed Panels	44
10.3	Charge Controller	45
10.4	Comparison	46
11	Motor	46
11.1	Propulsion	47
11.1.1	RC 550	47
11.1.2	RC 775	48
11.1.3	MY1020	49
11.1.4	Club Car 36/48V Motor	50
11.1.5	Comparison	51
11.2	Steering	52
11.2.1	Tonegawa 050 Servo	52
11.2.2	Automotive Windshield Wiper Motor	53
11.2.3	Nema 34	54
11.2.4	Firgelli Linear Actuator	54
11.2.5	Comparison	56

12 Wheel Hubs & Brakes	57
12.1 Power Wheels Brakes	58
12.2 Golf Cart Brakes	58
12.3 Home-Built Chassis Brakes	59
12.4 Comparison	60
13 Microcontroller	61
13.1 MSP430FR6989	61
13.2 ATMEGA328p	63
13.3 ATMEGA2560	65
13.4 Comparison	67
13.4.1 Core Processor	67
13.4.2 Core Size	68
13.4.3 Speed	68
13.4.4 Connectivity	69
13.4.5 Program Memory Size	69
13.4.6 Program Memory Type	70
13.4.7 EEPROM Size	70
13.4.8 RAM size	71
13.4.9 Voltage	71
13.4.10 Mounting Type	72
13.4.11 Number of I/O	73
13.4.12 Cost	74
13.5 Final Decision	74
14 GSM Module	75
14.1 SIM900 Quad Band GSM Module	75
14.2 XYGStudy GSM	75
14.3 Shield for Arduino with GPS - SIM7600CE	75
14.4 Comparison	76
14.5 Final Decision	76
15 GPS Chip	76
15.1 NEO-6 GPS Module	77
15.2 Grove GPS module	77
15.3 Grove Air530 GPS Module	77
15.4 Comparison	77
15.5 Final Decision	78
16 Interface Computer - Raspberry Pi 4	78
17 Touchscreen Display	80
17.1 Elo TouchSystems Touchscreen Display	81
17.2 Hannspree Multi-Touch Screen Monitor	81
17.3 BuyDisplay Serial SPI I2C Module Display	82
17.4 Raspberry Pi LCD Screen	83
17.5 Comparison	83
17.5.1 Display Size	84
17.5.2 Display Resolution	84

17.5.3	Display Connectivity	84
17.5.4	Total Cost	85
17.6	Final Decision	86
18	AI Computing Device	86
18.1	NVIDIA Jetson TX2 NX	86
18.2	NVIDIA Jetson Nano	88
18.3	Raspberry Pi Compute Module 4	89
18.4	Comparison	91
18.4.1	AI Performance	91
18.4.2	CPU	92
18.4.3	GPU	93
18.4.4	Memory	93
18.4.5	Display	94
18.4.6	Total Cost	94
18.5	Final Decision	95
19	Sensors	95
19.1	Camera - Kinect	96
19.2	NFC Sensor	96
19.2.1	DFR0231-H	96
19.2.2	Grove NFC	97
19.2.3	Comparison and Decision	97
19.3	Temperature Sensor	98
19.3.1	Adafruit MCP9808	99
19.3.2	SEN0206	99
19.3.3	DS18B20	100
19.3.4	AM2320	101
19.3.5	Comparison and Decision	101
19.4	Humidity Sensor	102
19.4.1	AM2320	102
19.4.2	HTS221	102
19.4.3	Comparison and Decision	103
19.5	Oxygen Sensor	104
19.5.1	MIX8410	104
19.5.2	SEN0322	104
19.5.3	Comparison and Decision	105
19.6	Carbon Dioxide Sensor	106
19.6.1	T6703-5k	106
19.6.2	SEN0219	106
19.6.3	Comparison and Decision	107
19.7	Ultraviolet Sensor	108
19.7.1	VEML6070	108
19.7.2	ML8511	108
19.7.3	Comparison and Decision	109
19.8	Particulate Matter Sensor	110
19.8.1	PMSA003I	110
19.9	Volatile Organic Compounds Sensor	111

19.9.1	SGP40	111
19.10	Light sensor	112
19.10.1	Adafruit VEML7700	112
19.10.2	Adafruit TSL2591	113
19.10.3	Comparison and Decision	113
19.11	Proximity sensor	114
19.11.1	HC-SR04	114
19.11.2	MB1040 LV-MaxSonar-EZ4	115
19.11.3	Comparisons and Decision	116
20	Testing and Design	117
20.1	User Interface	117
20.1.1	Options for User Interface	117
20.1.2	HTML5 Web Application	118
20.1.3	Python Application	118
20.1.4	Windows Form Application	119
20.1.5	Comparison	119
20.1.6	Final Decision	120
20.2	Backend Connection	120
20.2.1	Serial Communication	120
20.2.2	Wireless Communication	121
20.2.3	Comparison	121
20.2.4	Final Decision	121
20.3	Jetson Nano Test	122
20.4	ATMEGA2560 & ATMEGA328p Test	122
A	Permissions	

List of Figures

1	SAE Autonomy Chart [1]	10
2	House of Quality Diagram	11
3	Block Diagram	15
4	Curiosity Rover - Free Use Photo	22
5	UCF Autonomous Shuttle - Reprinted with permission from –	24
6	UART IC Communication	32
7	UART Package Structure	33
8	SPI Connection Structure	34
9	I2C one Controller to many Peripherals	35
10	I2C many Controllers to many Peripherals	35
11	I2C Package Structure	36
12	Power Wheels Battery	37
13	Milwaukee M18 Battery	37
14	Interstate Deep Cycle Battery	39
15	Interstate 6V Golf Cart Battery	40
16	18650 Pack Configuration	41
17	18650 Cell Holder [2]	42
18	100W Flexible Solar Panel	44
19	300W Fixed Solar Panel	45
20	Solar Panel Charge Controller	46
21	RC-550 Motor	47
22	RC775 Motor	49
23	MY1020 Motor	49
24	Club Car Dual Voltage Motor	50
25	Tonegawa 050 Servo Motor	52
26	Automotive Wiper Motor	53
27	Nema 34 Stepper Motor	54
28	Firgelli Linear Actuator	55
29	Golf Cart Drum Brake	59
30	Do It Yourself Disc Brake Kit	60
31	MSP430FR6989 Microcontroller	62
32	ATMEGA328p Microcontroller	65
33	ATMEGA2560 Microcontroller	65
34	Raspberry Pi 4B - Reprinted with permission from Adafruit	79
35	Elo TouchSystems Touchscreen Display	81
36	BuyDisplay Serial SPI I2C Module Display	82
37	Raspberry Pi 7-Inch LCD Screen	83
38	Jetson TX2 NX Module - Reprinted with permission from NVIDIA	87
39	Jetson Nano Development Kit - Reprinted with permission from NVIDIA	89
40	Raspberry Pi Compute Module 4 - Reprinted with permission from Adafruit	90
41	Raspberry Pi Compute Module 4 IO Board	91
42	Xbox 360 Kinect Sensor	96
43	DFRobot DFR0231-H NFC Module	97
44	SeeedStudio Grove NFC Module	97

45	MCP9808	99
46	SEN02068	99
47	SEN0206 Field of View Graph	100
48	Waterproof DS18B20	100
49	AM2320	101
50	Adafruit HTS221	103
51	Seedstudio MIX8410	104
52	DFRobot SEN0322	105
53	T6703-5k CO2 sensor	106
54	DFRobot SEN0219	107
55	Adafruit VEML6070	108
56	DFRobot ML8511	109
57	ML8511 Analog Voltage Ouput Graph	109
58	PMSA003I Sensor from Plantower	111
59	SGP40 from Adafruit	111
60	VEML7700	112
61	VEML7700 Relative Radiant Sensitivity vs. Angular Displacement .	112
62	TSL2591	113
63	HC-SR04 Ultrasonic Distance Sensor	115
64	MB1040 LZ-MaxSonar-EZ4	116
65	Jetson Nano Object Recognition Test	122
66	ATMEGA2560 & ATMEGA328p I2C Test	123
67	NVIDIA Permission	
68	ELO TouchSystems Permission	
69	BuyDisplay Permission	
70	AdaFruit Permission	
71	Circuit Basics Permission	
72	Solar Panels & Controller Permission	
73	Go Kart Axle Permission	
74	Nema 34 Stepper Permission	
75	Interstate Battery Permission	
76	Universal Windshield Motor Permission	
77	AutoTex Motor Permission	
78	Golf Cart Motor Permission	
79	RC Motor, Servo, Actuator Permission	
80	18650 Cell Holder Permission	

List of Tables

1	Requirements Table	2
2	Rough Obstacles Table	8
3	Item List Table	18
4	Bill of Materials Table	20
5	Project Milestones Table (Senior Design 1)	21
6	Project Milestones Table (Senior Design 2)	21
7	Retrofitting Comparison Factors	26
8	Hand-Built Vehicle Comparison Factors	26

9	Power Wheels Comparison Factors	27
10	Lithium-Ion vs Sealed Lead Acid Battery Comparison Table	38
11	12V vs 6V Battery Options	41
12	Comparison of Drive Motors	52
13	Linear Actuator Physical Dimensions Based on Stroke Length	56
14	Comparison of Steering Motors	57
15	Comparison of Brake System Types	58
16	MSP430FR6989 Specifications	63
17	ATMEGA328p Specifications	64
18	ATMEGA2560 Specifications	66
19	GSM Comparison Table	76
20	GPS Comparison Table	78
21	Raspberry Pi 4B Specifications	79
22	Comparison Table for Touchscreen Displays	84
23	NVIDIA Jetson TX2 NX Specifications	87
24	NVIDIA Jetson Nano Specifications	88
25	Raspberry Pi Compute Module 4 Specifications	90
26	NFC Sensor Comparison Table	98
27	Temperature Sensor Comparison Table	101
28	Humidity Sensor Comparison Table	103
29	Oxygen Sensor Comparison Table	105
30	Carbon Dioxide Sensor Comparison Table	107
31	Ultra Violet light Sensor Comparison Table	110
32	Light Sensor Comparison Table	114
33	HC-SR04 Specifications	115
34	MB1040 LV-MaxSonar-EZ4 Specs	116
35	Proximity Sensor Comparison Table	117

1 Project Narrative

1.1 Background

The exploration of hazardous and unsafe environments is an ongoing obstacle mankind is constantly trying to overcome. Whether it's the deepest trenches in the ocean, the peak of a mountain, or even into the furthest reaches of space, we do our best to conquer the next frontier. In order to reach these goals, technology has been our greatest ally in assisting to achieve them. Vehicles, in particular, are one of the greatest technologies we have made to date. They're capable of transporting us through many terrains, such as: land, sea, air, space, and underwater.

The focus of this project will be on land based transportation. The ability to move across the ground faster than on foot is an important part of today's society. In fact, without modern vehicles, it is impossible to live as we do today. As our technology progresses, so do our vehicles. All modern transportation vehicles contain one or more microcontrollers. These computers control all of the electrical components of the vehicle. Many higher end models have multiple computers that control different parts of the vehicle. Along with transportation vehicles, we've developed vehicles that were made to journey through inhospitable areas.

The most famous of these vehicles is undoubtedly Curiosity, the Mars rover created by NASA. Curiosity is a rover capable of semi-autonomous operation, which allows it to travel the harsh Mars landscape through limited autonomy, while also getting orders from a ground control team on Earth. Tesla is also creating semi-autonomous vehicles that are capable of navigating traffic, driving within road markings, and to some degree self driving without an operator in the vehicle. Closer to our project scope is an autonomous vehicle made by Infosys, an Indian technology company. This vehicle was retrofitted from an existing vehicle, and adapted it to be able to drive autonomously using a "Drive-by-Wire" system developed by the company. Yamaha, a Japanese vehicle manufacturer, is also producing a driverless vehicle used for public transport and cargo transport.

1.2 Project Description

The concept for our project is in the scope of all the vehicles mentioned in the previous section. Our team aims to design a vehicle capable of semi-autonomous driving and navigation with a feature rich web portal for statistics and control. This vehicle will have an abundance of sensors on it to gather data about the surrounding environment. These sensors include: temperature sensors, humidity sensors, oxygen sensors, carbon dioxide sensors, ultraviolet sensors, particulate matter sensors, volatile organic compound sensors, ozone sensors, light sensors, ultrasonic (proximity) sensors, and multiple cameras. This data will aid in the study of the condition of the environment, and also give us insight to the habitability of the areas the vehicle traversed.

1.3 Motivation

This project is to demonstrate our collective knowledge and experienced gained at the University of Central Florida. Our goal is to work as a group to produce a working and scalable prototype that will push us to learn more and continue

moving forward with our education. We hope to design and present a product to show future employers the caliber at which we're capable of producing products, as well as give our professors a reason to be proud of us as we finish our last milestone before we obtain our degrees.

2 Requirement Specifications

2.1 Design Requirements

Requirement	Description
Run Time	Must run a minimum of 3 hours.
Charge Time	Must charge in a maximum of 8 hours.
Manual Driving Speed	Must be able to maintain a manual driving speed of 32km/h.
Autonomous Driving Speed	Must be able to maintain an autonomous driving speed of 8km/h.
GPS Display	Must display correct GPS coordinates within 2m.
Wifi/GSM Connectivity	Must be Wifi/GSM connected at 5Mb/s for live data transmission.
Planning/Routing Capabilities	Must be able to plan and execute routing within 10s.
Quick GUI Feedback	Screen must give feedback within 100ms.
Occupant Weight Limit	2 seats for occupants with a 175lb limit each.
Maximum Weight Limit	Must be able to hold a maximum of 400lbs
Object Detection	Must detect external objects within 1000ms at 10m.

Table 1: Requirements Table

2.2 Definition of All-terrain

All-terrain is an ambiguous term used to describe any vehicle capable of driving across multiple types of terrain. This section will define [3] exactly what types of terrain material our vehicle will be able to drive on. The vehicle may be capable of driving across terrain composed of a material not listed (or not yet defined/discovered), but it will not be built specifically for those types of terrain, and any attempt to drive the vehicle over the terrain could result in damage to the vehicle, its load, or injury to the passengers.

2.2.1 Silt

Silt is classified as any particle ranging from 1 to 100 μm . Dust will be considered silt in this definition since they have a similar particle size. Sand and gravel can fall into this category under certain circumstances, but will be considered separate materials for a more accurate classification. Any claims made will be under the assumption the dust fall is natural, and has not been compacted, wetted, or disturbed.

- Particle ranging from 1 to 50 μm :
This vehicle will be able to drive on any dust particle of size 50 μmeters or less.
- Particles ranging from 50 to 100 μm :
This vehicle will be able to drive on a dust particle of less than 100 μ if the height of the dust is no larger than 3/4 the radius of the tire.

2.2.2 Sand

Sand is classified as any particle ranging from 62.5 to 2,000 μm . It is often found near bodies of water such as: beaches, rivers, and lakes. Sand also makes up the majority of deserts. The main compound that makes up sand is silicon dioxide in the form of quartz. Any claims made will be under the assumption the sand has not been compacted, wetted, or disturbed.

- Particles ranging from 62.5 to 125 μm :
Sand of this size is considered very fine sand. The vehicle will be able to drive over this type of sand that is less than 20cm in height and no more than 1% in grade.
- Particles ranging from 125 to 250 μm :
Sand of this size is considered fine sand. The vehicle will be able to drive over this type of sand that is less than 20cm in height and no more than 1% in grade.
- Particles ranging from 250 to 500 μm :
Sand of this size is considered medium sand. The vehicle will be able to drive over this type of sand that is less than 20cm in height and no more than 1% in grade.

- Particles ranging from 500 to 1,000 μm :
Sand of this size is considered coarse sand. The vehicle will be able to drive over this type of sand that is less than 10cm in height and no more than 1% in grade.
- Particles ranging from 1,000 to 2,000 μm :
Sand of this size is considered very coarse sand. The vehicle will be able to drive over this type of sand that is less than 5cm in height and no more than 1% in grade.

2.2.3 Gravel

Gravel is classified as any particle ranging from 2,000 to 65,000 μm . Gravel can be found almost anywhere, and can be either natural or man made. It is a granular material that is mostly made up of eroded rocks. Any claims made will be under the assumption that gravel is the only material present.

- Particles ranging from 2,000 to 4,000 μm :
Gravel of this size is considered very fine gravel. The vehicle will be able to drive over this type of gravel that is less than 5cm in height and no more than 1% in grade.
- Particles ranging from 4,000 to 8,000 μm :
Gravel of this size is considered fine gravel. The vehicle will be able to drive over this type of gravel that is less than 5cm in height and no more than 1% in grade.
- Particles ranging from 8,000 to 15,000 μm :
Gravel of this size is considered medium gravel. The vehicle will be able to drive over this type of gravel that is less than 15cm in height and no more than 1% in grade.
- Particles ranging from 15,000 to 30,000 μm :
Gravel of this size is considered coarse gravel. The vehicle will be able to drive over this type of gravel that is less than 20cm in height and no more than 1% in grade.
- Particles ranging from 30,000 to 65,000 μm :
Gravel of this size is considered very coarse gravel. The vehicle will be able to drive over this type of gravel that is less than 20cm in height and no more than 1% in grade.

2.2.4 Rock

Rock is classified as any particle ranging from 65,000 to 256,000 μm . Any claims made will be under the assumption the rock has not been compacted, wetted, or disturbed, and that rock is the only material present.

- Particles ranging from 65,000 to 125,000 μm :
The vehicle will be able to drive over rocks of this size assuming the surface of the rock is 70% uniform, does not have any sharp edges, and is less than 2% in grade.

- Particles ranging from 125,000 to 256,000 μm :
The vehicle will not be able to drive over rocks of this size.

2.2.5 Soil

Soil is classified as any particle ranging from 1 to 32,000 μm . Any claims made will be under the assumption the soil has not been compacted, wetted, or disturbed, and that soil is the only material present. The vehicle will be able to drive over soil that is less than 10cm in height and no more than 1% in grade.

2.2.6 Grass

Grass is a ubiquitous family of flowering plants known as grasses. For this classification we will assume that grass makes up 90% of the plant matter present, and the grass will be rooted in a compound of comprised of 75% soil or higher. Assuming these restrictions, the vehicle will be able to drive on any grass that is less than 30cm in length, and on soil that is no more than 1% in grade.

2.2.7 Asphalt Concrete

Asphalt concrete is a material that made up of a mineral aggregate compound bound together with asphalt, a petroleum based substance. For this classification we will assume the asphalt concrete is the only material present, and will be assumed to be bare, which means unpainted or treated. Assuming these restrictions, the vehicle will be able to drive on any asphalt that has less than 5% of the surface with any flaws, flaws with a diameter no larger than 5cm, and a grade of no more than 3%.

2.2.8 Concrete

Concrete is a material that is made up of a mineral aggregate compound bound together with cement. For this classification we will assume the concrete is the only material present, and will be assume to be bare, which means unpainted or treated. Assuming these restrictions, the vehicle will be able to drive on any asphalt that has less than 5% of the surface with any flaws, flaws with a diameter no larger than 5cm, and a grade of no more than 3%.

2.2.9 Wood

Wood is a structural tissue found in woody plants. For this classification we will assume the wood is the only material present, and will be assume to be bare, which means unpainted, and only treated with a weatherproofing sealer if applicable.

- Fallen wood:
The vehicle will be able to drive over fallen wood, and wood debris such as twigs, sticks, and branches, assuming the fallen wood is no more than 5cm in diameter.
- Cut wood:
The vehicle will be able to drive over any cut wood meant for foot paths or roads that is no more than 2% in grade.

2.2.10 Water

Water is a liquid substance comprised of hydrogen and oxygen. For this classification we will assume any water is free from large debris, and is not mixed with a material to form a highly viscous substance. Any claims made will be under the assumption the water is standing, and the material underneath is solid and without flaw.

- Fresh water:

Fresh water is defined in this context to mean water free from salt. This water can come from natural sources such as rain, or non-natural sources such as a faucet. The vehicle will be able to drive through any fresh water that is less than 15cm in height.

- Salt water:

Salt water is defined in this context as water that has a high concentration of dissolved salt. This water can come from natural sources such as the ocean, or non-natural sources such as a cooling plant. This vehicle will be able to drive through any salt water that is less than 15cm in height. The vehicle will have to be thoroughly washed after driving through salt water to avoid corrosion to the vehicle.

- Gray Water and Black Water:

Grey water is defined in this context as waste water of any kind that does not contain human waste. Black water is defined in this context as waste water of any kind that does contain human waste. This vehicle will be able to drive through either gray and black water that is less than 15cm in height, and does not contain any passengers or have a load with any living organisms.

2.2.11 Colloid

Colloids are a substance that has been dispersed and suspended in another liquid substance. Examples of colloids are: quicksand, marshland, quagmire, and fen.

- Colloids less than 5cm in height:

This vehicle will be able to drive over colloids under 5cm in height.

- Colloids more than 5cm in height:

Depending on the density of the colloid, the vehicle may be able cross short distances over colloids of any depth as long as the vehicle does not stop, however this cannot be accurately tested, and the vehicle is not meant to drive over colloids.

2.3 Definition of Autonomy

Autonomy is an ambiguous term used to define anything capable of making choices, but for our purpose it is used to describe the rapid ability for a computer to make navigational decisions based on data provided by sensors on-board the vehicle. This section will define the components of our system that enable our vehicle to be autonomous.

2.3.1 Geofencing

A geofence is classified as a perimeter made up of coordinates that will act as a boundary and has the ability to allow the triggering of conditions when an object is present inside, or exits the predetermined area. Geofencing will be used in our project to designate a safe operating area for our vehicle, and if it detects its location as outside the acceptable region the computer will immediately power down and the vehicle will come to a stop.

- Requires a minimum of 3 coordinates marked via GPS to create an allowable area of operation
- Requires a minimum area that is 10 times larger than GPS accuracy

2.3.2 Obstacle Detection and Classification

Anytime the vehicle is on, both idle and motion, artificial intelligence will be collecting data from sensors and cameras to determine obstacles that are potential collision hazards. The hazards that are identified will then be categorized based on the potential safety threat that they pose. This will be sensed with cameras and infrared dot projection and analysed with machine learning.

- Wide angle front facing cameras, and sensors, will provide object detection for obstacles in the path
- Safety threats assigned in order of least to most severity: Disregard, Attempt Passage, and Reroute

Listed below are potential hazards that the vehicle could encounter. The vehicle will be able to detect objects at a distance of up to 10m, and hazards will start to be flagged at 5m. This will give the on board AI ample time to make a decision based on the hazard detected. If the hazard is more than 15cm in height, the vehicle will be rerouted, as a hazard of this height is reaching the limit of 20cm that the vehicle can pass over safely. If any solid object is between 5 to 15cm in height and is marked as a hazard, the vehicle will attempt to pass over it at a reduced speed. If any liquids, sand, or silt are marked as a hazard, the vehicle will attempt to cross through it, until the wheels begin to lose steady contact with the ground underneath, and if this occurs the vehicle will attempt to reverse and reroute. If a object of less than 5cm is marked as a hazard, this is simply not flagged and the vehicle will continue on its normal route.

Obstacle	Detection Range	Action
0-5cm (height)	1m	Disregard
5-15cm (height)	2m	Attempt passage, reduce speed

15+cm (height)	5m	Reroute, reduce speed if approaching
Liquid	2m	Attempt Passage, reduce speed
Concrete/Asphalt	5m	Disregard
Sand/Silt	2m	Attempt Passage, reduce speed

Table 2: Rough Obstacles Table

2.3.3 Attention

Human attention is the feature that separates full automation from partial automation, such as lane assist. To keep the occupants safe while using features similar to lane assist, modern vehicles will often require the driver to touch the steering wheel. This can be avoided with an array of sensors and computer vision. The goal of our vehicle is to not require human attention beyond selecting a waypoint list from the on board touchscreen panel and then initiating the trip.

- Low speed allows for the terrain and obstacles to be recorded and the level of danger they pose assessed
- Computer vision will enable the machine to reroute it's path around an obstacle if it's risk of danger is low, or moderate
- Rapidly able to stop the vehicle if the danger level exceeds safe values where avoidance is not possible

2.3.4 Waypoints

Using pathfinding there are many acceptable ways to design a system that safely transports its occupant(s) from the initial to final position. The means of our navigation is called waypointing. This style is based around preselected GPS locations that the vehicle will travel from in a direct path in order of addition.

If an obstacle is detected while the vehicle is traveling along the determined route it will be assessed and categorized whether it can traverse over the obstacle, or to avoid it. Due to the way that waypointing will be implemented, the craft will detour from the direct path in a calculated semi-circular route based on the size of the hindrance, and then rejoin the original navigational track once the obstacle is determined to be avoided.

- Requires at least one waypoint on a map to navigate to

- Must be contained inside the geofence
- Treats each waypoint as an endpoint destination that changes to an origin point before navigating to the next

2.3.5 Automation Level

With the development of more advance technology, automated vehicles are broken down into groups that precisely break down their level of competence. The Society of Automotive Engineers (SAE) developed a widely used system that will classify vehicles based on their skill of driving. The standard is J3016. The figure of the complete SAE levels are shown in the figure below. The levels are simplified here, subsected as:

- Level 0 - No automated control
- Level 1 - Some driver assistance; steering or throttle control
- Level 2 - Some driver assistance; steering and throttle control
- Level 3 - Automation; operator controls vehicle when there is a fault
- Level 4 - Automation in a regional area during set times or conditions; computer controls faults
- Level 5 - Automation anywhere and during all times; computer controls faults

During the research and development of the prototype autonomous vehicle, using the standards that SAE have developed, we have determined our vehicle to be compliant with SAE Level 4. This level was chosen because it aligns with our goal, with minor variances. The deciding factors were that it is operated in a manner according to:

1. An enclosed, or approved, geofence area
2. Computer controls faults of the craft, such as rerouting, or safely parking outside of traffic
3. Passengers are not required to be vehicle operators
4. Controls acceleration and steering simultaneously
5. Self-Routing without drawing direct path
6. Will never have unconditional control of it's navigation

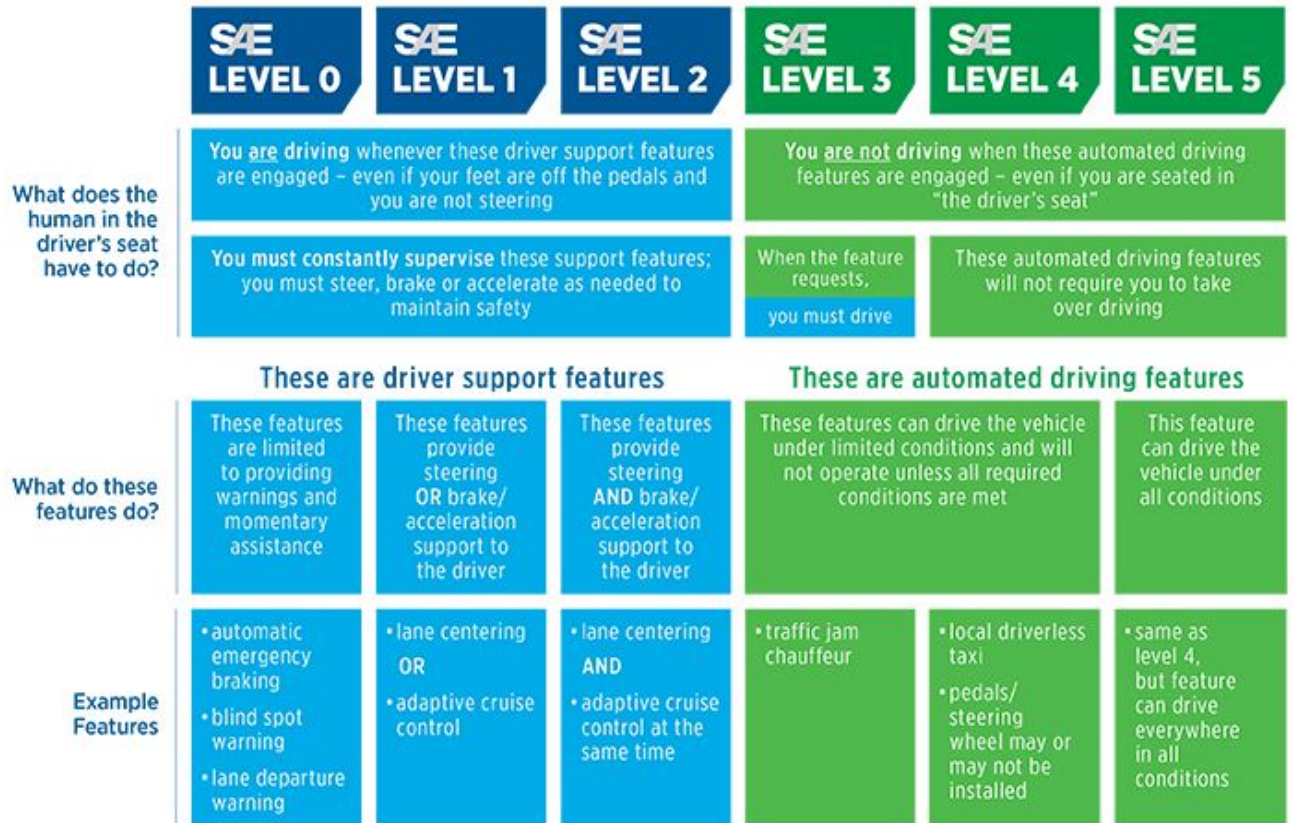


Figure 1: SAE Autonomy Chart [1]

A distinction between the levels of autonomy and their respective conditions can be seen in the graphic above. This is tremendously helpful to quickly illustrate how our vehicle stands apart from similar technologies being deployed in the automotive industry, such as lane assist.

2.4 House of Quality Diagram

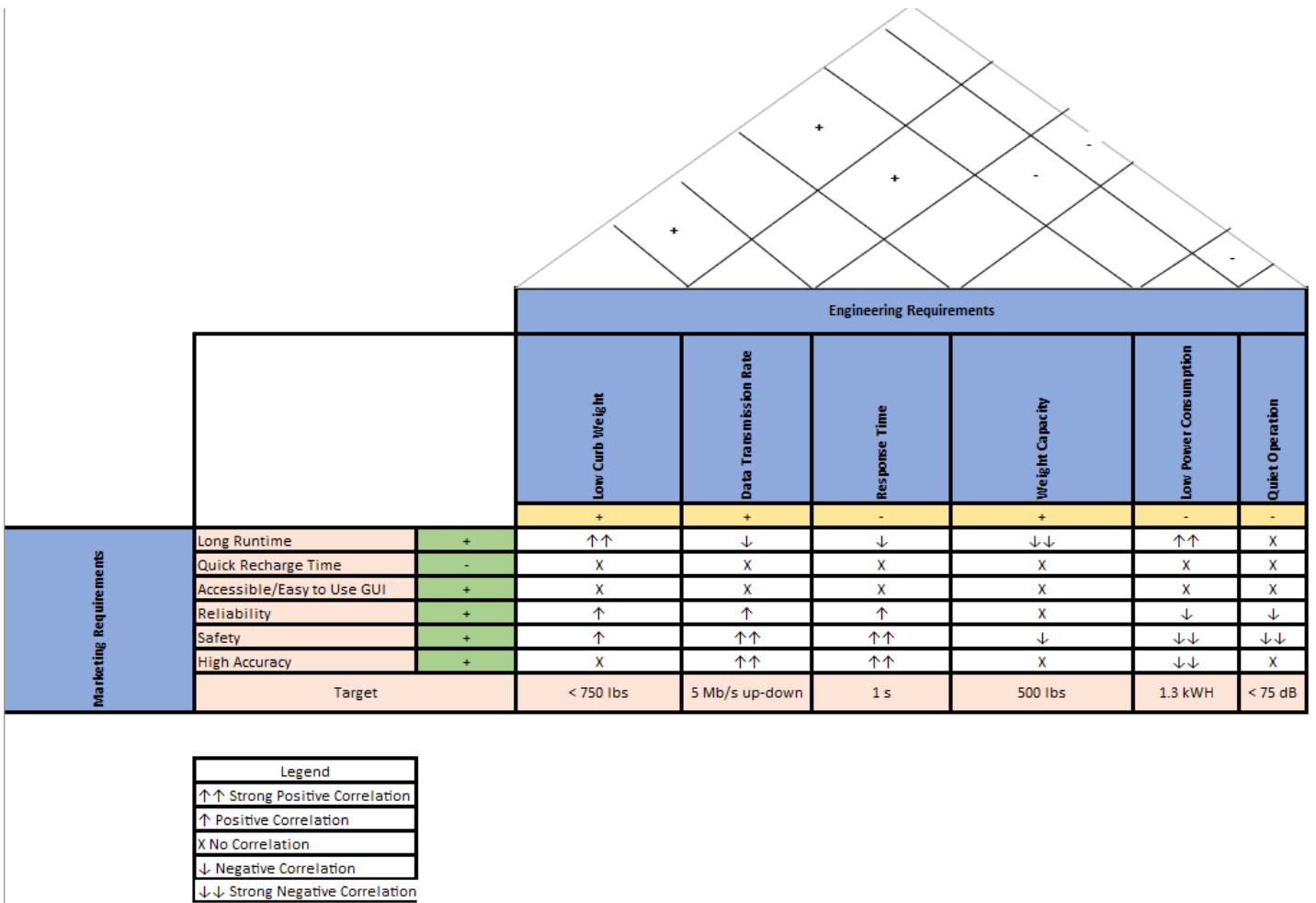


Figure 2: House of Quality Diagram

The House of Quality diagram shows the associations between the engineering requirements and marketing requirements. Our team wants to focus on the qualities of a vehicle that would have the greatest impact on the consumer. Our goal is to make the vehicle as user friendly and reliable as possible without jeopardizing key features necessary to make the vehicle function properly. The marketing requirements our team will prioritize are: long runtime, quick recharge time, an accessible and easy to use GUI, reliability, safety, and high accuracy.

2.4.1 Long Runtime

The first marketing requirement our team will focus on is long runtime. This is an important feature for a vehicle because you want to be able to travel long distances. Since our vehicle is meant for exploration and data collecting, having a long runtime will allow for the vehicle to collect more data over a large area. In order to maximize runtime, we will need to make sure our vehicle doesn't have a lot of weight, and have a low power consumption. If our vehicle weighed a lot, it would cause the motors to have to work harder to move, which would drain the batteries extremely quickly.

We also have to make sure our electronics are efficient. If our electronics draw

a lot of power constantly, it would also cause the batteries to drain quickly. Our team will also have to lower the total weight capacity of the passengers and cargo in order to maximize runtime. We will have to do extensive testing to make sure the consumer will be satisfied with the total weight capacity while still having the longest total runtime possible.

2.4.2 Quick Recharge Time

The second marketing requirement our team will focus on is quick recharge time. This requirement is completely reliant on how our team chooses to supply power to the vehicle. A quick recharge time is a feature any electric vehicle should invest in. The vehicle's users will rely on a quick recharge time to be able to use the vehicle multiple times throughout the day. It would not be a good design decision to have our vehicle take a long time to charge, such that the users cannot use the vehicle whenever they wanted to.

Our team also needs to think about recharging away from the main charger. Currently, our team is choosing to recharge the batteries with solar panels and a 12V battery charger. Having solar panels to recharge the vehicle constantly will help the vehicle maintain a charge while away from the charger. This will also give the vehicle another means of charging, rather than relying completely on a charger.

2.4.3 Accessible and Easy To Use GUI

The third marketing requirement our team will focus on is an accessible and easy to use GUI. Our team needs to design the GUI in a way that makes it possible to use while in motion. Response time will play a big part in the usability of the GUI. If the GUI takes too long to update the user may have to stop the vehicle and wait for the screen to update. Since we want our vehicle to have some autonomy, and have the capability of self driving, it would be a bad design choice to force the user to not use the autonomous functions of the vehicle while they wait for the screen to update.

Our team also has to create the GUI in a way that is optimized for driving conditions. Qualities such as large touch areas and high quality graphics are key to making this possible. The screen will be positioned in the center of the dashboard, which is far away from the passengers. To make the screen usable our GUI will have to account for the distance, and be as clear as possible. It will also have to have touch areas that are suited for a moving environment. We don't want the users to accidentally press the wrong button and go through the trouble of navigating back to the previous screen, or change a setting they didn't intend to. To make this possible we will have to position the buttons away from each other, and make them large enough to where the user does not have to be precise in their actions.

2.4.4 Reliability

The fourth marketing requirement our team will focus on is reliability. It's important to have a vehicle that people want to use because it makes their tasks easier to complete. Our team is aiming to create a vehicle that will be useful to the user

while showing that it is a tool they can repeatedly turn to for their needs. To meet this requirement, the vehicle should have features that make it efficient. A low curb weight will make the vehicle less likely to run into trouble while moving across terrain. The vehicle needs to be able to transport the passengers wherever they need to go for their work.

The vehicle also needs to be able to reliably transmit data. If the data cannot be collected fast enough to meet the user's needs, it may cause the user to have to go back and spend more time waiting for the data to be collected, which would impact the work the user is doing. Likewise, we want the vehicle to be responsive to any input the user gives to it. Our team does not want the vehicle to lag in any way, doing so would cause the user to have to spend more time waiting on the vehicle, which negatively impacts the user's trust in the vehicle. To accomplish this our team will have to manage the power the vehicle consumes so the vehicle can have powerful electronics while also keeping the other marketing requirements achievable.

2.4.5 Safety

The fifth marketing requirement our team will focus on is safety. Our team must prioritize safety over all other requirements. Most of the safety features that will be implemented into the vehicle will come from the technology used for autonomy. This includes the ultrasonic sensors for determining the distance from objects, and the camera used for object recognition. These two features will work together to make sure the vehicle doesn't crash into any objects. The ultrasonic sensors will be placed on each side of the vehicle, and will be able to sense if anything comes within a certain range of the vehicle, if this happens actions will be taken to either correct the vehicle's path to prevent hitting the object, or moving out of the way of an object about to hit the vehicle. The camera will be placed on the front of the vehicle, which will be used to identify all objects of a certain size that could potentially harm the vehicle or vice versa, and take appropriate action to either stop the vehicle or move the vehicle away from the object. Having a quick response time and fast data transmission is also important for safety. If our vehicle is unable to respond fast enough it could cause an accident even if the software and AI are working properly.

Having a safe vehicle comes with some challenges our team will need to minimize. These include a larger power consumption, and a slower operation. These two things directly impact some of our other marketing requirements, and will have to be carefully balanced in order to create a vehicle that is safe and efficient. Optimized software will be important for balancing these two requirements. Likewise, having a power system capable of handling the software and hardware will allow our team to do more to protect the vehicle and its passengers.

2.4.6 High Accuracy

The final marketing requirement our team will focus on is high accuracy. Since our project is based around collecting a lot of data of the environment, our vehicle needs to provide accurate results. For scientific use, the data we're collecting may need to have a high degree of accuracy to provide useful information. Having

inaccurate data, even by a couple decimal values, could invalidate all of the collected data if it's being used for research. It's also important that our vehicle has accurate data for its autonomy purposes. As the previous section discussed, safety is a high priority for our team. It would not be safe for our ultrasonic sensor to be inaccurate, or it may give false readings of nearby objects. This could cause a crash, and harm the passengers or pedestrians around the vehicle. Our vehicle will need a reliable and fast way of data transmission and a fast response time to make all of this possible.

By making all of this possible, power will be sacrificed to give the electronics the speed they need to be as fast as possible. Our team will have to find a way to maximize our power in our electronic heavy vehicle. Solar panels and efficient batteries are one way to accomplish this task, however it may not be enough to satisfy the other marketing requirements. These are challenges our team will work towards overcoming so our vehicle can meet all of the requirements specified.

3 Block Diagram

The block diagram shown below shows the current information our team has gathered and compiled into an easy to read diagram. The individual responsibilities of each group member are color coded, with their respective color filling in the node they are responsible for. Node with a border color represent if another group member will be assisting with the node, in which case the border will have the corresponding color of the group member assisting.

Devon Wilkerson will be working on the computer vision and AI. This will allow our vehicle to see any external stimuli, as well as react to it. This system will prevent the vehicle from crashing into objects in the path of the vehicle, and will also be used to judge whether or not the vehicle can safely proceed forward over certain terrain. Kris Choudhury will be designing the NFC application that will lock/unlock the vehicle, and design the GUI that will link a local web server with the microcontroller which will be displayed locally on a front-end application. Benjamin Goerdts will be working on linking the motor controller to the microcontroller, an electric steering system, and implementing parts onto the chassis, which entitles control via software. Sergio Gonzalez will be working on the embedded systems. He will also be selecting the sensors, and implementing each one into the vehicle through the microcontroller.

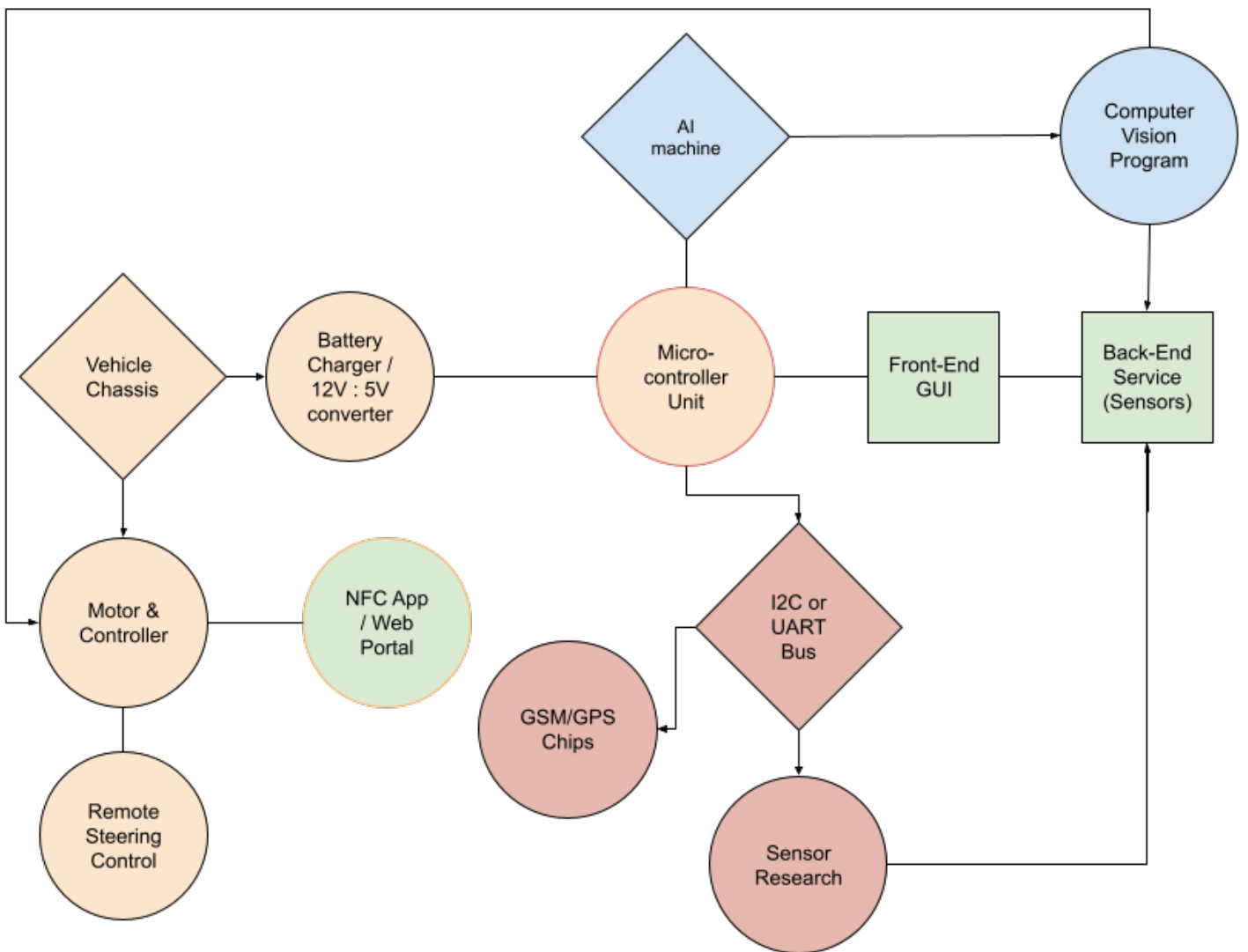
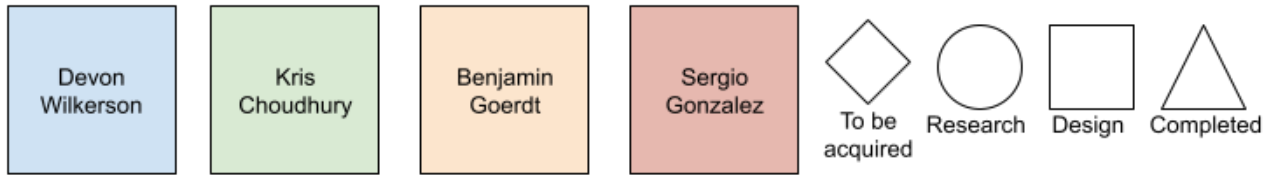


Figure 3: Block Diagram

4 Budget

The following parts list shows the components required for our project. Although all components are essential to the design, the most important components are the vehicle chassis and the AI computing device. The chassis is the main component that we are building around, as it provides us with a rigid frame to add smart features to. The AI computing device will act as the brains of our vehicle as it will be primarily used for computer vision.

The cost of our design has not been fully accounted for yet mainly due to parts, such as the vehicle, that do not have set prices and are dependent on the seller and condition of the component. Alternatively we have the option to build the vehicle ourselves which may prove to be cheaper in the long run. We are also in the process of finding sensors that fit our performance needs, and getting a complete bill of materials. The goal is to minimize overall costs but ensure quality components that meet our specifications.

Our team has also considered retrofitting a Power Wheels toy car instead of a golf cart or hand-built vehicle. The price of it is far cheaper than the former two options and more convenient as it already functioning out of the box. However, due to its size it can only be used to demonstrate as a prototype for our design. Additionally due to the size limit of these toy cars, we would have to change our weight constraint in Requirements Specification section. Despite the limitations, this is a good option when considering all the benefits.

As an estimate, our team has found golf carts to be around \$1200, go-karts to be around \$500, while a hand made frame could be built for around \$300 using wood or metal. A Power Wheels car would be cheaper ranging from as little as \$200. Sensors range from a couple dollars to multiple hundreds of dollars. As our design cycle progresses we'll be able to figure out which sensors to prioritize and spend more money on, and which ones we can afford to replace with a budget oriented option.

Item	Description	#	Price (Ea)
Vehicle	Main system	1	\$200-1000
Batteries	Batteries to power motor	4	\$100
Motor/Controller	Component for movement	1	\$500
Motor/Controller	Enables control of wheels	1	\$40
Solar Panel	Alternative power		Price
Touchscreen Display	User interaction	1	\$150

AI Computing Device	Computer vision processing	1	\$200
NFC Sensor	Near-field Communication	1	Price
Communication Bus	Data transfer between components	1	Price
GPS Module	Global Positioning Component	1	\$13.10
GSM Module	GSM Component	1	\$86.99
Temperature Sensor	Temperature Sensing Component	1	\$3.95
Humidity Sensor	Humidity Sensing Component	1	\$3.95
Oxygen Sensor	Oxygen Sensing Component	1	\$53.90
Carbon Dioxide Sensor	Carbon Dioxide Sensing Component	1	\$37.99
Ultraviolet Sensor	UV Sensing Component	1	\$5.90
Particulate Matter Sensor	Particulate Matter Sensing Component	1	\$39.95
Stoplight Switch	Momentary Switch Component	1	\$5
Volatile Organic Compounds and Ozone Sensor	Atmospheric Sensing Component	1	\$14.95
Light sensor	Light Sensing Component	1	\$4.95
Ultrasonic Proximity sensor	Range Finding Component	4	\$7.95
Kinect Camera	Camera for Computer Vision	1	\$0
LED Headlight	Front Lights	2	\$20

LED Taillight	Rear Lights	2	\$20
Relays	High Current Controllers	15	\$2
Inline Fuse Holders	Overcurrent protection	5	\$5
Voltage Converter	DC:DC Converter	3	\$5

Table 3: Item List Table

4.1 Bill of Materials

The bill of materials below is all of the components our team has been able to find up to this point. Some parts are still begin researched so we can find the best price, and these parts are not included in the bill of materials yet. We will continue buying parts as they come available, and when our team has extra money to buy more expensive parts. Some parts, such as the Power Wheels, may be excluded from the list since we are currently finding the best option for purchasing them. Some parts do not have a part number, and a website link will be added in place of the part number.

Manufacturer	Part No.	Description	Quantity	Price	Quantity on Hand
NVIDIA	945-13541-0000-000	Jetson Nano System on Module - SOM Development Kit 4GB RAM Linux	1	99.00	5
CanaKit	PI4-4GB-STR32F-C4-BLK	Raspberry Pi 4 - Starter Kit 4 GB Ram + 32 GB of Storage	1	99.95	1000+
XYGStudy	SIM7600G-H 4G HAT	GSM Communication Module 4G 3G 2G for Raspberry Pi	1	86.99	10

Arduino	A000066	Arduino Uno Rev 3 based on ATMEGA328p	1	23.00	1000+
Seeed Studio	109020022	Grove Air530 GPS Module	1	9.88	126
Raspberry Pi	LCD-13733	Raspberry Pi Foundation 7" Touchscreen LCD Display	1	60.00	1000+
Seeed Studio	113020006	Grove NFC Sensor	1	23.70	10+
Mighty Max	YTX12-BS10	Spare Aftermarket Battery	2	26.99	99+
Xinpuguang	100W	Solar Panel & Charge Controller	1	80	268
Traxxas	RC775	High Power Drive Motor	1	40.95	199+
Firgelli	LA35	Linear Actuator	1	110	499+
Best Choice Products	12V-BL	Power Wheels Vehicle	1	189.99	20+
Adafruit	VEML7700	Light Sensor	1	4.95	761
Adafruit	VEML6070	UV light sensor	1	5.95	226
Adafruit	AM2320	Temperatur and Humidity Sensor	1	3.95	304
DFRobot	SEN0322	Oxygen Sensor	1	53.90	68
Amphenol Advanced Sensor	T6703-5k	Carbon Dioxide Sensor	1	37.99	95
Adafruit	PMSA003I	Particulate Matter Sensor	1	37.99	6

Adafruit	SGP40	Volatile Organic Compounds Sensor	1	14.95	N/A
Adafruit	HC-SR04	Proximity Sensor	4	7.94	11

Table 4: Bill of Materials Table

5 Project Milestones

The major milestones have been noted in the table below. As the design cycle progresses a more detailed timeline will be possible for the Senior Design 1 section. We do not currently know the exact path we're going to take regarding a lot of choices. As we have more discussions and come to conclusions about how we're going to design our project, the Senior Design 1 section will become more complete. The Senior Design 2 section is currently unknown regarding milestones, and only base level filler milestones are currently listed. Once we get closer to the start of Senior Design 2 we'll be able to more accurately fill in that section.

SENIOR DESIGN 1

Task:	Assigned to:	Start Date:	End Date:	Status:
Assign Group Roles	Group	05/27/2021	06/15/2021	Complete
Component Research	Group	05/27/2021	06/15/2021	Complete
Initial Project Document	Group	05/27/2021	06/15/2021	Complete
D&C 2 Document	Group	06/15/2021	06/25/2021	Complete
Start buying components	Group	06/15/2021	08/03/2021	Complete
Vehicle Decision	Group	06/15/2021	08/03/2021	Complete
Microcontroller Design	Ben	06/15/2021	08/03/2021	Working
Computer Vision Design	Devon	06/15/2021	08/03/2021	Working

Sensor Implementation	Sergio	06/15/2021	08/03/2021	Working
GUI interface	Kris	06/15/2021	08/03/2021	Working
NFC App	Kris	05/27/2021	08/03/2021	Working
60 Page Draft Document	Group	05/27/2021	07/09/2021	Complete
100 Page Draft Document	Group	05/27/2021	07/23/2021	Complete
120 Page Final Document	Group	05/27/2021	08/03/2021	Complete
Have most components purchased	Group	06/15/2021	08/03/2021	Complete

Table 5: Project Milestones Table (Senior Design 1)

SENIOR DESIGN 2

Task:	Assigned to:	Start Date:	End Date:	Status:
Prototype	Group	TBD	TBD	Future
Testing	Group	TBD	TBD	Future
Final Build	Group	TBD	TBD	Future
Final Report	Group	TBD	TBD	Future
Final Presentation	Group	TBD	TBD	Future

Table 6: Project Milestones Table (Senior Design 2)

6 Related Research

6.1 Similar Devices

The initial idea for our design led us to researching autonomous vehicles that already existed. Our team wanted to find ways we could improve or differentiate our project from these designs. The concepts stated in the project narrative section lists vehicles that promoted full autonomy such as The Curiosity Rover, The Infosys' Driverless Golf Cart, and Yamaha's Driverless Buggy for Public Transport. Other examples of autonomous vehicles includes University of Central Florida's Autonomous Shuttle, and our research extends further than land-based vehicles, which includes aircrafts and watercraft, such as drones and submarines.

The biggest attribute with our design that sets our project apart from these examples is that our vehicle is not explicitly an autonomous vehicle. The vehicle will be designed as a surveying tool with a variety of sensors on board. Because of this, we're also researching other data-collecting vessels as well. Military UAVs are an example of this, which constantly scan and capture high resolution images.

6.1.1 Curiosity Rover

One of NASA's largest and most capable rover, Curiosity's goal was to traverse the surface of Mars collecting soil, rock, and air samples to be processed for on board analysis. The purpose of this was to identify whether Mars could be capable of supporting microbial life.



Figure 4: Curiosity Rover - Free Use Photo

Since the terrain on Mars is highly irregular, the scientists at NASA needed to make sure that it would be able to navigate across through terrain safely. The scientists used autonomous navigation to circumvent this, by using the camera it

would analyze the images it would take and calculate a safe driving path to take in real time. Our design would be heavily inspired by Curiosity, as we wanted to build an autonomous vehicle that would be able to traverse different terrains and environments, in which sensors were utilized to gather information on the surroundings.

6.1.2 Infosys' Driverless Golf Cart

The autonomous buggy design from Infosys demonstrates a vehicle meant to be used for civilian use in metropolitan areas. The main goal of the design is to navigate a controlled campus environment to safely drive civilians to their destination and utilize redundant mappings to ensure civilian safety.

The main technologies used are 3D LiDAR, as well as GPS and RADAR sensors to help power the autonomy of the design. Our design would draw some influence from Infosys' as we would also want the buggy to be able carry passengers and utilize sensors such as GPS and radar. The GPS sensor will be used to map out the surrounding area and display on a GUI. For the radar sensor, our goal is to use something similar, the UltraSonic sensor, to help determine nearby obstacles that could impede movement or even damage the vehicle.

6.1.3 Yamaha's Driverless Buggy

Yamaha's design is quite similar to the Infosys product, as its main usage is meant for civilian transportation. One of its main features that intrigued us was the mobile app. According to the information on Yamaha's website on this design. The vehicle would be able to communicate with a mobile application and users would be able to reliably navigate to said civilian's area. Although the technology for navigation isn't listed on the website, we have other sources we can base our system on for our project.

The mobile application that we want to design for our system would be different from Yamaha's. Instead of auto navigation directly to the user, we decided that we would rather create a mobile app that would be used to "unlock" the vehicle. Using Near-Field Communication (NFC), we would like to have the mobile app send a signal to an NFC sensor on the vehicle where a user could turn the vehicle off and on at their discretion.

6.1.4 University of Central Florida's Autonomous Shuttle

In an effort to reduce car accidents and promote the benefits of AVs (Autonomous Vehicles), The Federal Department of Transportation (FDOT) partnered with The University of Central Florida (UCF) to help combat these issues as Orlando is a rapidly growing region that would be ideal to test these technologies. Although not much documentation is present on the driverless shuttle operating at UCF, it appears to follow along a set path utilizing GPS for this and use sensor technologies to analyze obstacles and civilians for quick rerouting.



Figure 5: UCF Autonomous Shuttle - Reprinted with permission from –

6.1.5 Military UAVs

Although we have discussed only land-based vehicles, we also drew inspiration from other types of vehicles such as aircrafts and submarines. The one vehicle that interested us the most was a military unmanned aerial vehicle (UAV). What caught our attention about the vehicle was the technologies it used. Like the previously discussed designs, it utilizes GPS for self-navigation on a predetermined course. Other technologies used are ultrasonic sensors, computer vision, and autonomous recharging. These three aspects will be used in our design where the ultrasonic sensors will be used for proximity detection for obstacles. Computer vision will be the main technology powering our autonomy. As for autonomous charging, we wanted to include a solar panel that would help our design travel for longer runs without having to worry about energy problems.

7 Vehicle

This section is about the discussion of the vehicle part of our project. It discusses the options we have for vehicles, the comparison of the different options, and the vehicle our team chose to implement into our project. Choosing a vehicle is an important step in our design step, if our team doesn't consider all of the possible drawbacks to each option, it could compromise the project goals.

The first option we have is retrofitting an existing vehicle. The two vehicles we chose for consideration are a golf cart and an off road go-kart. These vehicles were chosen because they already have off-road capabilities making them ideal for the exploration aspect of our project goals, and they are also light weight enough to easily transport to testing locations. They are also small enough to not seriously

injure any bystanders or passengers if an accident occurs.

The second option we have is building a vehicle from wood or metal. This option was chosen for consideration due to the readily available materials and customization. Our project has a limited time frame for completion, so our team needs to be flexible when choosing parts for our project. A retrofitted vehicle would be more convenient to use, however it's not an option that's guaranteed to be available when we're ready to purchase the vehicle. We need to consider building a vehicle as an option if purchasing a vehicle is no longer an option due to time constraints.

The final option we have is buying a Power Wheels toy. This option was chosen for consideration due to the price and small size. Our team is completely self funded, so we need to cut costs wherever possible. This option is the cheapest one being considered, and would allow us to spend more money on other parts if needed. We also need to take into account that some parts may break during testing and we may need to buy spare parts, which wouldn't be possible if we used our entire budget. This option is also the smallest one being considered. Our team needs to prioritize safety during this project, and using a larger vehicle for the project comes with inherit risk to our safety, and the safety of any bystanders. Power Wheels are extremely small and while this option would jeopardize other aspects of our project, it might be worth reevaluating our project goals in favor of safety, which would be done by re-branding our project as a proof of concept instead of a working prototype.

7.1 Retrofitting

The first consideration for our vehicle is retrofitting an existing vehicle. Retrofitting will provide us with a fully functioning vehicle that we can retrofit to our needs. This would greatly reduce the work needed to create a functioning vehicle, since we wouldn't have to design a completely new chassis. This method allows us to work on the important parts of our project, and optimize the electrical components for better function. The downside to this method is the cost.

As stated in the budget section, a golf cart is the most expensive option we have. We have searched extensively, and all the golf cart we have found sell in the range of \$1200 used, and up to \$15,000 new. This is far out of our budget of \$200 - \$300 for a vehicle. The only way this would be a viable option is if we get lucky and find a golf cart in our range. Cost aside, a golf cart would give us access to batteries, and a roof to attach solar panels on. Golf carts are also fully electric, which is beneficial to our team since that system will already be implemented on the vehicle.

A less expensive option that still allows us to focus on the important parts of our project is a go-kart. A go-kart would give us a similar benefit as the golf cart, and provide us with a fully functional vehicle. A go-kart is also more budget friendly, albeit still slightly out of our budget range, at around \$500 used. This would be a viable option if we can negotiate the price down to a more reasonable amount, which is a common occurrence with used vehicles. The downside to this option is that go-karts are mostly gas powered, and we would need to design an entire electric power system. In the long term this would cause more work, and the possibility of causing more errors, for our team.

Moderate	Low	Chance of Failure
1-2 weeks	1-2 Days	Time Investment
Moderately Reliable	Very Reliable	Reliability
1,000.00	1,000.00	Possible Repair Cost (\$)
1,200.00	500.00	Cost (\$)
Retrofit - Golf cart	Retrofit - Go-kart	

Table 7: Retrofitting Comparison Factors

7.2 Hand-built

The second consideration for our vehicle is building a new one from wood or metal. Building our vehicle would allow us to completely customize the shape and function of our chassis and main systems. This would create more work in the long term, but will give us more control over the final product. Having complete control over our vehicle's design means we will be able to choose where we can put our sensors, motors, batteries, etc, without the need of cutting or adding parts which would be necessary with an existing chassis.

High	Chance of Failure
1-2 Months	Time Investment
Not Reliable	Reliability
900.00	Possible Repair Cost (\$)
300.00	Cost (\$)
Hand-built	

Table 8: Hand-Built Vehicle Comparison Factors

The cost of this option is also more acceptable for our budget when compared with retrofitting. While it is on the higher end of our budget, this option will

always be available, since we only need to go to a hardware store to get any part we would need for the chassis. While the chassis itself would be cheap to make, we would also have to consider the other parts that go into making a vehicle. If we don't have the vehicle systems already implemented into the vehicle, such as a steering system, suspension system, etc, the cost of this option comes into question. It may be possible to get the vehicle systems a junkyard, which could be more cost friendly, but unlike building a chassis, getting the vehicle systems from a junkyard won't be always available.

The biggest concern of this option is part failure. This team is made up completely of electrical/computer engineers, so we don't have the experience that a mechanical engineer would when it comes to building a structurally stable vehicle chassis. Because of this, our chassis would have a high chance of failure if we built it by hand. If any part of our chassis failed during testing, this would drive the cost up, possibly higher than a go-kart. With all of these considerations, this is a last resort option, and will only be considered if none of the other options can be chosen.

7.3 Power Wheels

The third consideration for our vehicle is buying a Power Wheels, or other off-brand ride-on toy. Buying a Power Wheels, like the retrofitting option, would provide us with an entire vehicle that we can use. This vehicle does come with some trade-offs, however. For starters, it wouldn't be a fully functioning vehicle. Power Wheels vehicles are essentially a molded plastic chassis fitted with two motors that power the back wheels. Using this option would mean sacrificing reliability, and scalability in our project.

Moderate	Chance of Failure
1-2 Days	Time Investment
Not Reliable	Reliability
150.00	Possible Repair Cost (\$)
150.00	Cost (\$)
Power Wheels	

Table 9: Power Wheels Comparison Factors

Since Power Wheels toys aren't an actual vehicle, it would have limitations when it comes to traversing the terrain we'd require. A portion of the initial goal of this project is to create a vehicle that is capable of driving over multiple different

types of material. A Power Wheels toy simply will not be able to accomplish this. Likewise, a Power Wheels toy isn't constructed like an actual vehicle. It's lacking all of the fundamental systems that would be needed to consider it an actual vehicle. This means it will have to be a proof of concept, rather than a functional prototype. Taking this option would come with some consequences, including providing a scalable product, but it is the most budget friendly option.

The cost of a Power Wheels toy is one of the biggest benefits for consideration of this option. A new Power Wheels sells for around \$150 to \$250, depending on the model. A used Power Wheels would sell for less than \$100. The cost is within the lower end of our budget, and unlike the hand-built vehicle, there wouldn't be side costs that would be needed. If a part of the Power Wheels fails at any point, replacing them would be cheap. Power Wheels also has in depth troubleshooting guides, as well as authorized parts sellers, and non-functional Power Wheels can be bought second hand for parts. With access to cheap, and readily available, parts we wouldn't have to worry about going over our budget or not being able to get a part we need if one fails.

7.4 Comparison

Comparing the different options, each one has their own benefits and downsides. A table comparison has been included in the respective sections. There are many factors we must consider before choosing an option, such as chance of failure, time investment, reliability, possible repair costs, and cost to buy the initial parts. A comparison of these factors will be discussed below based on the tables above.

7.4.1 Chance of Failure

The scale in the chance of failure section in the table above shows which options have the likelihood of having a critical failure, and the scale is as follows: None(no chance of failure), Low(1%-25%), Moderate(26%-50%), High(51%-75%), Severe(76%-100%).

The lowest chance of failure option is the go-kart. This is due to the off-road environments go-karts are meant to traverse. Go-karts are built to withstand harsh impacts, so having one structurally fail is unlikely to happen.

The highest chance of failure option is the hand-built vehicle. As discussed in the hand-built section, our team does not have anyone with experience making a physical structure, therefore the chance of it structurally failing is high.

The other two options, the golf cart and Power Wheels, are in the middle of the other options. While the golf cart is meant for off-road driving, it's not built for high impacts like the go-kart. Likewise, the Power Wheels can go off road to some extent, but wasn't built for it, and the construction of the wheels means there's no suspension to absorb high impacts, leading to the motors having the chance of breaking. The chassis is also made of plastic, which is prone to breakage.

7.4.2 Time investment

The time investment section in the table above shows which options will take the most time for repairs and the possibility of rebuilding.

The lowest time investments are the go-kart and Power Wheels. The go-kart would be bought used, and will most likely need some repairs. Some of our team members have experience repairing vehicles, so any minor repairs shouldn't take more than a couple days. As we saw in the last section, the possibility of the go-kart failing is low, so that doesn't factor into the time investment. The Power Wheels would come new out of the box, and wouldn't need to be built in any way. This wouldn't need any repairs to start, and the simplicity of the build would make any future repairs quick, and we would only have to consider the time it takes to get parts shipped.

The golf cart, like the go-kart, would be bought used, and any repairs should be just as fast as the go-kart. The main concern with the golf cart is failure. The more repairs needed to keep the gold cart functional is more time needed to invest in repairing it, which could add up to weeks.

The hand-built vehicle has the largest time investment, at 1-2 months. This is due to the time it would take to build the chassis, and like the golf cart, the chance of multiple repairs due to chance of failure.

7.4.3 Reliability

The reliability section in the table above shows which options will be able to successfully traverse the environment, and the scale is as follows: Not reliable(0%-15%), Reliable(16%-45%), Moderately Reliable(46%-70%), Very Reliable(71%-100%).

The lowest reliable options are the hand-built and Power Wheels. The hand-built option will most likely not be structurally stable enough to traverse rougher terrain, such as rocks, and large gravel. It may also not hold up to repeated traversal in water, since a hand-built vehicle wouldn't be completely waterproof. The Power Wheels would also have trouble traversing rougher terrain, since the motors have no shock absorbency and would have a high chance of failure after repeated impacts. The motors are not water proof either, which would prevent the vehicle from going through any water that has the chance of reaching the motors. Due to the size of the Power Wheels, it would not be able to traverse larger objects, and would only be capable of serving as a demonstration of the autonomy of the vehicle on flat land.

The golf cart is classified as moderately reliable. Since golf carts are meant for off road use to some degree, they're built well enough to traverse rough terrain. The issue with the golf cart is with the shape. The chassis is built in such a way that leaves it with an extremely low ground clearance. This would prevent the vehicle from passing through deep water. The golf cart would have an easier time passing through rough terrain than the hand-built or Power Wheels options, however it would still have the issue of not being able to pass over large objects.

The go-kart is classified as very reliable. Go-karts are built to go off road, and as such would have no issues with clearance, or shock absorbency. Since the go-kart was designed for the purpose we need, it should be able to handle anything we need it to do, and possibly more since the optimal operating conditions are often below the max threshold the vehicle can operate under.

7.4.4 Possible Repair Cost

The possible repair cost section in the table above shows the possible cost of repair for each option.

The lowest possible repair cost is the Power Wheels at \$150. Since the Power Wheels option is mainly plastic, most damage done to the chassis wouldn't need to be repaired. The only damage worth considering repair would be the motors or batteries. The worst possible damage would be a total failure of all parts, in which case the repair cost would be the cost of replacing the entire vehicle and buying a new one.

The next highest possible repair cost is the hand-built option at \$500. Many things could go wrong throughout the lifetime of the project, and if we decide to build the vehicle from scratch, this would lead to a lot of repair costs. Like the Power Wheels, if the worst outcome happens and the entire vehicle fails, we would buy an entirely new set of parts and build a new vehicle. The main issue come from repeated failure. Anything can go wrong when under a time constraint, and mistakes will happen. If multiple mistakes happen over the course of multiple different iterations of a build, the repair costs will continue to add up.

The two options with the highest possible repair costs are the golf cart and go-kart. As stated in the previous sections, if our team chooses the golf cart option we'll be using the golf cart outside of its intended purpose, and the likelihood of something breaking is without question. The golf cart and go-kart alike would also come used, and would likely needs repairing before they're fully operational.

7.4.5 Cost

The cost section in the table above shows the predicted initial cost of buying the vehicle for each option. As with every part of this project, cost is an important factor when determining which vehicle to choose. Our team has a limited budget, and we will not be able to meet the goals of this project without managing every dollar we spend, and cutting costs where able.

The lowest cost option is the Power Wheels at \$150. The low cost of this option isn't without drawbacks. While it is the most budget friendly option out of all of the available options, it is also the most restricting of the options. Our team wouldn't be able to fully demonstrate the capabilities of a vehicle at this scale, and it would severely limit the features we would be able to implement.

The next lowest cost option is the hand-built vehicle at \$300. Like the Power Wheels, the low cost of this option has multiple drawbacks. Construction quality would suffer since our team doesn't have the experience of building a vehicle, and we wouldn't be able to build the vehicle to drive on all the terrain we would like. While it is double the cost of the Power Wheels, our team isn't confident it would be better to invest in, rather than just buying multiple Power Wheels.

The next option is the go-kart at \$500. As discussed in the previous sections, the go-kart is by far the most reliable option to choose, and the cost isn't too far out of our budget. The main issue with this option is the conversion from a gasoline powered engine to an electric engine. While the initial cost isn't too high, the conversion costs might make this option far above the budget our team has.

The most expensive cost option is the golf cart at \$1,200. After the go-kart, this option would be the next best option to choose for reliability. This option

also comes with an electric powered engine, which would possibly make it cheaper in the long term than the go-kart. The initial cost, however, is far outside of our budget. The high cost makes it hard to consider this option, but it's possible to get a golf cart below the projected cost due to damage, or if the golf cart is no longer operational.

7.5 Final Decision

After considering all of the available options our team will use the Power Wheels vehicle in our project. The chance of failure is within a reasonable range to where our team is confident in the ability of the Power Wheels. Time is a big factor in this project, since we have a limit amount of time to complete the project. Because of this we need to be able to fix any broken parts fast. If the Power Wheels were to fail, our team will quickly be able to fix any parts that failed and continue to progress with the project. Parts are also widely available for the Power Wheels, and we would not have to worry about parts being out of stock or discontinued. While the reliability is the lowest out of all the options, this is only due to the size of the Power Wheels. As a scaled representation of what our vehicle could be capable of, the Power Wheels will be enough to provide a proof of concept for our project. This option is also the most budget friendly. Our team has a limit budget, and choosing the Power Wheels gives us extra money to spend on other components that will provide a better chance of our project meeting the requirements.

The biggest factor for choosing the Power Wheels that was not mentioned in the comparison is safety. This project is a learning experience for all of the members of our team, and we cannot risk our project having a catastrophic failure and one of our team members or someone from outside of our team getting injured. When making a vehicle safety has to be a priority, and implementing autonomy gives a greater chance of something going wrong. We have to prioritize the safety of anyone around the vehicle over making a project that is fully functional. This decision will impact the final design of the vehicle, and we will have to redefine the requirements of our project, but this choice will be for the best interest of the team and the University.

8 Serial Communication

This section will discuss the different types of serial communication that are most popularly used. Serial communication is vital to our project due to the amount of sensor information we must transfer from the sensors to the center display on the vehicle.

Serial communication means to stream bits of data from one device to another one bit at a time. Over the years there have been many methods of serial communication introduced, with the main three among them being I2C, SPI, UART. Our vehicle will come equipped with multiple different kinds of sensors and these sensors range in their method of communication, so understanding how these protocols work is critical to the success of our vehicle.

8.1 UART

Short for Universal Asynchronous Receiver/Transmitter, UART is a form of serial communication between two devices. A quick way to find out if a device communicates using UART is to look for the Tx (Transmitter) and the Rx (receiver) pins. There are a few large differences between UART and SPI or I2C, one of which being that it is not a communication protocol but instead its own Integrated Circuit (IC). The UART IC takes in parallel bits, multiple bits at the same time, and transmits the bits serially to the other UART device. When receiving, UART does the opposite; it takes in the serial data and is converted to parallel data. The figure below illustrates the process of converting parallel data to serial data then back to parallel.

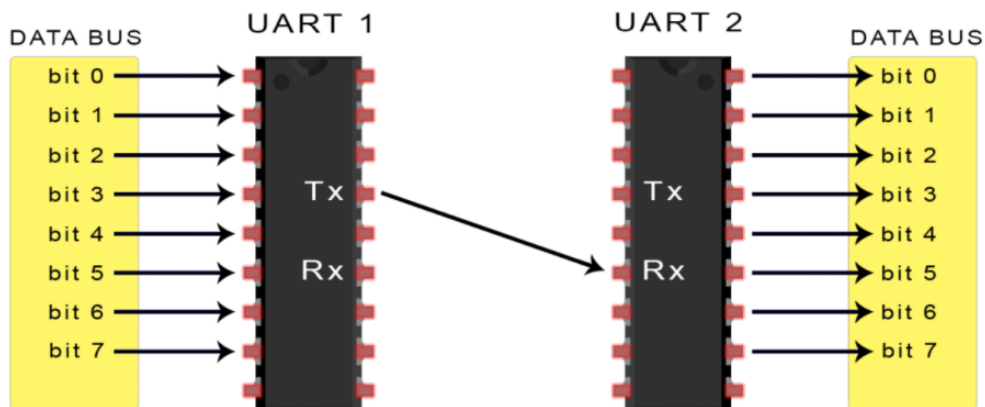


Figure 6: UART IC Communication

The transmitted data are in the form of packets. These are a structured set of bits that give the receiving UART more than just the bits of data. These packets include a start bit, parity bits, and stop bits. Without these extra bits that get transmitted, the receiver will have no way of determining where the set of data begins or ends; it would just be a mess of ones and zeros that don't have any meaning. The receiving end looks for the start bit so that it knows that the next 5-9 bits are the actual data being transferred. The parity bits after the data are to ensure that there was not any error with sending the data. If the parity bit is a 0, and the data has an even number of ones or if the bit is 1 and the data has an odd number of ones, then the data was transmitted correctly. When the result is not as it should be depending on the parity bit, the UART knows that the data became corrupted. Lastly, there are stop bits to signal the end of the transmission.

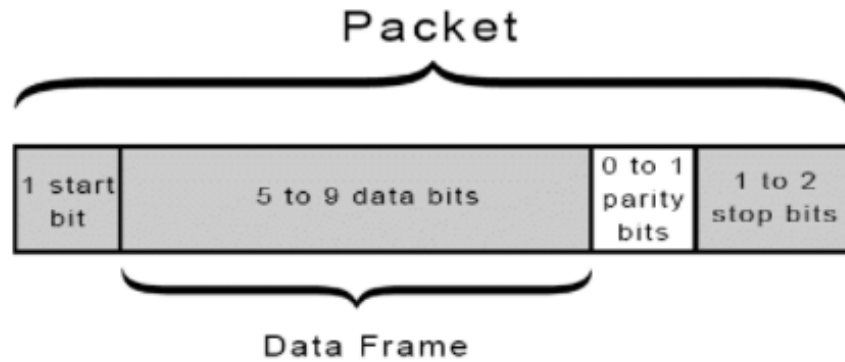


Figure 7: UART Package Structure

The other main difference between SPI and I2C is that fact that UART is asynchronous. This is done using the data packets mentioned above instead of having a clock signal such as SPI and I2C use. Using UART for serial communication is beneficial when needing an asynchronous platform.[4]

8.2 SPI

The Serial Peripheral Interface Bus, better known as SPI, is a protocol for serial communications. Some of the current applications for SPI include SD card modules, RFID card reader modules and wireless transmitter and receivers.

SPI is based on the Controller-Peripheral relationship which was known back in the day as the Master-Slave relationship. The controller represents the Controlling device such as the micro controller and the peripherals usually represent the sensors, display, or maybe a memory card. SPI allows for multiple peripherals to be connected to a single Controller but does not allow multiple peripherals to be connected to multiple Controllers. This is due to the MOSI, MISO and SCLK pins connecting to only a single Controller. A single controller-peripheral setup requires 4 wires:

- MOSI (Master Output/Slave Input) - Controller sends data to the Peripheral
- MISO (Master Input/Slave Output) - Peripheral sends data to the Controller
- SCLK (Clock) - Clock Signal
- SS/CS (Slave Select/Chip Select) - Controller can select which peripheral to send data too.

The Controller creates the output clock signal and selects which Peripheral to communicate with. If the Controller does not have multiple SS/CS pins, then it is required that all of SS/CS pins of the Peripherals be daisy chained together which is shown below in Figure 7. Using the clock signal output from the Controller the Peripheral knows at what rate to transmit the data. Thanks to the MISO and MOSI wires, data can be both transmitted and received at the same time. There is no package structure for transmitting data as UART and I2C have; instead, there is no limit as to how long the stream of bits must be. This does create some

small issues such as, there is no acknowledgment to know if the data that was sent was even received. Nor is there any way to check if the data being sent has been corrupted such as the UART has with the parity bits. Aside from that, SPI is still a widely used protocol that allows for data transmitting speeds that are faster than the I2C protocol.[5]

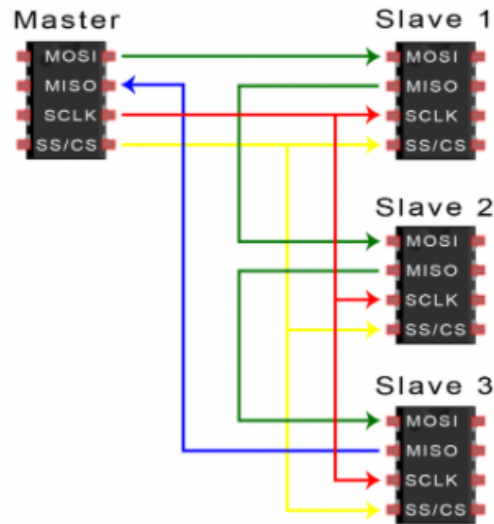


Figure 8: SPI Connection Structure

8.3 I2C

Short for Inter-Integrated Circuit, I2C is somewhat of a combination of both SPI and UART. Like with the SPI protocol, I2C can connect multiple Peripherals to a single Controller, but exclusive to I2C, it can have a single Peripheral connected to multiple Controllers. This is done with only two wires, similar to UART which also only needs two wires.

- SDA (Serial Data) - This pin is where data will be transmitted between the Controller and the Peripheral
- SCL (Serial Clock) - The clock signal will be sent to the Peripheral from here

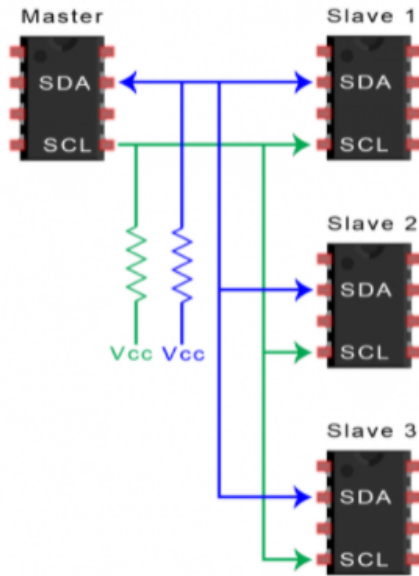


Figure 9: I2C one Controller to many Peripherals

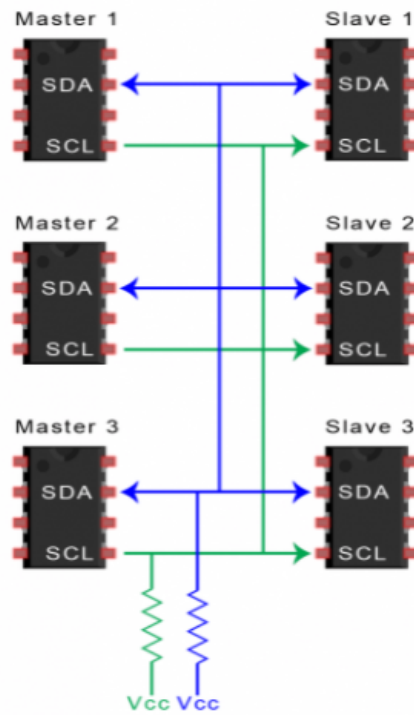


Figure 10: I2C many Controllers to many Peripherals

Similarly to UART, I2C also sends data through packages, except the packages for I2C are structured differently. Within each I2C package there are frames which are just a set of bits that correspond to something. Each package has the start and stop bit, but they also have a couple extra bits and frames such as the read/write bit, the acknowledgment bit, address frame and the data frame. For SPI, to differentiate the different Peripherals, each device had a SS/CS pin that the Controller would be able to reference. For I2C, each device has a specific address and there can be no two device with the same address or there will be conflicts with the data. Some device have a set address that cannot be changed while some others have the ability to change it. When the Controller sends everyone the address of the device it would like to speak to, that respective device then sends the acknowledgement back to the controller. Then there is the data frames; each data frame is accompanied by an acknowledgement bit. Each I2C device can send multiple data frames with each frame being 8 bits long. After sending a data frame, the next data frame cannot be sent until there is acknowledgment of the previous data being sent[6]

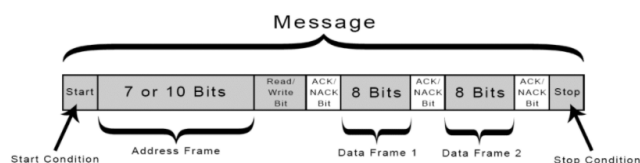


Figure 11: I2C Package Structure

9 Batteries

The voltage that the vehicle is run on is the foundation of the system, and has to be thoroughly considered since a higher voltage will allow us access to more power, torque, and speed when considering motor applications. However a lower voltage will be safer and cheaper for the end-user. There are a few options that our team will consider, but the biggest difference between them is what vehicle chassis we choose. Comparing battery form factor and technology will be broken down into sections based on their chassis below.

9.1 Power Wheels

The power wheels chassis has the most diverse range of batteries that could be adapted to power it. Running on 12V is safe and one of the most popular battery platforms being used today.

9.1.1 12 Volt OEM



Figure 12: Power Wheels Battery

The first, and easiest, option would be to use a sealed lead acid battery that is included with the purchase of the Power Wheels. This is normally a 12V-8Ah (Amp Hour) moderately dischargeable battery that has the ability to be recharged. The 12V system is a good idea due to being synonymous with the automotive industry. This will enable us to more easily interface with readily available components, such as light bulbs, without a voltage regulator.

The down sides to using the included batteries are the price, proprietary Power Wheels connectors, higher price, as well as a large physical size. The proprietary connector must be removed and standardized spade terminals attached to enable us to extend the runtime by placing multiple in parallel, or even series to overdrive the motor.

9.1.2 18V Milwaukee



Figure 13: Milwaukee M18 Battery

The second option is a popular modification due to the household availability of power tool batteries. These normally run on an 18V platform and include a built-in over-current and discharge protection circuit. Most generic brands could be used, but Milwaukee M18 batteries are being considered for this category due to already owning multiple batteries in the M18 ecosystem.

The greatest benefit of these are that they are made of 18650 lithium-ion cells. This means that they are lightweight, support very high amperage, able to be discharged much farther, and are much more efficient than a similarly sized lead acid battery.

9.1.3 Comparison

An aftermarket 12V 8Ah battery (EXP1280) is compared to a 18V 12Ah Milwaukee power tool battery, as it is the only battery with a technical data-sheet available. For reliable information with the M18 battery, data for individual cells are used, Samsung INR18650-30Q.

3.35 lbs	9.61 lbs	Weight
6 x 3.9 x 3.4 in	5.87 x 5.63 x 4.5 in	Dimensions
95%	50%	Usable Capacity
120 A	36 A	Current
300 - 73%	410 - 50%	Number of Charge Cycles & New Capacity
M18 [7]	SLA [8]	Category

Table 10: Lithium-Ion vs Sealed Lead Acid Battery Comparison Table

It can be seen from this table that the Milwaukee battery is the best option for performance, but for the cost and labor of upgrading the stock Power Wheels battery, there is not much improvement. It is determined that unless the Power Wheels battery can not support the load demanded from it, that it will not be replaced with an alternative.

9.2 Golf Cart & Hand-Built

For a chassis that is larger than a Power Wheels a higher voltage motor will be needed to provide the torque and speed to move increased magnitude of weight. There are two common voltages that these bigger vehicles operate at, 36V and 48V. This is not a standard battery voltage to buy, so cells must be linked in series, and then those packs in parallel to provide the necessary voltage and current.

9.2.1 12V Marine Deep Cycle



Figure 14: Interstate Deep Cycle Battery

The easiest solution is to purchase off the shelf moderately dischargeable batteries and link 3 or 4 of them in series to produce a nominal 36, or 48V. This will provide us with enough power to move our vehicle, and more batteries can be added in parallel to increase the operating duration available. This is the cheapest option for this category as it only requires 3 identical batteries as a proof of concept. The rough cost of this option is \$260 for a 36V system, and it is the mid-weight option compared to the other two options shown below. A large concern is the weight of the batteries, and the orientation they can be mounted in. The chemistry of these are flooded lead acid, which will leak if turned sideways. More negatives for a battery pack being used in this fashion are that there is not a fine control of the battery size, nor can we create custom cells to balance weight. The only control available is the battery capacity, and where we place each of the batteries. A large AWG wire must be ran between terminals, and as a result we increase the possibility of danger, whether in a crash or accidental slicing of the wire. This solution will provide 160 Ah, and a peak current draw of 600A if needed.

9.2.2 6V Golf Cart



Figure 15: Interstate 6V Golf Cart Battery

The second, and heaviest, option is to use the standard golf cart battery configuration which is comprised of six 6V batteries aligned in series to provide 36V with a capacity of 210 Ah, and a peak of 40A. The positives about this option is that there is already wiring and battery holders on a golf cart. This would be the safest option in the sense of weight balancing to prevent a rollover, as well as well the proper wire being used. A distributed load of more batteries will also run at a lower temperature. The downfalls come with a large price of roughly \$900 for a set of 6, as well as a large number of objects to secure, and connect, if our team does not use a stock golf cart. This option would only be used on a golf cart chassis, and not be viable on a hand built model because it would be too heavy and unsafe.

9.2.3 Comparison

Batteries used for comparison are Interstate: SRM-27 and M-GC2-UTL, with a respective price of \$139.99 and \$142.95 respectively. Both choices are readily available and were chosen due to their median level price-point, reputability of the manufacture, and available data sheets.

50.3 lbs	58 lbs	Weight
12.75 x 6.75 x 9.5 in	12.75 x 7.13 x 11 in	Dimensions
600 A	40 A	Peak Current
3	6	Quantity Needed
Deep Cycle	Golf Cart	Category

Table 11: 12V vs 6V Battery Options

9.3 All Applications

9.3.1 18650 Cells

This is the most time consuming and potentially dangerous solution, but will provide the best for results if done correctly. All 18650s are rated at 3.6-3.7V with a variance of capacity, a rough value being 2000mAh. The choice for our build would be the Samsung INR18650-20S. Its specifications are 3.7V, 2000mAh capacity, 30A constant draw, and price tag of \$6. The sizing of each is a little larger than an AA battery, at 18mm diameter and 65mm in height. A 36V pack built out of 18650 batteries will provide the best of both alternatives. The positives of this are the cells are lithium-ion cells for better performance, and that they can be separated and placed around the vehicle to balance weight, as well as the ability to create a platform on top of the chassis for the batteries.

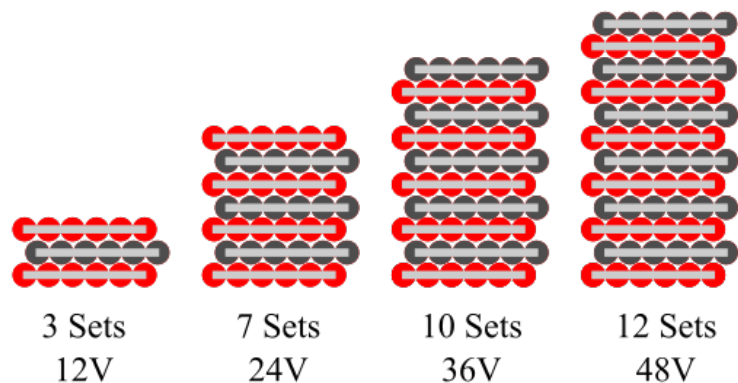


Figure 16: 18650 Pack Configuration

The platform would be a thin layer of the cells that are spot welded to strips of Nickel and then heat shrunk together. This could be any height or thickness we needed, and then encased in wood or metal to protect the batteries. This can be seen in the image above about the manufacturing and arranging of the individual

batteries. It would be very easy for us to change the voltage of our system from 36 to 48V by simply changing from a set of 10 batteries, to 13.

A rough calculation would require 10 batteries for 2Ah, and the usual draw of a motor at 20A would require 100 cells to operate for 1 hour continuously. This price estimate on a large vehicle would approach \$600, while significantly less for a power wheels at roughly 18 cells, a cost of \$110. To get an accurate weight estimate, each cell has a maximum weight of 1.7oz. This relates to a 2 lbs for a Power Wheels, and roughly 113 lbs for a full size golf cart, or hand made craft. Both of these estimates are significantly less than other options.

The major downfall to these batteries are their temperature sensitivity and danger when manufacturing the packs. The lead acid batteries described beforehand are corrosive, while lithium batteries can be explosive. As stated above, a strip of Nickel has to be spot welded, or soldered, to each side of the cell. Solder will rapidly raise the temperature of the battery and can ruin the battery, and potentially cause an explosion. Spot welding is preferred because it does not heat up the cells and will create a stronger bond. The only issue is that spot welding machines are \$200 for a budget machine, and up to \$20,000 for professional grade.

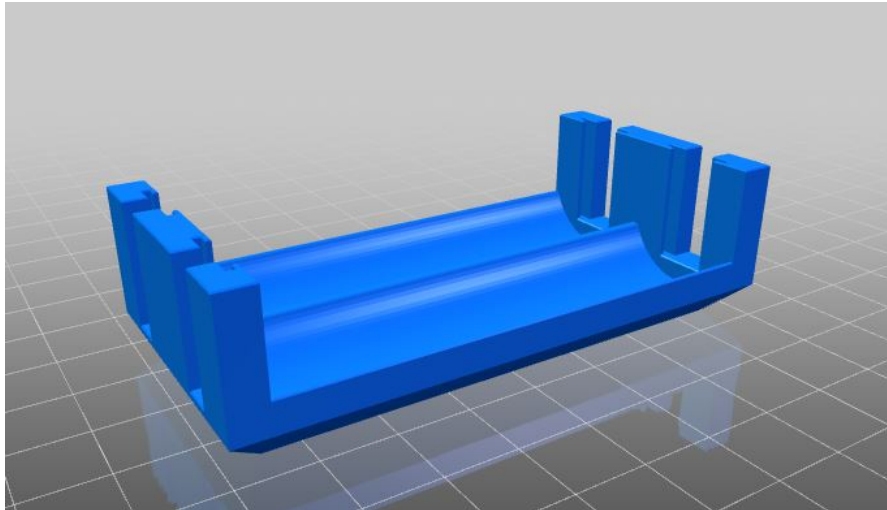


Figure 17: 18650 Cell Holder [2]

Lastly there is a final, more expensive option that could be used. Cell holders are priced in bulk at \$0.50 each and have leads for connecting the trays, rather than the batteries themselves. This allows for easy replacement if a cell dies. Storage using battery holders could be designed to accommodate all of the 18650s and not require any soldering or welding. This is not ideal as the springs can weaken over time, they can compress during bumps, and there is not always a good contact area between the terminals of the batteries. This option has proven to be safer during construction, but will show its faults in the increased physical form factor, reliance on more parts, and the ability to properly secure the batteries from moving and vibrations.

10 Solar Panels

For a larger vehicle, the purpose of a solar panel fixed on the roof is to provide a trickle charging resource to keep the battery topped off during times of inactivity. On a smaller vehicle the purpose can be shifted to charge the battery while the vehicle is not in use at a rate of roughly 8A. A solar panel is a beneficial due to the power draw of the resources on-board, such as the computers and cameras that always draw power. By adding a smaller panel to the roof we can negate the power drain of these components and increase our run time by a significant amount.

Each of our batteries have the ability to be charged by a solar panel, which is another reason they were chosen. A 100W solar panel will normally have an output voltage of 12, or 18V to cover our range. If a larger voltage is chosen, more panels must be used to make up the voltage, or a the panel could solely be used for charging the power supply that runs the computing and navigating devices.

A lower voltage is preferred for the panels due to simplicity and a cost factor. As stated above, a 100W panel running at 12 volts will produce 8.3A of charging current to recharge our unit. The draw we are looking at is around 15-20A, and this would mean that the car could almost be fully sustained by solar with the battery only being discharged when more power is needed. A battery charger for packs of this size ranges from 3-5A, so while the motors are not in use the extra current must be disposed of properly. A solution to this could be to run the panel at 18V and use a voltage divider built of resistors to burn 33% of the power. In doing this, a moderate amount of heat would be generated, but the current would plummet to 3.7A, which is safe for the batteries.

On the contrary, a 300W panel would allow the entirety of the power needed for a Power Wheels to be drawn from sunlight which means it would be powered by 100% renewable energy. Running at 12V we could achieve 25A of current, which amounts to a little more than needed to continuously run the drive motor and computers.

10.1 100W Flexible Panel



Figure 18: 100W Flexible Solar Panel

At the time of researching, the 100W panel is the most optimistic panel to prove as a proof of concept for recharging our batteries, and is easily expandable. The panel is a generic product that can be sourced from AliExpress under many different names for around \$80. Rather than go for a specialized and true to spec panel, that will be saved for the final revision as the cost is significant for our budget. The light absorption material is made of Monocrystalline Silicon, and the overall dimensions of the panel are 1050 x 540 x 3 mm. This translates to 41 x 21 x 0.1 in.

The ideal location depends on the chassis that we go with. In reference to the Power Wheels, the most suitable position is to be on the hood and up onto the windshield. This arrangement is possible due to the flexibility of the solar panel, but the downfall is that the sloped panel on the windshield will have sunlight blocked by a passenger, or when the vehicle is traveling away from the sun. This effect will be dramatically affected when the time of the test is not noon. Another consideration is smudging and dirt from being so close to the road, or the occupants hands.

If the golf cart/hand built chassis is chosen then we can position the panel on the roof to collect the most sunlight while being out of the way. It has the ability to lay flat which will prolong the life of the device as bending the silicon over a windshield can create a stress fracture after experiencing rough terrain, or a collision with road debris is more when it is front centered.

10.2 300W Fixed Panels



Figure 19: 300W Fixed Solar Panel

Similarly to the panel above, a cheaper generic panel would be selected due to the cost, and in-stock availability of purchase for a singular panel from a supplier. A fixed panel is not suitable for a vehicle without a roof because of its large size. The larger panel will be designated to full power delivery to operate the vehicle, and not only to part time recharging. With the limitations described in the 100W panel being an issue, it is not feasible to be low mounted.

The most economic solution is a glass panel that is sold in pairs of 150W each. Each panel can be ran at 12, or 18V from a built in charge controller. These panels have the ability to have two mounted on the roof of a golf cart. The frame is made of aluminum, the interior is made of Monocrystalline Silicon, and tempered glass holding the system together. The dimension of each panel is 970 x 530 x 24 mm, which translates to 38 x 21 x 1 in.

The largest reason that these are being considered is their ease of manufacturing, and their tried-and-tested technology. With these specifications we believe the vendors will be able to put out a better product in terms of durability. This is roughly the same size as the previous panels, but there is more freedom to mount them from the rigidity of the case. This enables the panels to be mounted over the roof edge, where the flex panel couldn't.

10.3 Charge Controller



Figure 20: Solar Panel Charge Controller

The charge controller is the most important aspect of a solar panel. It regulates the power being introduced to the system from the photovoltaic cell to prevent an over voltage or current issue. Sold in a bundle with both of the panels above, the generic charge controller will be used as it has the proper technical specifications. The voltage has the ability to be preset to either a 12V or 24V setting, and sustain a current of 30A. There are multiple connections on the device that link it to the panel, batteries, and the DC load.

In the case of the dual panels for 300W of generation, two of these units would have to be used to compliment each other. There would a slight increase in complexity and planning as they would be installed in parallel for 12V system, or in series for a 48V system. The connections do not need any special hardware or connections. The only calculation that must be made is the wire gauge depending on distance, voltage, and max current. We have estimated to use 10 AWG wire for a 12V run of 8 feet.

10.4 Comparison

As the chassis of the Power Wheels was selected, the only viable option is the flexible solar panel. This will allow the unit to be mounted anywhere on the vehicle. We do not need the large current provided by the hard panels which require 2 large units. The charge controller is included with their variation of the solar panels so there is no need for a decision on that.

On the Power Wheels the solar panel will be linked to the charge controller and providing 100W of power during the daytime. This will be used to charge the main 12V battery system and prolong length of run-time. From preliminary calculations, the motor will draw a max of 20A, so during sunlight the charger will be able to increase the time spent operating the vehicle by 50%.

11 Motor

There will be a minimum of 2 motors on the vehicle to control both latitudinal and longitudinal travel. In the prototyping stage a different setup may have to be

designed with a more complex traversing system due to unforeseen issues, such as the motor not being strong enough to move and steer at the simultaneously.

We will be using a large motor to act as a propulsion device to move the vehicle forwards, and potentially in reverse. In each of the chassis we present below, there is a gearbox that is included with the exception of the homemade vehicle. That option would not need a gearbox as the motor is overpowered for the size.

The more in-depth issue is to control the steering in the longitudinal sense. There must be a separate motor, or actuator, to dictate and force the front wheels to the left or right. They will work in tandem, and according to the latitudinal speed, the steering motor will adjust its amount of turn via software. This is due to needing less wheel turn at a higher rate of speed, compared to a low speed maneuver. This will be programmed to be proportional to the rate of travel. There will also be a safety cutoff to remove power to the steering motor if it were to jam, or fail.

11.1 Propulsion

The propulsion motor will have the highest power consumption of any electrical device used in this project. This is attributed to the amount of net weight that it is required to move, and as a result should have the most forethought into choosing which one to use.

In the case of the Power Wheels Motors, the gearbox will fit both motor candidates without modification to change the high rotational speed of the motors into a lower speed with much more torque. This is needed as both motors are used primarily in Remote Control cars and have been instead used in ride-on toys. They are somewhat suitable for powering Power Wheels, but when the toys stop working it is normally found to be the motor that burned up, rather than the gearbox. This leads to the conclusion that the vehicle is demanding more power and it should be upgraded if it fails, rather than a stock replacement.

11.1.1 RC 550



Figure 21: RC-550 Motor

This is the stock motor that comes installed in a power wheels. It has modest specs of power and can be ran at 12, or 24V to increase power and speed. At 12V it's rotational speed is 15,000 RPM, while at 24V it is 30,000 RPM. This will be translated by the stock gearbox to provide the torque that is required to get the vehicle moving. The dimensions of the motor are 57 x 36 with an output shaft having a diameter of 3mm, and a length of 10.5 mm.

The power wheels motor is driven by a potentiometer inside the pedal that increases, or decreases the resistance based on the amount of deflection from foot pressing. The further down it is pressed, the resistance goes towards the ideal of 0 ohm and gives full power to the motor. The pedal will be removed, and this process will be replicated via software.

Luckily, this is one of the easiest motors to work with as it is a standard brushed motor. The terminals are set up to be connected to a DC voltage source, and when polarity is reversed the motor will spin in the opposite direction. This is a very simple process and does not require the motor to come with a custom controller. We can build one by varying the input voltage to determine the speed we want the vehicle to move at. The microcontroller will be connected to a relay to drive the motor due to the current draws of 1.10A at 7.4V, 1.3A at 12V, and 2.55A at 24V being too large to direct connect to the board.

A great benefit to choosing this option is that very little modification would be needed for the system. None of the drivetrain would need to be disassembled, rather only the speed control. These are very cheap motors at \$15.99 each and widely used in the hobbyist world. If the motor were to burn out another one could be sourced locally.

For issues with lack of power, another could be installed with some fabrication and the motors can be driven in tandem to provide one motor to each rear wheel. This opposes the normal way of the motor inputting to rear differential that splits to drive both wheels. This is a big downfall for the off-road community and the future of our vehicle outside the proof-of-concept design. The most common setup of modern vehicles being driven is an open differential. This allows the wheels to spin at different speeds and allows them to make turns without hopping or binding. Without these, turns would be very difficult, and the life of the differential would be very short lived for on-road travel. While on-road these will prove to be great, off-road they are seen as the rival of the chassis. By allowing the slip rate of the wheels to be different differential will always spin the wheel with the least resistance. This phenomenon is explicitly shown in mud, sand, or surfaces covered with water. If one wheel latches onto resistance, the other will freely spin and rob power from the wheel that should be spinning. A very capable vehicle can immediately get stuck on the smallest of off-road obstacles, perhaps even a dirt road after a rainstorm.

With this issue being known, the rear drive with dual 550 motors instead of a singular one would be the preferred setup for this motor if chosen. This would act as a locked differential and allow us to force power to both rear wheels at the same time.

11.1.2 RC 775



Figure 22: RC775 Motor

The RC 775 is also a hobbyist motor that is described as the big brother to the RC 550. It gained its popularity from competitive racing of remote-controlled cars. This eventually leaked out into the home world and became a drop in motor to replace the 550 in Power Wheels when kids grew older.

This motor is very similar in form factor as its dimensions are 37mm diameter by 67mm in length. The shaft thickness and length are identical to the 550. It is a dual voltage motor that can be ran anywhere from 12V to 36V, depending on the power needed. The rotational speed can be pushed from 3500, to 9000RPM depending on the power given to it.

This motor is less resistant to burnout compared to others as it has a dual fan setup, and is made with high quality ball bearings. This will allow us to run it at a higher duty cycle, or a lower duty cycle with more weight. It is also commonly available for \$45, and is easily replaceable. This is a considered option if the RC550 can not supply enough power, or burns out.

11.1.3 MY1020



Figure 23: MY1020 Motor

The MY1020 is a motor designed by Monster Scooter Parts to be used in electric bicycles, scooters, and go-karts. It is running on 48V and uses 500W of power; a max current draw of 13.7 A. This drive selection would only be used on a homebuilt cart as it has an 11-tooth sprocket affixed to the end to accept an

8mm chain. It would require mounting to the top of the floorboard in the rear and a chain drive to a solid axle. The axle would be a simple bar of steel between the 2 rear wheels with another sprocket welded onto that. The tooth count of the sprockets will determine the torque and speed ratio.

This brushed motor is equivalent to a 0.66 HP gasoline engine which is about the smallest that we could use to propel a hand-built vehicle. Its dimensions are 135 x 108 mm, or 5.3 x 4.25 in and produces 1.90 Nm of torque. It has a max rotational speed of 3000 RPM, while 2500 RPM is recommended for cooling and longevity. The vehicle would not require a gearbox unless the torque was too little. It can drive a similarly weighted vehicle at speeds of 25 mph.

With a modest price of \$59 this is a very approachable option for a homebuilt vehicle, and currently the main one being considered.

11.1.4 Club Car 36/48V Motor



Figure 24: Club Car Dual Voltage Motor

The largest motor in consideration is a replacement motor that for many Ez-Go and Club Car golf carts. It will drop into both vehicles and was used for almost a decade so the replacement motor from an aftermarket company will be used for ease of data collection. This allows us to compare and contrast the motor inside the golf cart chassis, as well as purchasing it separately to use in a home-made chassis.

The aftermarket vendor who produces the motor is D&D Motor Systems, and it is marketed as a house brand. Throughout research it seems the motor is a generic and mass-produced Chinese motor like the solar panels. It is listed at \$530 and marketed as matching the stock speed and enhancing the torque by 10%. The weight is 52.00 lbs and spins at 4400 RPM to move the craft at speeds of 19 mph. There is no listed current draw, but the motor is rated at 9.8 HP. Through the calculation of $746W : 1 \text{ HP}$, the power consumed is 7,310 W. This can be broken down to 203 A at 36V, and 152 A at 48V. The voltage is selectable based on production environment it will be installed into.

The current draw is very high on this motor and will require a large set of deep cycle batteries, or a custom lithium-ion pack to be built. For our use case 48V is preferred to drop the current to more manageable levels, while only sacrificing space for one more 12V battery. It is the only option to use this motor in a stock golf cart, as it will come preinstalled and none of the other configurations will be able to drive the heavy vehicle. Very similarly to all the above motors, its direction is polarity controlled and can be done anytime the motor is at a standstill. A motor controller is bundled with the golf cart, and a toggle switch is located under the driver's seat to control forward and reverse. This allows a microcontroller to piggyback the controller and determine both the speed, and linear direction of travel. This simplifies a lot as the controller is already bucking the voltage to a level safe to directly control with our custom board and software.

For any application other than a golf cart, where it is included in the purchase price, the motor is very cost prohibitive. The reasons are that it requires an expensive motor controller, a large wire linking it to the battery, a high voltage battery pack, as well as the MSRP of roughly 10x the second most expensive option.

11.1.5 Comparison

Below is a comparison table about the different drive motors and their qualities. From this list and the decision on the chassis, the motor has been narrowed down between the RC 550 and RC 775. Since the RC 550 comes pre-installed with the Power Wheels, it is our decision to use that motor until it presents issues to us. Upon preliminary testing of the vehicle, if it does not have enough torque or speed to carry the weight that we need, it will be upgraded to the RC 775 as it is a direct swap motor. It is controlled in the exact same way as the stock motor so there will be no modifications needed to our software besides adjusting the acceleration speed.

The other motors were passed over due to the chassis that we chose, and the cost factor of implementing those. The redundancy factor was also in our favor with the cheaper motors as they will not destroy the budget if we ruin them during testing and assembly. The golf carts that interested us in our budget were old and the concern of a brushed motor deteriorating was a big decision to avoid the chassis.

RC 550	RC 775	MY1020	Club Car	Category
\$16	\$45	\$59	\$530	Cost
30,000 RPM	9,000 RPM	3,000 RPM	4,400 RPM	Speed
7.4-24 V	12-36 V	48V	36/48 V	Voltage

Power Wheels	Power Wheels	Homemade	Golf Cart / Home-made	Use Case
All	All	No M18, No PW	No M18, No PW	Battery Compatibility

Table 12: Comparison of Drive Motors

11.2 Steering

The drive motors must be a high speed, and torque, setup while the steering apparatus does not have to be as stout. The steering driver will link with the drive motor to allow a full range of motion, rather than just forward and reverse. There are three types of devices that are being considered; they are stepper motors, servos, and linear actuators. Each of these devices have their own speciality and can be configured properly for our project.

The functionality of a stepper motor is a very high pole count DC motor that can be controlled to rotate. It will be hooked to a gearing, or drive system, where the vertical rotation of the motor will be translated to left and right movement of the wheels.

A servo is very similar to a stepper motor, except that it has fewer poles. Where the stepper motor can be stopped at any pole, the servo motor must be controlled with a controller for precise movement. This is good for quick movement, but is not as accurate as a stepper.

Finally a linear actuator will remove the rotational element, and can be thought of as a retracting and expanding piston. If a track bar is connected to both wheels, when the linear actuator is toggled it will pull or push the bar forcing the wheels to turn. This is a good solution for automated driving, but will not work for manual steering - such as a steering wheel.

11.2.1 Tonegawa 050 Servo



Figure 25: Tonegawa 050 Servo Motor

The Tonegawa servo is a very well built device that is mainly found in industrial and medical equipment that requires a low failure rate. Due to that fact its cost is higher due to more research and development. The small steering motor is roughly \$200 while boasting high torque numbers. It contains 3 poles, and will produce 49 kg/cm of torque at its highest setting at 8.4V. Its miniature form factor is contained in a package of 76 x 45 x 80 mm, or 3 x 1.75 x 3.1 in. It requires 0.21 seconds to rotate 60 degrees, and can make one full rotation in 1.26s.

Rather than having to create a custom linkage, the servo can be attached to the steering stem to imitate the steering wheel being turned by a passenger. For appearance it could even be tucked under the dash and the steering wheel left in tact for passenger comfort. The typical rotation for an automobile turn is 90 through 180 degrees while at a moderate speed, and up to 2 turns for low speed. With the speed the craft is projected to be traveling at, there is no conflict of the max speed of the servo and the possible maneuvers that will need to be made.

The only downfalls that could be found about this motor are the high price tag, and the need for dedicated 12V rail to power the device. The 12V rail is not a big issue as the microcontroller will be determining the angle of rotation, while the power will come from a separate battery.

11.2.2 Automotive Windshield Wiper Motor



Figure 26: Automotive Wiper Motor

A more budget candidate is an old car windshield motor that can be repurposed. There are many generic models, but the Crown Automotive 12V kit is the most approachable from it's many 5-star reviews. These types of motors are great for the elements and terrain due to their rugged and small design. Normally they will sit in a cowl and be exposed to heat, moisture, and extreme use when needed. It offers a rapid change in direction without a cam/linkage. The torque on this model is not listed, but from experience they are lower than the servo discussed above.

The assembly is 4 x 5 x 5.5 in and has a screw terminal to mount a DC connection. The max current draw is 4A at 12V. Unlike the servo, this will require

a relay as there is only one power line to the motor, there is no control wire. A third yellow wire is attached as a parking feature for automotive, but will not be used in our application. These motors are also equipped with 2 or 3 speeds, rather than an being adjustable. This will cause the steering to be too slow, or quick in certain situations.

11.2.3 Nema 34



Figure 27: Nema 34 Stepper Motor

Producing almost the same numbers in a form factor similar to the industrial servo above, is the Nema 34 Stepper. This was chosen to fill the stepper category for precise motion. It is a 2-phase motor that has an arc of 1.80 degrees per step and can produce 49 kg/cm of torque. The current draw is 6A, and can sustain speeds of 1500 RPM, or 25 full rotations in 1 second. The dimensions are very similar at 86 x 86 x 80 mm, or 3.4 x 3.4 x 3.1 in. The hosing has 4 mounting holes around the shaft, and a keyed rod that is 37 mm, 1.5 in, in length.

It is currently priced at \$30 and a great option if a suitable 2-phase motor controller can be found.

11.2.4 Firgelli Linear Actuator



Figure 28: Firgelli Linear Actuator

This is potentially the easiest, and most elegant, solution that could be added. The simplicity of mounting and steering of the wheels will eliminate a lot of issues with returning the stepper motors back to their initial position. Rather than fuss with complicated algorithms, the linear actuator can be controlled with a simple closed loop feedback system. It is simply thought of as an extended piston mounted in parallel with the front axle. If the mounting position has the piston facing right, when it extends it will push the front right wheel outward, dragging the front left wheel inward causing the vehicle to steer. When the polarity is reversed it will draw in the piston and it will turn left. Rather than convert rotational energy through linkages and lose torque, the simplicity of pushing and pulling the wheels to the direction they need to go is the easiest.

The Firgelli Actuator is ran on 12V and a max current of 5A on the model with the largest force, of 200 lbs. They are highly configurable and the piston can be swapped to accommodate different lengths. They are sold in 30 different setups. Three of the options are force; there are 35, 150, and 200 lb options. Beyond that the other configurable option is the stroke length of the piston. Lengths of 1, 2, 3, 4, 6, 9, 12, 18, 24, and 30 inches are available. For our application the 35 lb force is all that is needed for powerwheels, and the 150 lb option for a golf cart or home-built vehicle. The actuator stroke length is to be determined when a prototype is selected, but estimated at 4 inches for a power wheels, and 9 inches for a larger vehicle. The stroke length is 2 in per second, so the shorter the better.

4"	9"	Stroke Length
12.53"	22.53"	Extended Length
8.53"	13.53"	Retracted Length
1.92 lbs	2.48 lbs	Weight

Table 13: Linear Actuator Physical Dimensions Based on Stroke Length

The form factor of the actuator housing varies based on stroke length and is shown in the table above. Variances are not noted between models of varying force.

11.2.5 Comparison

This decision of selecting the Firgelli Linear Actuator as our device of choice was done based on the amount of force it could provide and the ease of integrating it with our system. The big concern with force was for low speed maneuvers when the vehicle was near its max weight. With an indirect driver there would be gear ratios to calculate, as well as designing a gear system, as well as retaining the steering wheel for manual control. With the servos and steppers the wheel would be replaced by a cog and then the motor attached to that. With this linear actuator it would be mounted under the vehicle and retain the steering wheel. The linear actuator could be overpowered by the user, with a slightly larger resistance being felt by the operator, compared to stock.

The manual operation of the power wheels was a big plus so that it could be pushed into a building, or onto a trailer, if the electronics die. Since there is no battery backup after the main drive battery dies there would be no other easy way to transport an inoperable vehicle. With this issue being solved, the actuator can provide 35 to 150 lbs of force which is more than enough to steer any vehicle we considered. The final decision that will be determined during prototyping and after acquiring the Power Wheels is to decide the stroke length needed. The lead time is very low as they are in stock and 2-day shipping is included.

The only downfall of choosing this actuator is that it will be exposed to the elements of the road, and terrain the vehicle is traversing. Since it is mounted below the vehicle it could potentially come in contact with water or rocks. To slightly mitigate this issue, the actuator will be mounted higher than the vertical center-point of the front wheels. This should protect it from a majority of the issues and a splash shield can be designed if the effects of the environment are found to be too harsh.

Tonegawa 050	Windshield Motor	Nema 34	Linear Actuator	Category
\$200	\$60	\$30	\$110	Cost
Servo	Servo	Stepper	Linear Actuator	Type
12 V	12 V	3 V, 2-Phase	12/24 V	Voltage
Power Wheels	Power Wheels	Homemade & Power Wheels	All	Use Case
Fast	Fast	Very Fast	Very Fast	Speed
Steering Shaft	Steering Shaft	Steering Shaft	Front Axle	Mount Location
49 kg/cm	Unlisted	49 kg/cm	35-200 lbs	Force
No Golf Cart	Power Wheels	No Golf Cart	All	Use Case

Table 14: Comparison of Steering Motors

12 Wheel Hubs & Brakes

Essential to any vehicle is a braking system that can safely stop the chassis in both, regular and emergency, situations. The braking system must be designed for the maximum gross weight of the cart, plus about 30% to accommodate for any failures of the system. With the extra stopping power, the vehicle can lose one wheel and still safely halt itself.

There are 2 main styles of brakes that are used on vehicles, disc and drum brakes. Disc brakes work by having a rotor attached to the axle or wheel hub, and mounted to the chassis is a caliper that sits on the top of the rotor. The rotor will spin as the drive-line does, and the caliper compresses the brake pads to contact the rotor and add friction, thus slowing the vehicle down. This creates a tremendous amount of heat, but is the most efficient method of mechanical non-regenerative braking. It is also very simple to work on and the easiest to set up.

The second braking option, drum brakes, are an older technology that has been

phased out due to its complexity and inefficiency. This works by having a cylinder that is hollow slide onto a grouping of brake shoes that are pulled inward to the center by an array of springs. When the brake pedal is pressed fluid will force the brake shoes outwards towards the brake drum causing friction and the shedding of speed.

There are benefits and drawbacks to both, but the summary can be seen in the table below.

Disc	Drum	Category
High	Low	Cost
Less Dust	More Dust	Dirtiness
Higher	Lower	Stopping Power
High	Low (Brake Fade)	Heat Dissipation
On Hub, or Axle	Hub Only	Location
Yes	Poor Wet Performance	All-Weather Use
High	Average - More Moving Parts	Safety

Table 15: Comparison of Brake System Types

12.1 Power Wheels Brakes

Power Wheels have an interesting setup as they do not have conventional brakes as discussed above. They use the principle most commonly referred to as engine braking, with gasoline engines. A motor can be wired to a system in two ways, one being connecting the motor to a battery to force it to spin, and another being spinning the motor manually to generate a voltage across its terminals.

When the power wheels is in motion, the motor is connected to the battery in the first manner to make it spin. When the pedal is released the contacts on the top of the motor are short circuited. This forces the motor to generate electricity rather than use it. By doing this, energy is converted away from the kinetic energy causing motion. The vehicle will slow to a halt. This is possible due to the low speeds and high torque of the RC 550, or similar, motor.

12.2 Golf Cart Brakes



Figure 29: Golf Cart Drum Brake

On a majority of golf carts drum brakes come standard on the rear driver and passenger wheels. Only 30% of braking power comes from the rear axle, but to cut costs and improve the handling of the cart, they are only mounted on the rear. When a front brake is triggered the cart will have a tendency to dive forward and is felt by the occupants. To combat this the rear is equipped with the stopping power and the front suspension supports the vehicle on deceleration. Even though the front has the possibility to provide much more leverage for braking power, it is not needed due to the low speeds and weight distribution.

The golf cart braking system would be kept stock and is designed as a single axle drum brake configuration. It is cable driven so that when the brake pedal is pressed, the cable will contract forcing the the shoes outward. The reason that cable actuation is possible compared to requiring a fluid is from the small size of the drum, infrequent use due to mechanical turning of the motor, and slow speeds of travel. If a modification was done to increase the top speed, a fluid and master cylinder would be required due to the heat.

12.3 Home-Built Chassis Brakes

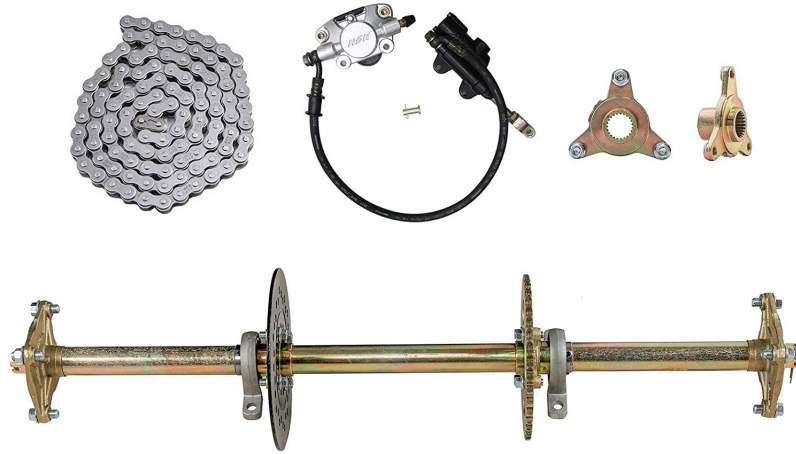


Figure 30: Do It Yourself Disc Brake Kit

By building our own chassis we have the most freedom to design a braking system that is competent for our vehicle, as well as the potential of having a 4-wheel breaking system for redundancy and better performance. The kit that has the most potential is made by the company Fuerduo, and retails for \$190. It is a rear axle kit that mounts with a U-bolt to the chassis and has 2 sealed ball bearings to isolate the rotation from the chassis. It includes one rotor mounted to the axle, a sprocket to be driven by the motor, 2 outer hubs, and one caliber that is cable actuated.

The reason for considering this kit is that it can support speeds of 40 mph and has the benefits of disc brakes, with a low entry cost. The axle is made of HSS, high strength steel, and can support 500 lbs of direct weight. It is perfectly balanced and symmetric so that the sprocket could be positioned on either side.

Front brakes would not be used on our chassis due to cost constraints and lead times of those parts. The axle would serve as a proof of concept for future expansion and meets all of our weight and stopping requirements.

12.4 Comparison

There is no need to compare the brakes as the chassis will determine the product that is used. The Power Wheels frame does not use the conventional brake system so nothing will be need to be installed. The only issue that could possibly arise is the amount of stopping torque that the RC 775 motor could provide compared to stock. If this is much more than the stock setup it could potentially jolt the vehicle to a stop when the accelerator is released. To mitigate this issue a resistor could be added across the motor terminals to slowly stop the motor rather than a rapid deceleration. Another option to mitigate this risk is to use software to gradually interpolate the value from its cruising speed to a stop. This will decelerate the Power Wheels while maintaining comfort and stability. Both options are valid solutions, and a if a modification is needed it will be added during the building and testing phase.

13 Microcontroller

This section will discuss the microcontroller that will be implemented into the project. It will discuss the options that we have for the microcontroller, the comparison of the different options, and the microcontroller our team will choose to implement into the project. If our team does not pick a microcontroller that meets the needs of our project we could have many issues, such as overloading the microcontroller's processor which would cause a crash, not having a microcontroller that can handle all of the sensors we require, or having a microcontroller that simply isn't fast enough to relay the data to the display fast enough.

The microcontroller controls the custom circuit board that will be implemented into our project. This custom circuit board will handle most of the sensing the vehicle will need to do. This sensing doesn't include the more taxing sensing such as object recognition, which will be handled by an AI machine discussed later.

The first option is the MSP430FR6989. This option was chosen because every member of the team has experience working with this microcontroller. This microcontroller also has the lowest power consumption out of the available options. This option is also the only option that has a 16-bit processor, the other two having an 8-bit processor. If our team needs our microcontroller to be fast and power efficient, this would be the best option to choose.

The second option is the ATMEGA328p. This option was chosen because it's small and has a DIP, also known as through hole, configuration for easy prototyping. The ATMEGA328p is the least powerful option out of the available options, however it is capable of using serial communication, so we would be able to use multiple microcontrollers throughout our vehicle instead of having a single microcontroller handle all of the sensors. The ATMEGA328p is a viable option if our team needs to iterate through a lot of different prototype circuits.

The third option is the ATMEGA2560. This option was chosen because it's easy to program while still having good specifications. This option also has twice the memory of the MSP430FR6989. Due to the large memory size, and having an 8-bit architecture, this option consumes the most power, which may impact the amount of batteries our team will need to implement into the vehicle. This option will be the best option if our team want to prioritize a single chip for handling the sensors, and doesn't need it to be very fast.

13.1 MSP430FR6989

The first consideration for the microcontroller is the MSP430FR6989 from Texas Instruments. This is a 16-bit ultra-low power microcontroller that is designed for sensing applications. This microcontroller would allow our team to collect all of the data from our sensors from a single chip while having the least amount of power usage out of the available options. This option is much harder to program however, due to the 16-bit architecture. The community support from Texas Instrument's products are also not as good as Atmel's. This wouldn't be an issue if our team had a long time to create build our project, but with the limit time frame we do have it wouldn't be in our best interest to allocate a large amount of it towards reading data sheets and getting the experience needed to comfortably program the MSP430FR6989.



Figure 31: MSP430FR6989 Microcontroller

CPUXV2	Core Processor
16-Bit	Core Size
16MHz	Speed
I ² C, IrDA, SPI, UART/USART	Connectivity
128KB (128K x 8)	Program Memory Size
FRAM	Program Memory Type
-	EEPROM Size
2K x 8	RAM Size
1.8V 3.6V	Voltage - Supply (Vcc/Vdd)
Surface Mount	Mounting Type
83	Number of I/O
10.11	Cost (\$)

MSP430FR6989	
--------------	--

Table 16: MSP430FR6989 Specifications

Even though this is the most advanced microcontroller, it's not the most expensive out of the available options. The cost of this microcontroller is around \$10, which is slightly cheaper than the ATMEGA2560 microcontroller. There is an additional cost not factored into the price of the microcontroller. This price is the cost of the TI LaunchPad Development Kit. Our team will each need to buy one or more of these Development Kits since the MSP430FR6989 only comes in a surface mount pin style, and we will not be able to prototype the microcontroller on a breadboard because of this. This also means if one of the Development Kits shorts out and becomes unusable we will have to purchase another one to continue development.

The potential of this microcontroller makes it a great consideration for this project, since it is a strong, low-power, option. The extra costs and learning curve are some things our team needs to weigh against the potential however. If our team doesn't have the time to learn how to comfortably use the MSP430FR6989, it could mean the entire project could be at risk, since this is a key component in our vehicle functioning. The added cost of the TI LaunchPad Development Kit is also something our team needs to take into account. It would quickly burn through our budget if we have to buy too many Development Kits on top of the costs of the microcontroller itself. When our team has all of our sensors chosen and power options figured out, we can decide if this is the best option, or if another option would benefit our project more.

13.2 ATMEGA328p

The second consideration for the microcontroller is the ATMEGA328p from Atmel. This is a lower end microcontroller that has limited general purpose input/output (GPIO) pins. Despite this being a lower end microcontroller, it can contend with the other two options. The limiting factor on this board is the small amount of GPIO pins, having 1/4 of the amount as the other options. The small amount of pins on this microcontroller means we would have to use multiple microcontrollers throughout our vehicle in order to handle all of the sensors. Depending on the serial communication our team decides to use, this could cause problems since not all of the serial communication options are capable of having multiple peripherals, or get more complex with every new peripheral added.

AVR	Core Processor
8-Bit	Core Size

20MHz	Speed
I ² C, SPI, UART/USART	Connectivity
32KB (16K x 16)	Program Memory Size
FLASH	Program Memory Type
1K x 8	EEPROM Size
2K x 8	RAM Size
1.8V 5.5V	Voltage - Supply (Vcc/Vdd)
Through Hole	Mounting Type
23	Number of I/O
2.63	Cost (\$)
ATMEGA328p	

Table 17: ATMEGA328p Specifications

This microcontroller has the perk of having the capability of being programmed in multiple different languages, however. The first language being C, and the second being Arduino. Arduino is an easy to use language that has an extensive sensor library. This would give our team a big head start when prototyping, and would allow us to focus on the implementation of the sensors, rather than worry about the troubles that programming microcontrollers can have. This would also allow the members of the team that don't have a lot of experience with C the opportunity to help with the sensors and learn more about embedded systems.

The biggest downside of this option is the possibility of power issues caused by running all of the microcontrollers and sensors, as well as the dashboard and the motors. Because of the power issues we may run into by powering multiple microcontrollers, we would have to take precautions and either add more batteries to the vehicle, which would add a lot more weight, find better solar panels, which has the possibility of pushing us over budget, or completely redesigning the power system, which may prove to be too difficult to do given the time frame. There

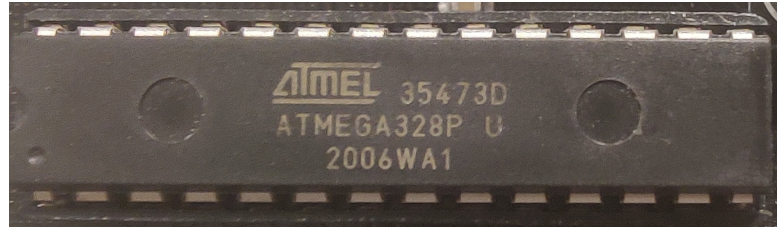


Figure 32: ATMEGA328p Microcontroller

are a lot of factors that are hard to control by having tens of microcontrollers throughout the vehicle. If we run into power issues by overtaxing the system, we could possibly cause a fire, or short out another electrical component that could cost hundreds of dollars to replace.

13.3 ATMEGA2560

The third consideration for the microcontroller is the ATMEGA2560 from Atmel. This is a high end 8-bit microcontroller that was made for complex projects that require a large amount of memory and GPIO pins. This microcontroller is a good median between the other two microcontrollers. This microcontroller would allow us to comfortably collect the data we need from a single chip, with the caveat that this microcontroller uses more power than the other two options. This microcontroller can also be programmed in two languages like the ATMEGA328p, however. Having the same language options, C and Arduino, this option also has the perk of Arduino's extensive sensor library and excellent community support.



Figure 33: ATMEGA2560 Microcontroller

AVR	Core Processor
-----	----------------

8-Bit	Core Size
16MHz	Speed
EBI/EMI, I ² C, SPI, UART/USART	Connectivity
256KB (128K x 16)	Program Memory Size
FLASH	Program Memory Type
4K x 8	EEPROM Size
8K x 8	RAM Size
4.5V 5.5V	Voltage - Supply (Vcc/Vdd)
Surface Mount	Mounting Type
86	Number of I/O
14.00	Cost (\$)
ATMEGA2560	

Table 18: ATMEGA2560 Specifications

The program memory and RAM are the perks of this microcontroller. The large program memory means we wouldn't have to worry about running out of space for our program. Since we're using many sensors that are connected with both an AI machine and a central computer, it's possible that our program will grow to be very large. Like the ATMEGA328p, this large amount of program memory means that the program we make doesn't have to be perfectly optimized in order to be functional. This is important since time is limited in this project, and it's better to have imperfect code that works, rather than running out of time trying to make dozens of optimizations for our code to fit and not having any working code that can fit on the microcontroller.

This microcontroller is a good balance between the other two microcontrollers, and will be a great choice if our team wants to prioritize other aspects of the project. This option will allow our team to not have to worry about the smaller

details of implementation, while also being a strong microcontroller capable of handling complex tasks. The only issue our team would have with the microcontroller is the power management, which means we would have to add extra batteries into our power system, or have a dedicated solar panel and battery for providing constant power, so other systems wouldn't be starved.

13.4 Comparison

Comparing the different options leaves our team with a lot of decisions to weigh. The specifications in the microcontroller sections above will be used to compare the different microcontrollers. The factors our team are considering that are listed in the specification tables above are: core processor, core size, speed, connectivity, program memory size, program memory type, EEPROM size, RAM size, voltage, mounting type, number of I/O pins, and cost. There are many more features that are left out of the comparison because they all meet the same specification for the features, or will not impact the final design.

13.4.1 Core Processor

The core processor in the tables above shows which processor core the microcontrollers use. The core processor determines the register banks, ALU, data path, and control logic. This is essentially the brain of the microcontroller, and will handle all data processing.

The focus in this section will be about the software development environments used to program the microcontrollers. It is worth noting however, that the MSP430FR6989 uses a von-Neumann architecture, while the ATMEGA328p and ATMEGA2560 both use a modified Harvard architecture. This architecture determines how memory is accessed, and how many clock cycles it takes to perform an instruction. The von-Neumann architecture uses the same data bus for instruction fetches and data transfer, while the modified Harvard architecture uses separate busses for instruction fetches and data transfer. This means the von-Neumann architecture will take 2 clock cycles to execute an instruction, while the modified Harvard architecture takes 1 clock cycle to execute an instruction. Strictly speaking the modified Harvard architecture is more suited for microcontrollers, though there are arguments for use of the von-Neumann architecture.

The MSP430FR6989 has a CPUXV2 core processor. This microcontroller has the option of being programmed in TI's Code Composer Studio (CCS), or the open-source software Energia. Code Composer Studio is an IDE that is based in C, and has a full list of features that make developing for the MSP430FR6989 straightforward. All of the members of our team has experience coding in CCS, and developing for the MSP430FR6989, so this is a great choice to get everyone involved in the software side of this project. The other option, Energia, is an IDE that bridges between the Arduino framework and TI's microcontrollers. None of the members of our team are familiar with this IDE, however Arduino based environments are easy to develop in, and have a full range of examples that help to make coding as streamlined as possible.

The ATMEGA328p and ATMEGA2560 both have an AVR core processor. These microcontrollers have the option of being programmed in Microchip Technol-

ogy's MPLAB IDE or Arduino IDE. MPLAB is an IDE that is based in C, and is used for developing for the PIC microcontroller line, however the latest release MPLAB X is used for project management, code editing, debugging and programming PIC and AVR based microcontrollers. None of the members of our team has any experience developing with MPLAB, however after some testing it has a similar interface to CCS, which would make developing in MPLAB easy to learn. The other option, Arduino IDE, was designed to be an easy to learn, and easy to use, environment that uses a C based language. The one downside to Arduino IDE is that it's not a full featured IDE, and lacks a lot of the convenience of developing in modern IDEs. The biggest feature it lacks is intellisense, which is used for autocomplete, code hints, and autocorrect. Debugging is also a challenge in Arduino IDE, and is a feature that will be important for developing our software.

13.4.2 Core Size

The core size in the tables above shows which type of core size the microcontroller use. The core size determines the width of the bus used for data. The data bus is used for arithmetic and logic operations within the core processor. The bit size corresponds to the length of a word that the microcontroller uses. This also tells you the size of the registers, the number of memory addresses, and the largest numbers they can address.

The MSP430FR6989 has a 16-bit core size. This core size is more complicated to work with, and may involve a lot of time to get familiar with the architecture. However this core size may be useful if our team has to do a lot of data processing with large numbers. A 16-bit core size can handle numbers from 0 to 65535, or 2^{16} . Data type sizes also change depending on the core size. This can lead to many problems if our team is not careful with the development of our code. The best part about a larger core size is the faster processing speed associated with it.

The ATMEGA328p and the ATMEGA2560 both have an 8-bit core size. This core size is easier to work with than the 16-bit core size. This will allow our team to quickly learn the architecture and work on development. Time is a big limiting factor in our project, and we cannot afford to spend too much time learning how to program a microcontroller. However, working with bigger numbers is a challenge our team will have to overcome if we decide to work with this core size. An 8-bit core size can only work with numbers up to 255, or 2^8 . It's not impossible to work with larger numbers, however. By splitting the number into parts and using multiple cycles to calculate them, you can work with larger numbers on a smaller core size. This will require more work from our team though, and could become unnecessarily complex.

13.4.3 Speed

The speed in the tables above shows the maximum frequency the clock of the microcontroller runs at. A faster clock speed will determine how fast the processor is. Clock speed also determines other factors of the microcontroller, such as baud rate, the speed of the ADC, and the rate the timer counts at. A microcontroller running at a faster clock speed will also use more power than one going at a slower clock speed. All of these factors are important to consider.

The MSP430FR6989 and ATMEGA2560 can use up to a 16MHz clock. The ATMEGA328p can use up to a 20MHz clock. Since our vehicle will be taking data readings in intervals, it may not be useful to choose a microcontroller based on its maximum clock speed. For our case, we will be using a 16MHz external crystal oscillator, regardless of the microcontroller we choose. Since the MSP430FR6989 takes 2 clock cycles for a single instruction due to the architecture, it may impact how fast we can collect data, however. Our team will have to do extensive testing on this matter to determine the appropriate course of action in regards to clock speed. In general, clock cycle is not the determining factor for choosing a microcontroller, as other requirements are often more important. Clock cycles can also be changed depending on what the microcontroller is doing. Dividers are used for this purpose so that a clock can slow down during a low power state, so that the microcontroller does not consume as much power.

13.4.4 Connectivity

The connectivity in the tables above shows the different communication methods the microcontrollers have available. Our team will be using many different sensors in our project, and it's important that our microcontroller has communication options that are compatible with the communication types of the sensors. The options our team are considering are UART, SPI, and I2C communication. These all have their respective advantages and disadvantages. The comparison and decision of these communication types are discussed in the Serial Communication section.

All of the options we are considering have options for I2C, SPI, and UART/USART communications. The deciding factor for the microcontroller will not be the communication type, since all of the microcontrollers are capable of using the communication types our team is considering. How hard it is to implement these communications into our microcontroller will be discussed at a later time, and this may impact the decision we make. If an IDE and microcontroller are particularly difficult to program these communication types in will have to be researched further, and may cause us to have to use a different microcontroller if it proves too difficult to program on a certain IDE.

13.4.5 Program Memory Size

The program memory size in the tables above shows the size of the program memory the microcontrollers have. The program memory holds the code that our microcontroller will execute. Broadly speaking, a large program memory size is favorable, however it's not a simple choice in our case since the large amount of sensors our team is using in our vehicle could be split between multiple microcontrollers. If this is the case, the program memory would be best if it was smaller so we don't waste a lot of resources on unused space.

The MSP430FR6989 has a 128KB program memory size. This is half as much as the ATMEGA2560, and may be a problem if our teams wants to run all of our sensors off of a single microcontroller. Since the program has not been created, it's unknown whether or not this will be large enough to store the program. Since this microcontroller is programmed in a different IDE than the other two options,

it may not be a good choice to risk our program growing larger than the program memory size of this option.

The ATMEGA328p has a 32KB program memory size. This seems like a small amount of program memory, however this microcontroller would be paired with either the ATMEGA2560, or with multiple of the same microcontroller. By doing this, the microcontrollers would be individually programmed to function with one or two sensors, which would break up our code into many smaller programs. This choice relies on I2C or SPI communication, which can have multiple peripherals, and will not be an option if our team chooses to use UART.

The ATMEGA2560 has a 256KB program memory size. This microcontroller has the largest program memory size of the available options. This option would allow us to program a lot more onto the microcontroller, twice as much as the MSP430FR6989 and eight times as much as the ATMEGA328p. If our team wants to use a single microcontroller to control all of our sensors on the vehicle, the ATMEGA2560 is the safest choice if our team wants to ensure that the microcontroller has a large enough program memory to store a large program.

13.4.6 Program Memory Type

The program memory type in the tables above shows the type of program memory the microcontroller use. The program memory type has a big impact on how fast the program memory can be written. Power consumption is also impacted by the program memory type. Our team will have to determine if a faster write speed is important for our project. Depending on the programs, we may update the system many times before settling on a final program. Over the course of our project we could save a lot of time by choosing the best program memory type.

The MSP430FR6989 uses an FRAM program memory type. This is a newer technology than FLASH memory, and it extremely fast. On average there is a 16x speed increase from FLASH memory when writing to FRAM and a 21x energy consumption improvement [9]. This would be good for rapid updating and prototyping. FRAM is also capable of being rewritten many times more than FLASH. While our team does not plan to exceed FLASH's writing capacity, it's a safer choice to choose FRAM over FLASH for safety.

The ATMEGA328p and ATMEGA2560 both use a FLASH program memory type. As discussed in the previous paragraph, this is an inferior memory type when compared against FRAM's speed and energy saving, however it's not without its advantages. The best part of FLASH is that it has a near infinite amount of reading cycles. Our team will not exceed the thousands of write capacity that FLASH has, but it is possible that our vehicle will read from the program memory hundreds of thousands of times over the course of its life. It may be better to consider using a microcontroller with FLASH rather than FRAM due to the stability of its read cycles.

13.4.7 EEPROM Size

The EEPROM size in the tables above shows the size of the EEPROM the microcontrollers have. EEPROM is a type of memory that is used as non-volatile memory. It can be erased and rewritten in circuit. This type of memory is impor-

tant to store data that won't change often or at all. EEPROM can be used with FLASH memory to preserve the FLASH memory cells.

The MSP430FR6989 does not have an EEPROM. This is not a problem since the MSP430FR6989 uses an FRAM program memory type. The write cycle limit of FRAM has a much larger capacity than FLASH, so there isn't a need to have EEPROM.

The ATMEGA328p has a 1Kx8 EEPROM. The small amount of EEPROM shouldn't be a problem for this microcontroller since our team would be using multiple microcontrollers to control the sensors. Much like the small amount of program memory, having a small EEPROM is preferable with this microcontroller so there isn't a lot of unused space.

The ATMEGA2560 has a 4Kx8 EEPROM. This is a large enough EEPROM to store all of the data our team should need for our program. Since our team would use a single microcontroller for this choice our microcontroller would need a large EEPROM to reduce the number of write cycles of the FLASH program memory. If during the development process our program storage exceeds the limit of the EEPROM it would be best to use a different microcontroller so the microcontroller doesn't have to rely on the FLASH program memory and ruin the FLASH cells.

13.4.8 RAM size

The RAM size in the tables above shows the size of the RAM each microcontroller has available. RAM is used in a microcontroller for storing temporary variables, the stack, and the heap. If a microcontroller is tasked with storing a lot of values, it's important to have a lot of RAM to give the microcontroller ample space to do so. Without sufficient RAM, the microcontroller may freeze, or crash entirely.

The MSP430FR6989 and the ATMEGA328p have a RAM size of 2Kx8. Since the MSP430FR6989 has similar specifications to the ATMEGA2560, this amount of RAM could be a problem. If our team uses too many peripherals with this microcontroller, it's possible that it will exceed the amount of RAM it has available and freeze. This would not be good for data collection, and if we don't take measures to inform the users of a crash, such as if our data collection was entirely headless, it could result in hours of wasted time for the users. This amount of RAM is fine for the ATMEGA328p, however. Our team would be using multiple microcontrollers to control sensors, since the ATMEGA328p has a small number of pins. Having the peripherals spread out over multiple microcontrollers would give each individual one plenty of RAM for each peripheral.

The ATMEGA2560 has a RAM size of 8Kx8. This is 4x as much as the other two options. Having a large amount of RAM is good for this microcontroller, due to the number of I/O pins it has available. Our team will be able to comfortably run all of our sensors from this one microcontroller without the risk of freezing or crashing.

13.4.9 Voltage

The voltage in the the tables above shows the operating voltage the microcontrollers are capable of running on. An ideal microcontroller should have a wide range of voltages it can operate between. The voltage will usually determine the

internal clock speed of the microcontroller. It also determines what kind of peripherals the microcontroller will be able to use. Choosing a microcontroller with the proper voltage range of our sensors will be an important step in the design process, since having a sensor with a voltage requirement outside of the microcontroller's operating voltage will render it useless.

The MSP430FR6989 runs on a voltage ranging from 1.8V to 3.6V. This is a power efficient microcontroller that has a lot of power saving options. One of these power saving options is an ultra-low power mode, which reduces this operating voltage even more, often into the millivolts. Using this option would give our solar panels time to recharge the batteries between data readings, and would overall be a low consumption option that would benefit our power system. This could be a problem if we choose sensors that require 5V to operate, however. We would need a separate power system in order to power these peripherals. This could be a problem for our power needs if our peripherals don't have the option of being powered through the microcontroller, since they wouldn't take advantage of the low power mode options, and would always be drawing power from the batteries.

The ATMEGA328p runs on a voltage ranging from 1.8V to 5.5V. This is a balance between the other two microcontrollers, and can operate on a voltage at both the minimum and maximum operating voltages of them. This would allow our team to power most peripherals from the microcontroller, which usually operate at either 3.3V or 5V. We could also take full advantage of any low power options this microcontroller has, and not put too much pressure on the power system.

The ATMEGA2560 runs on a voltage ranging from 4.5V to 5.5V. This microcontroller is the most restricted of the three options. Our team will have to make sure our power system is able to deliver a relatively stable power supply to this microcontroller to avoid issues with power. We will have to design a step down voltage regulator that provides a constant 5V output to make this possible. We will also have to make sure the vehicle can handle the power requirements, without relying too much on solar charging to keep our vehicle powered. This microcontroller will be able to power all of our peripherals, however, by using a step down voltage regulator on the microcontroller itself to provide a 3.3V output for the peripherals that can't operate on 5V.

13.4.10 Mounting Type

The mounting type in the tables above shows the type of mounting the microcontrollers use for attaching to PCB's. The mounting type of the microcontroller will determine how cheaply and easily our team will be able to prototype our PCB design. If the mounting type is not capable of inserting into a solderless breadboard, it will make prototyping much more difficult, and will require a development board in order to physically build the different designs our team will make. Another option is to simulate the microcontroller inside a program, however reliably doing so cannot be counted on, and our team will solely focus on whether or not we can physically test our microcontrollers.

The MSP430FR6989 and ATMEGA2560 both use a surface mount mounting type. This type of mounting type could cause trouble throughout the life of the

project. A surface mount device is unable to be prototyped and tested on a solderless breadboard. Our team would have to buy development board from the manufacturer in order to test circuits with these microcontrollers. This could lead to a lot of money spent on these options. A surface mount device is also structurally weaker than a through hole mounting type. Since our vehicle is expected to traverse rough terrain, the constant vibration might eventually break the surface mount device off of the PCB. This would lead to our team needing to either resolder the microcontroller back onto the PCB, which is difficult and requires special equipment, or buy and entirely new PCB with a new microcontroller attached.

The ATMEGA328p uses a through hole (D.I.P.) mounting type. This mounting type is preferred for automotive uses, due to the stronger bond it has with the PCB. This would be the better option if our team doesn't want to risk the microcontroller breaking off of the PCB during testing or demonstration. This mounting type also allows for easy prototyping with a solderless breadboard, due to the structure of the pins.

13.4.11 Number of I/O

The number of I/O in the tables above shows how many I/O pins each microcontroller has available. I/O pins are necessary for our microcontroller to connect with peripheral devices. These devices include all of the sensors our vehicle will have, and also the parent computer the microcontroller will send all of the data to. If our microcontroller does not have enough I/O pins, our team will not be able to use all of the sensors with a single microcontroller. Instead, we will have to rely on a serial communication method that allows multiple peripherals. This method will make our design extremely complicated, and depending on the microcontroller, could require complex code that our team may not have time to design.

The MSP430FR6989 has 83 I/O pins. This includes 12 analog pins. The amount of I/O pins this microcontroller has will be able to handle all of the peripherals our team has. This will not be a determining factor for this specific microcontroller, since there are restrictions for the other specifications. Our team will have to do testing to see if this microcontroller will be able to handle all of the peripherals, despite the large number of I/O pins available.

The ATMEGA328p has 23 I/O pins. This includes 6 analog pins. This is a small amount of I/O pins that will not be able to hold all of the peripherals our team has. This is not an issue for this microcontroller, since our team plans to use multiple microcontrollers if this option is chosen. This will need careful planning for choosing which peripherals will be paired together for each microcontroller, or having a separate microcontroller for each peripheral.

The ATMEGA2560 has 86 I/O pins. This includes 16 analog pins. This is the most amount of I/O pins out of all of the options, and will be enough for all of our peripherals. This microcontroller shouldn't have any other restrictions that would prevent our team from being able to connect all of our peripherals. If, through testing, this microcontroller is too slow to handle all of the peripherals, our team will be able to easily transfer the code to the ATMEGA328p, since they are similar microcontrollers that use the same IDE and code.

13.4.12 Cost

The cost in the tables above shows an estimated cost of the microcontrollers. The cost of the microcontroller is not the most important factor, however it may determine how much we can prototype with them. Our team is expecting to make mistakes while designing and testing circuits for our microcontroller. Because of this, it's possible that we will accidentally ruin a few microcontrollers while testing. We have to take this into consideration when choosing a microcontroller. We also have to account for development boards, which were not included in the tables. These development boards will be the only way our team can rapidly prototype with some of the microcontrollers, and the price of them widely vary depending on the manufacturer.

The MSP430FR6989 costs around \$10. Since this option needs the development board due to the mounting type, extra costs would need to be accounted for when considering this option. Each development board costs \$20 from TI, and if our team needs multiple boards for testing this cost will quickly add up. This option is a good balance between cost and function, and with the newer technology TI uses in its products this option could be the best choice if our team wants to use features that the other microcontrollers don't have.

The ATMEGA328p costs around \$2.50. Since our team will need multiple of this microcontroller, the cost will eventually go up to around the other options. However, for testing purposes and development only one would be needed. This would allow our team to distribute the cost over many weeks, instead of having to spend a lot of money upfront. This would be good if our team needs to change to a different microcontroller during the designing of the vehicle. This microcontroller also doesn't need a development board, and the cost of the microcontroller will only be spent on what we can actually use in the final build, and not just for testing purposes.

The ATMEGA2560 costs around \$14. This is the most expensive option, and has more costs associated with it, like the MSP430FR6989. Due to the mounting type of this option, a development board will be needed. The cost of the development board can be as high as \$40. Our team may be able to get around this cost by doing testing and development with the ATMEGA328p, and adapting the code to an ATMEGA2560 for the final build. This will need careful planning, however it is possible, and our team has tested this method with successful results.

13.5 Final Decision

After considering all the available options our team will use the ATMEGA2560 in our project. While the technologies implemented into this microcontroller aren't the newest, it is still a powerful microcontroller that will be capable of meeting our needs. The one issue our team faces with this choice is the surface mounting type. If this microcontroller vibrates loose during testing, we will have to overcome many challenges in order to create a working vehicle before the demonstration date. This could be solved by adapting the code to the ATMEGA328p, which will be used as a backup if the ATMEGA2560 fails at any points.

14 GSM Module

In this section, we will be discussing the GSM module that will be used for our design. The GSM module will allow our vehicle to be able to connect to a remote network which allow for internet communications in areas that do not have a WiFi network. You can liken this to phone data and being able to browse the web using that data. Since phone networks operate on a subscription-based service, this means that we need to pay to be able to use the service provided by these networks, such as T-Mobile or AT&T. Another expense that is required to be able to use the module is a SIM chip. The SIM chip is what identifies the device to the network. The modules that we will be considering are the SIM900 Quad Band GSM Module, XYGStudy GSM/GPRS Module, and Shield for Arduino with GPS - SIM7600CE . The aspects that we'll be reviewing before we decide on which to use will be: size, wireless technology, and price.

14.1 SIM900 Quad Band GSM Module

This SIM900 GSM module is based off the Sim900 from SIMCOM and allows us to connect to a GSM cellular network. The module allows us to send SMS, MMS, GPRS, Audio, and HTTP commands via UART. The frequency band that this device has is up to 1900 MHz which means this device only supports 2G connection. It also has Embedded TCP/UDP which allows us to send data to a web server.

14.2 XYGStudy GSM

The XYGStudy is a Raspberry Pi addon that is based on the SIM7600G from SIMCOM. As noted this is a Raspberry Pi addon which we will be using as our computing device. The Raspberry Pi will be maintaining our GUI and must be able to receive information constantly from the microcontroller. With this, we need to make sure that the Pi can maintain a stable connection so that the GUI does not loss connection and fail to display important info. That being said, this module allows 4G connection to be able to browse the Internet which is much more robust than 2G or 3G, not only that we're able to do HTTP requests which is essential for our GUI.

14.3 Shield for Arduino with GPS - SIM7600CE

The SIM7600CE Shield for Arduino is another GSM module like the first option except it supports 4G which has a faster and stable connection, which is essential for our GUI as we need a constant stable stream for our data. It shares the same aspects as the XYGStudy module but is designed for use for Arduino-based systems.

85mm*57mm*20mm	56.21mm × 65.15mm	69*54mm	Dimensions (mm)
2G	2G, 3G, 4G	2G, 3G, 4G	Wireless technology
3 V - 5 V	3.3/5V	3.3/5V	Power Requirements
\$21.90	\$86.99	\$65.00	Price
SIM900 Quad Band GSM Module	XYGStudy GSM	SIM7600CE Shield	

Table 19: GSM Comparison Table

14.4 Comparison

In comparison, the latter two modules are almost the same it's dependent on which one you connect to whether it be a Raspberry Pi or an Arduino device which we will be doing. This means that both the XYG and Shield for Arduino are good options to use. The first option would be good for a different application but since the GUI and the information it will receive are vital to the project we cannot use it due to its 2G protocol. Since 4G is more robust, we feel that this may work out better in the end to use. Between the XYGStudy and Arduino shield, each are designed to be used for different devices: Raspberry Pi and Arduino. Now this does not necessarily mean they will not work with other other devices. However, it is in our best interest to use these as they were intended.

14.5 Final Decision

For our final decision, since we only really need the Raspberry Pi to be connected to the Internet. We decided that to go with the XYGStudy GSM module for our final design. Firstly, the initial option would not work well for our purposes. 2G is essentially outdated at this point of writing and in order for the GUI to be more stable we felt that a 4G connection would work much better. Not only that, comparing the XYG and Arduino GSM Shield together, we felt that they were quite similar spec-wise, and since the Arduino will be connected to the Pi via serial communication, we felt that the Arduino shield would not be necessary in our design. Therefore, the XYGStudy module will be used for our design.

15 GPS Chip

For our design, we discussed earlier the features that it would require in order for it to be autonomous. Specifically with Geofencing to distinguish a designated navigable area, as well as use the GPS to make rapid navigational decisions based on data provided by the sensors. The way we can accomplish this is by utilizing a GPS sensor that can return information to be displayed through our GUI. There are plenty of good GPS modules in the market that could serve our design well, but

we need to choose one. For our design, we considered these features for the GPS module: size, update rate, navigation sensitivity, power requirements, accuracy, and price. With these requirements in mind, our team found that these modules: the NEO-6 GPS module, the Grove GPS module, and the Grove Air530 GPS module.

15.1 NEO-6 GPS Module

The NEO-6 GPS module is a stand-alone GPS receiver that sits at 12 by 16 by 2 mm. It is one of the most popular GPS modules on the market. It has a MS621FE-compatible rechargeable battery for backup and EEPROM to store configuration settings. This module is based off the u-blox NEO-6M GPS engine and has a ROM/FLASH version of ROM 7.0.3. It has a configurable UART interface for serial communication, with a default UART baud rate at 9600. Unlike most GPS modules, the GPS signal is right-hand circular polarized, which means that the electric field vector rotates clock-wise with respect to the direction of propagation. Because of this, the antenna provided will be a patch type rather than the traditional whip type. In terms of accuracy, the module that 2.5m GPS horizontal position accuracy, has a -161dBm navigational sensitivity rate, and updates at 1 Hz to 5 Hz max. One major thing to note, is that it does not include any header pins which means we must solder them in ourselves.

15.2 Grove GPS module

The Grove GPS Module is another stand-alone GPS receiver that sits at 40 by 20 by 13 mm. The module is armed with a SIM28 module which is the actual module that drives the GPS functions. Similar to the NEO-6 module, it allows for serial communication and has a default baud rate at 9600. If we were to use a Grove development board, we could plug directly through its UART interface. The antenna included is a traditional whip-type. The specs listed on the manufacturers website state that it has: a navigational sensitivity at -160dBm, a 2.5m GPS horizontal accuracy, and an update rate of 1 Hz - 10 Hz max.

15.3 Grove Air530 GPS Module

The final module in our consideration is the Grove Air530 GPS module. This module is described as being a "high-performance, highly integrated multi-mode satellite positioning and navigation module." Spec-wise it's almost similar to the previous module, however it has support for 6 satellites at a time which is useful for areas with bad-signal. One major feature of this module is that it has a much lower power consumption sitting at 31uA. In terms of accuracy it has the same horizontal accuracy as the previous, but this module features a high positioning accuracy of 3.5m a speed accuracy of 0.1m/s, and a time transfer accuracy of 30ns.

15.4 Comparison

The three GPS modules are quite similar overall as they all have the same baud rate, navigation sensitivity, mostly similar power requirements, and horizontal

positional accuracy. Below we'll be listing all the specification side-by-side and then determine which one to use.

12mm*16mm*2mm	40mm*20mm*13mm	40mm*20mm*13mm	Dimensions (mm)
1 Hz - 5 Hz	1 Hz - 10 Hz	1 Hz - 10 Hz	Update Rate
-161dBm	-160dBm	-160dBm	Navigation Sensitivity
3 V - 5 V	3.3/5V	3.3/5V	Power Requirements
2.5m GPS Horizontal	2.5m GPS Horizontal	2.5m GPS Horizontal	Accuracy
\$10.005	\$25.90	\$13.10	Price
NEO-6	Grove GPS	Grove Air530	

Table 20: GPS Comparison Table

15.5 Final Decision

For our final decision, we carefully considered the three options and found that they are all essentially the same. In terms of specs they all have nearly identical besides the sizing. At a glance, it might be obvious to choose the NEO-6 over the others due to its price point and sizing; however, the need for soldering can add to our costs in terms of soldering material as well the as the machine to do so. Therefore, we decided to not go with the NEO-6. Between the two Grove modules they are almost identical as well. The Grove Air, however, boasts up to six satellites connectivity to help compensate in areas with bad signal. Since our design is going to operate in remote areas with barely any connection, we felt that this module would work well for us.

16 Interface Computer - Raspberry Pi 4

This section will discuss the interface computer we will use for our vehicle. This interface computer will be the computer that runs the main GUI, and has an outside connection to a database. This database will store all of the sensor information on an external server. This computer will be the main hub of all of our computers, and will display all of the sensor and GPS data to the user. For this interface computer our team will be using the Raspberry Pi 4. The table below lists the technical specifications for the Raspberry Pi 4.

What the computer will do is host a server that will then upload the data onto a front-end GUI. The server will receive our data directly from a database that will

update periodically with the information being received from the MCU. Firstly, we would need a middle-man program to decipher that data from the MCU and then upload directly, as there is no current support to send data directly from an MCU to a database. Once that is done, we can design an API that will perform HTTP requests and take that data from the database and hold it onto a server. Finally, our front-end will also perform HTTP requests to receive that data from a server and display it as necessary.



Figure 34: Raspberry Pi 4B - Reprinted with permission from Adafruit

Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz	CPU
2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE Gigabit Ethernet	Connectivity
2GB, 4GB or 8GB LPDDR4-3200 SDRAM	Memory
2 × micro-HDMI ports (up to 4kp60 supported)	Display
75.00	Total Cost (\$)
Raspberry Pi 4B	

Table 21: Raspberry Pi 4B Specifications

Our team decided to use the Raspberry Pi 4 due to the convenience and power it offers. This computer is small enough to fit inside our vehicle, and is capable of

displaying to 2 different screens at 60FPS and up to 4K resolution. The Raspberry Pi 4 has a quad-core processor and 8GB of RAM, which offers enough power to act as the controller for our vehicle. This computer also has connectivity options that will allow it to connect to our external server and transfer the data the vehicle gathers to it.

17 Touchscreen Display

For the touchscreen display, we indicated different displays that could work with our design, each with different sizes and features. In order to decide which one would fit best for our design we considered four aspects. First, touchscreen capabilities. Second, the size of the display. Third, connectivity via interface connection type as well as data communication types such as I2C or SPI. Finally, the final thing we considered was the cost. The three displays we are considering are: The Elo TouchSystems 19-Inch Touchscreen Display, The Hannspree 21-Inch Multi-Touch Screen Monitor, The BuyDisplay 10 Inch Serial SPI I2C Module Display, and The Raspberry Pi 7-Inch LCD Screen.

These four displays are all good choices for our vehicle, but we need to consider a few things. First of all, we are looking for a display that is priced between \$50-\$120. Another aspect we need to factor in is sizing. Currently, we are leaning towards a scalable prototype in which we'll use a Power Wheels car to reduce our overall budget. Because of the size constraint given, we need to consider how large the user console would need to be to fit on a smaller system such as this. Not only that, we need to consider how we'll be connecting the display to the microcontroller. Currently, we are deciding between two options for this: having the display connect directly to the microcontroller to return an interactable GUI, or using a separate device that will communicate with the microcontroller and run our GUI.

The former option presents a few problems, first being that if we used the MCU to "write" onto the display, we would be limited in the types of images to show. Since it's GUI components that should receive data from sensors and display, the MCU would constantly have to "rewrite" the information. Not only that, we also have the issue of the memory size of the GUI. Right now if we used the MSP430FR6989 we would only have 2 KB of RAM and 128 kB of FRAM available to us, and simply "drawing" a 1024x600 black and white image takes 75KB of memory. A solution to this would be adding external memory RAM/FRAM but it would be slow to access, which would not be ideal for our case if we want information in real-time.

The latter option would be to use a separate device that would communicate with the MCU through UART and would be solely dedicated for running the GUI application. The benefit of this would be that it would be able to receive data from the MCU in real-time and not have to worry about memory restrictions as well. Also, since memory isn't an issue anymore we can make the GUI more extensive as well. Therefore, the best approach to host the GUI application would be through a separate device, which will be a Raspberry Pi. Below we'll discuss in depth which display would work best for our design.



Figure 35: Elo TouchSystems Touchscreen Display

17.1 Elo TouchSystems Touchscreen Display

The Elo TouchSystems Touchscreen Display is a 19 inch touchscreen with a 1280 x 1024 resolution and is used primarily for POS systems that you would find in retail stores or restaurants. The price of the device is listed for \$99.99. The overall build is durable which would work well with our design and would be able to withstand different environments. One issue with this display is the size. Since we are considering using a Power Wheels car, we have limited space on the dashboard to fit a display. This means that because of how large the Elo TouchSystems display is, we cannot use it in our design. The company does offer different models of touch screen monitors to use, but our team wanted to consider other displays to use.

17.2 Hannspree Multi-Touch Screen Monitor

The Hannspree Touchscreen 15.6 Inch Display is our second option and features a 1366 x 768 resolution, HDMI + VGA + DP inputs, and connectivity to separate devices such as Raspberry Pi. This feature would be useful for us as we having support for a separate computing device would give us the option to add more features to the center console. However, there are three issues with choosing this as our display. The price for this display is currently listed under \$329.71 which is way beyond our price point and that is under an Amazon listing. We couldn't even buy this display directly from the Manufacturer as they are based in Europe and do not seem to ship to North America. Finally, since we are working with a smaller build the size of the display will be too large to fit onto the dashboard of the vehicle. Therefore, we need to reconsider this option.



Figure 36: BuyDisplay Serial SPI I2C Module Display

17.3 BuyDisplay Serial SPI I2C Module Display

The third display our team is considering is the BuyDisplay Serial SPI I2C Module Display. Firstly, this display is around 10 inches in size, 1024x600 pixel resolution, and at least \$90.70 with extra features for an added price. This display already passes our size constraint for our design, and not only that if we do need to increase/decrease the size BuyDisplay offers different sizes of displays. The way this display connects through pin header or via a ZIF connector. It's also listed under the specifications that it has the ability to use with I2C and SPI which allows for I/O expansion if needed as well and direct writing via a microcontroller or separate device.

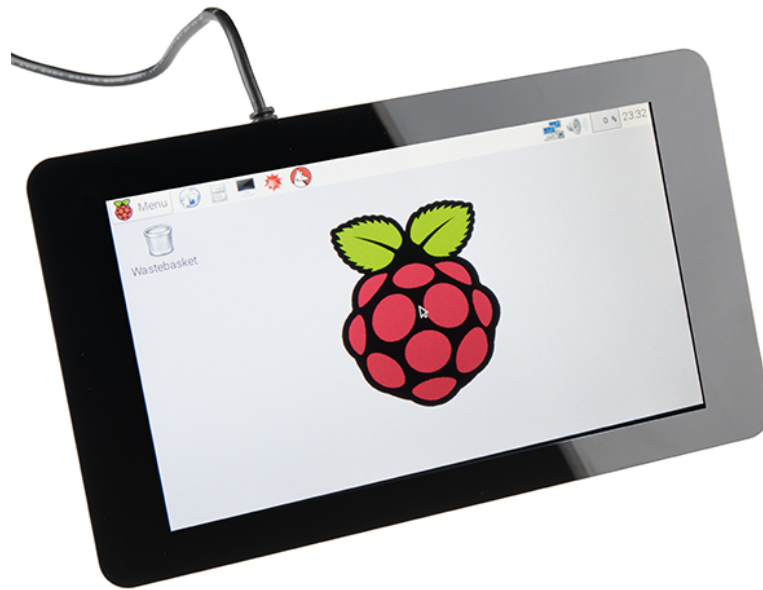


Figure 37: Raspberry Pi 7-Inch LCD Screen

17.4 Raspberry Pi LCD Screen

The final display our team is considering is the Raspberry Pi 7-Inch LCD Screen. This display is around 7 inches in size which is the smallest display out of the four, but is the cheapest only at \$60.00 and has the lowest resolution sitting at 800 x 480. This display is quite similar to the BuyDisplay as it connects via ribbon cable. What it differs from the BuyDisplay version is that it doesn't have the option to communicate via I2C and is meant primarily as a standalone display the connects to a Raspberry Pi. In a later section we discuss how we will get the information from the MCU to use in GUI. As a display, this could be a good consideration as it's price point and size profile would suit our needs.

17.5 Comparison

Comparing the displays above illustrated their strengths as well as their weaknesses. Below there is table comparing the factors that our team considered which includes: size, resolution, connectivity, and cost. Although there were more factors that would also matter, we felt that the listed aspects were more important at this stage of development.

19-Inches	15.6-Inches	10-Inches	7-Inches	Display Size
1280x1024 pixels	1366x768 pixels	1024x600 pixels	800x480 pixels	Display Resolution
Analog VGA	VGA, HDMI, and DP	Pin Header ZIF Connector	DSI Ribbon Cable	Display Connectivity
\$99.99	\$329.71	\$90.70	\$60.00	Total Cost (\$)
Elo TouchSystems Display	Hannspree Display	BuyDisplay Display	Raspberry Pi Display	

Table 22: Comparison Table for Touchscreen Displays

17.5.1 Display Size

This size section in the table above discusses the different sizes between the three displays and how it work with our design. As stated previously, we are highly considering the Power Wheels to act as a prototype of our ideal build. Because of this, we are limited in the size of different components we will have to use. This includes our touchscreen display, where if we were to choose a larger display, would not benefit the design but instead hinder it as it could obstruct the view of a passenger. Of course, a larger display would also be beneficial as we would be able to display more information overall. However, we need to understand that the dashboard of the prototype would be much smaller than the our ideal build. Therefore, our team would prioritize using a smaller display for the design.

17.5.2 Display Resolution

The resolution section in the table above compares the different resolutions between the three displays we are considering. This category is the least concerning between the four, as resolution would only really matter depending how far the dashboard would be from the passengers. The recommended resolution size for a UI according to official documentation written by Microsoft’s Windows App Developer Documentation [10], they recommend a minimum resolution of 800x600 pixels. With this in mind, comparing all the resolutions for the displays all but one exceed this minimum requirement. For the Raspberry Pi display, we feel that this would still work for our current design, as we are creating a smaller-scale prototype. Ideally for our final design we would follow the guidelines from Microsoft when developing the GUI, but for right now we won’t place a huge importance on this in our prototype.

17.5.3 Display Connectivity

This connectivity section is in relation to the row in the table above and compares the different displays we are considering and their connection types. The main reason why connection types matter is because it will influence overall visual

quality of the GUI application and even allow for extra options for I/O connection and data writing to separate devices. The first display utilizes a VGA connector which is currently outdated compared to the more common HDMI and more recent DP connection. That being said, it is still fine for our application since it doesn't need to be the maximum quality possible such as 1080p and beyond and we are limited by our resolution as well. As for connecting to separate computing devices, such as a Raspberry Pi or the like, we would need an adapter that converts micro HDMI to VGA which adds to our overall cost.

The second option has support for VGA, HDMI, and DP which is good as we can connect to different computing devices with or without the need for an adapter depending on the computing device. Again we would like to reiterate overall video quality is not important to the overall design, as long as the minimum quality of the display is 720p according to the documentation[10] it is acceptable.

The third option differs as it doesn't have traditional display inputs such as VGA or HDMI but instead uses a ZIF connector that drives the display directly through a microprocessor, which saves on overall space being beneficial for smaller applications. The only issue with this is that it might not be beneficial for larger versions of our design.

The final option, the Raspberry Pi, is similar to the third as it doesn't have traditional display inputs where it connects via a ribbon cable. The only downside to this one is that we're limited with connection as it does not have any I2C or SPI communication capabilities so we are only limited to using the Raspberry Pi as a means of communicating.

17.5.4 Total Cost

The total cost sections in the table shows the total cost of buying a display. As a team, we also needed to consider how expensive every component are design would be and needed to budget accordingly. Our team felt that the budget should be going to much more important components such as the AI computing device or vehicle. If we compare all our options we can easily see that the Hannspree Display is much more expensive that the rest and had reconsider. If we were creating the final larger design, we would most likely consider this one due to its size and features it had. Hannspree had other options for displays that would have worked for our design but the issue is that they EU-based and did not ship to other countries. The \$329.71 listing was available on Amazon which was why we were able to consider it in the first place.

This left us between three options: the EloSystems Touchscreen Display, the BuyDisplay Touchscreen Display, the Raspberry Pi Display. The three displays are all in the range of \$50-\$100. One thing to note is if we choose the EloSystems display, we also need to buy a VGA to microHDMI adapter if we wanted to use a Raspberry PI as our computing device. This extra cost actually makes it a bit more expensive that the BuyDisplay display. The Raspberry Pi display on the other is the cheapest option and includes all the necessary components we need to connect to a Pi device.

17.6 Final Decision

For our final decision, our team decided that the Raspberry Pi Display module would work the best. The reason for this is that it passed most of the requirements our team needed for our prototype as well as being the cheapest option available. The only issue would be scaling up our project if we weren't doing a prototype, we would have used one of the other options. Therefore, we will be going with the Raspberry Pi Display for our vehicle.

18 AI Computing Device

This section will discuss the AI computing device that will be implemented into the project. It will discuss the options we have for AI computing devices, the comparison of the different options, and the AI computing device our team will chose to implement into the project. Our team must consider the right AI computing decide that meets the needs of the project goals. Failing to do so could cause power issues, or cause the vehicle to injure a bystander or passenger.

The AI computing device controls the computer vision part of the project. It will identify objects in front of the vehicle, and determine whether or not the vehicle should continue on the planned path. If it chooses not to continue on the planned path it will then reroute the vehicle around the object and try to continue on the planned path after doing do.

The first option is the NVIDIA Jetson TX2 NX [11] AI computing device. This option was chosen for the power it has to offer. The Jetson TX2 NX is one of the best AI machines for non-professional use. It has roughly 2.5x the amount of processing power as the Jetson Nano and Compute Module 4. This would allow the vehicle to quickly identify any obstacles and plan the vehicles course of action much faster than the other two options.

The second option is the NVIDIA Jetson Nano AI computing device. This option was chosen for the amount of graphical processing power it has for the low price. This board is similar to the Compute Module 4, however the GPU is much better, and our project relies on a good GPU for quick graphics processing.

The final option is the Raspberry Pi Compute Module 4 IO board. This option was chosen for the low cost. This option is half the price of the Jetson Nano, and a quarter of the price of the Jetson TX2 NX. If our team needs to cut costs at any time during this project, this option will allow us to do that.

18.1 NVIDIA Jetson TX2 NX

The first consideration for the AI machine is the NVIDIA Jetson TX2 NX. The Jetson TX2 NX is a top of the line AI machine that would allow our team to create complex computer vision software that could handle most things it would be tasked with. Implementing this option into our project would also allow us to surpass the requirements listed and make our vehicle capable of driving faster, due to the faster object recognition available with this option.

The cost of this option is \$200, and the most expensive option of the three. This cost is within our budget, and is a reasonable cost for this machine. The one issue with this option is that the module does not come with a carrier board. The

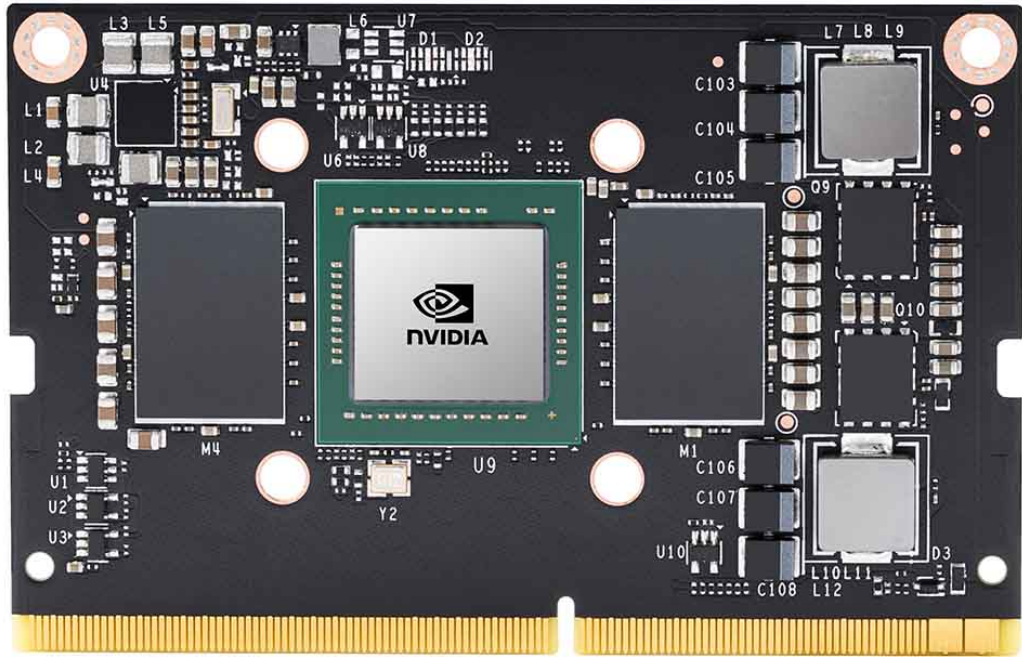


Figure 38: Jetson TX2 NX Module - Reprinted with permission from NVIDIA

1.33 TFLOPs	AI Performance
Dual-Core NVIDIA Denver 2 64-Bit CPU and Quad-Core Arm® Cortex®-A57 MPCore @ 1.4 GHz	CPU
256-core NVIDIA Pascal™ GPU	GPU
4 GB 128-bit LPDDR4 51.2GB/s	Memory
2 multi-mode DP 1.2/eDP 1.4/HDMI 2.0	Display
200.00	Total Cost (\$)
Jetson TX2 NX	

Table 23: NVIDIA Jetson TX2 NX Specifications

carrier board allows external interfacing with the module, such as USB connection, and HDMI output. A carrier board can be bought, or created by our team. If

the carrier board was purchased, there are two options available. The first option is the Jetson Nano carrier board, and the second option is the Xavier NX carrier board. If we purchased the Jetson Nano carrier board, it would require purchasing a Jetson Nano Development Kit for \$99, and will not fully interface with the Jetson TX2 NX since the pinouts are slightly different. If we purchased the Xavier NX carrier board, it would require purchasing a Xavier NX Development Kit for \$400, however this option will fully interface with the Jetson TX2 NX since the pinouts are identical. Purchasing a Xavier Development Kit would also allow us to make up the cost by selling the Xavier NX module. The cost for building the carrier board is unknown, and our team likely doesn't have the experience or time to build one.

These factors make it hard to consider this option, since it's not certain if the module will even be viable due to the lack of a carrier board. Purchasing a carrier board would be the easiest option for our team, and it would give us time to work on other parts of the project, without worrying about failure if we were to build the carrier board. As the project progresses it will become more clear whether or not we need the Jetson TX2 NX, or if a less powerful module will be the better choice.

18.2 NVIDIA Jetson Nano

The second consideration for the AI machine is the NVIDIA Jetson Nano. This module is a lower end budget AI machine. Even though it's not as competitive as the Jetson TX2 NX, it is still a strong AI machine that has a good GPU for computer vision. The Jetson Nano will allow us to comfortably meet our requirement specifications and still allow for some extra features if necessary.

472 GFLOPs	AI Performance
Quad-Core Arm® Cortex®-A57 MPCore @ 1.42 GHz	CPU
128-core NVIDIA Maxwell™ GPU	GPU
4 GB 64-bit LPDDR4 25.6GB/s	Memory
2 multi-mode DP 1.2/eDP 1.4/HDMI 2.0	Display
100.00	Total Cost (\$)
Jetson Nano	

Table 24: NVIDIA Jetson Nano Specifications



Figure 39: Jetson Nano Development Kit - Reprinted with permission from NVIDIA

The cost of this option is \$100 and is in the middle price range of all three options. The cost is well within our budget, and will allow us to spend more money on other parts of the project. This module also comes with a carrier board, unlike the Jetson TX2 NX. The addition of the carrier board means our team can focus on making the software as optimized as possible without worrying about losing time building a carrier board. We also don't have to worry about spending any extra money building a carrier board, and risking the board failing.

This option is on the top of our list of considerations. The only worry is that our team won't be able to purchase one due to the popularity of the module. The Jetson Nano is sold in a lot of online stores, however the stock is limited, and it often takes weeks for the stock to return.

18.3 Raspberry Pi Compute Module 4

The third consideration for the AI machine is the Raspberry Pi Compute Module 4. This is a single-board computer that can be used for embedded applications. This option does not come with a GPU, and will only be considered if the other options are not available for purchase. This computer is good in other aspects though, such as a processor that can compete with the Jetson Nano. This option will allow us to meet some of the requirements needed for our project, and will allow our team to show a proof of concept, rather than a fully functional prototype.

The cost of this option is \$33, making this the lowest cost option of the three we have. This cost is below our budget, leaving plenty of extra money to spend on other parts of our project. There are other costs to this option, however, such as an IO board needed to interface with external components. This IO board

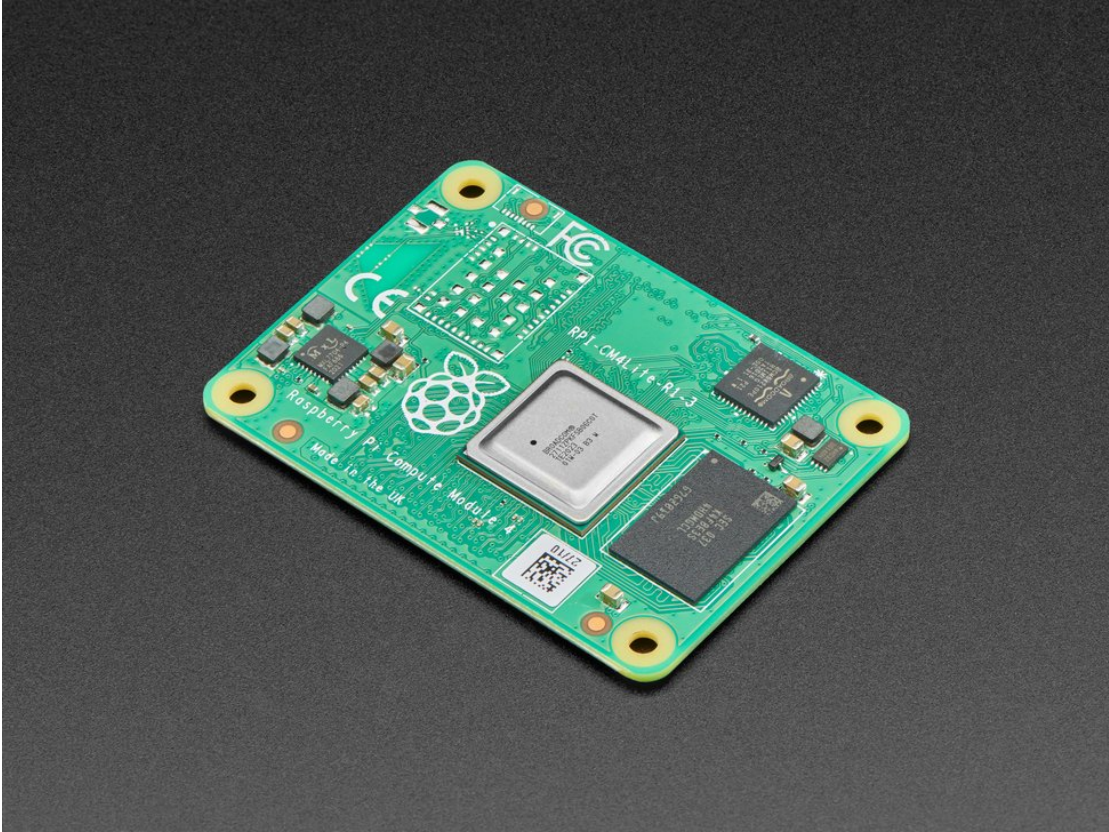


Figure 40: Raspberry Pi Compute Module 4 - Reprinted with permission from Adafruit

N/A	AI Performance
Broadcom BCM2711 quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz	CPU
N/A	GPU
1GB, 2GB, 4GB or 8GB LPDDR4	Memory
Dual HDMI interface	Display
71.00	Total Cost (\$)
Compute Module 4	

Table 25: Raspberry Pi Compute Module 4 Specifications

functions similarly to the carrier board the Jetson TX2 NX and the Jetson Nano require. The cost of the IO board is an additional \$38. The additional cost brings this option to a total of \$71 dollars. The added costs are still \$30 below the Jetson Nano, making this option a good choice if we need to spend more of our budget on other parts of our project.

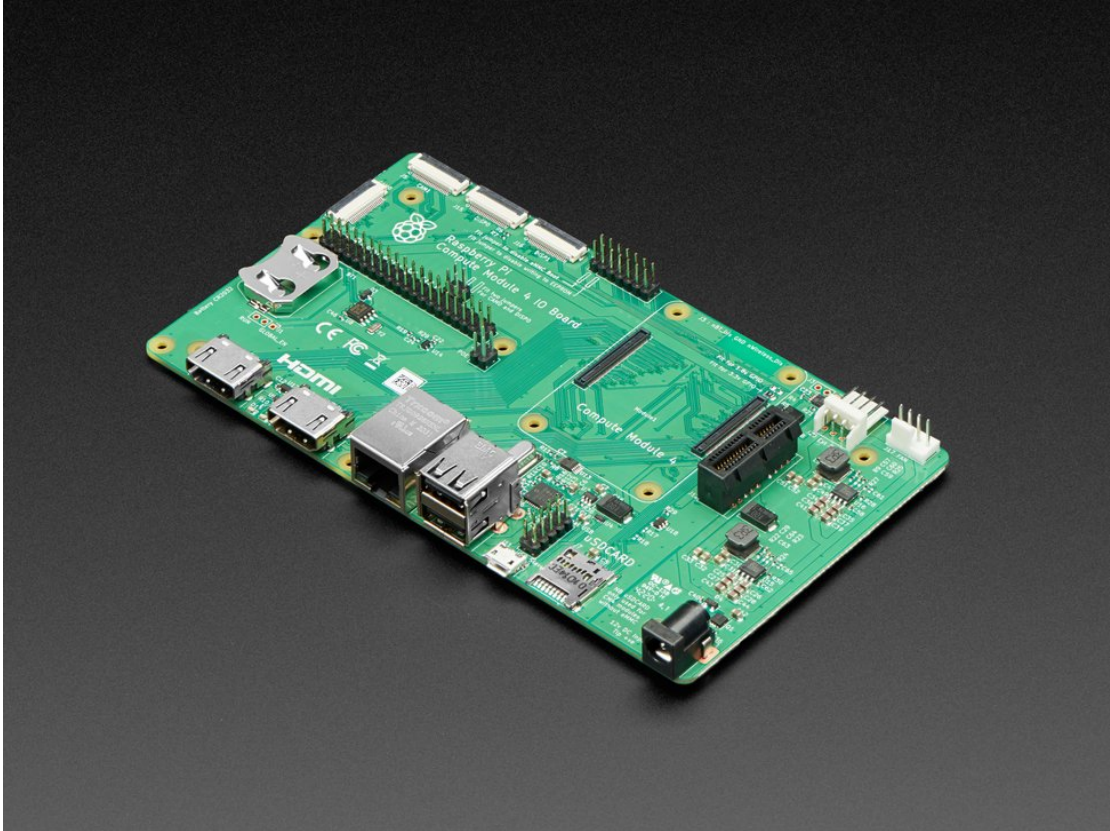


Figure 41: Raspberry Pi Compute Module 4 IO Board

This option is being considered as a backup choice if the other two options aren't available for purchase. If we choose this option it will severely hinder our ability to produce a functional prototype, and our team will have to take steps to keep our project goals obtainable.

18.4 Comparison

Comparing the different options shows us a lot about the strengths and weaknesses of the modules. The specifications above will be used as a comparison between the different modules. The factors our team is considering that are listed in the tables include: AI performance, CPU, GPU, Memory, Display, and total cost. The GPU is what dictates the AI performance, however they'll be compared separately for a broad overview of the specifications and limitations of the different options. A comparison of these factors will be discussed below.

18.4.1 AI Performance

The AI performance section in the tables above shows the speed at which each module will be able to compute. AI performance is the measurement of calcula-

tions the can be done per second. The measurement unit is a FLOP, or floating point operations per second. This will be an important factor in choosing an AI machine, since our team will need the AI machine to quickly recognize objects in the path of the vehicle.

The option with the best AI performance is the Jetson TX2 NX. The Jetson TX2 NX is roughly 2.5x as powerful as the Jetson Nano, with 1.33 TFLOPs. This will allow for on average 25 frames per second for object detection. While driving at a max autonomous driving speed speed of 8kph, or 2.22mps, this will allow the Jetson TX2 NX to take 11 samples per meter. The sample rate should allow the detection of any fast moving objects that move in front of the vehicle.

The option with the second best AI performance is the Jetson Nano. The Jetson Nano has an AI performance of 472 GFLOPs. This will allow for on average 13 frames per second for object detection. While driving at a max autonomous driving speed speed of 8kph, or 2.22mps, this will allow the Jetson Nano to take 5 samples per meter. The sample rate may not be enough to detect fast moving objects that move into the vehicles path, however, it should allow the vehicle to detect any objects already in front of the vehicle.

The last option, the Compute Module 4, doesn't have a GPU on the module, so there is not a measurable AI performance available for this option. Since this option does not have a GPU, there is not reliable measurement for how fast the Compute Module 4 would be able to detect objects, or even how many times per second it could update.

18.4.2 CPU

The CPU section in the tables above shows which CPU each option has, and the speed of the CPU. The CPU is an important part of machine learning, since there is a lot of computation needed in order to process the data. Processors that are faster with more cores will allow for more processing able to be done per second, and will allow for multiple processes to be done simultaneously.

The first option with the best CPU is the the Jetson TX2 NX, has a four core Cortex A57 processor and a two core Denver 2 processor. The Denver 2 processor is what sets the Jetson TX2 NX apart from the other two options. Since the Cortex A57 processor of the Jetson TX2 NX and Jetson Nano and the Cortex A72 processor of the Compute Module 4 are relatively equal in power, the Denver 2 allows the Jetson TX2 NX to out perform the other two options. With the ability to split tasks between the two processors of the Jetson TX2 NX, depending on the application, this can have a huge impact on performance.

The second option, the Jetson Nano, has a four core Cortex A57 processor. This processor has a nearly identical performance to the Compute Module 4's Cortex A72 processor. Without the added benefit of the Denver 2 processor that the Jetson TX2 NX has, the performance between the Jetson Nano and the Compute Module 4 are going to be about the same.

The third option, the Compute Module 4, has a Cortex A72 processor. As mentioned in the previous paragraph that discussed the Jetson Nano, the Compute Module 4 and the Jetson Nano have processors that are nearly identical in performance.

18.4.3 GPU

The GPU section in the tables above show which GPU each option has. As discussed in the AI performance section, the GPU is an important part of the AI machine. Without a GPU, there can be no significant graphical processing done, and will have little to no capability of object detection.

Most of the important factors of the GPU are discussed in the AI performance section. However this section will further add some details about the GPU, and show why the AI performance is impacted by the GPU.

The first option, the Jetson TX2 NX, has a 256 core NVIDIA Pascal GPU. The Pascal architecture is the newer module of the Maxwell architecture. The result of this is minimal, however it's worthwhile to note that the manufacturing process is different, and the Pascal GPUs were made using a 16nm process. The Pascal GPU is also more energy efficient. The 256 cores allow the Pascal GPU to perform better than the Maxwell GPU of the Jetson Nano, which only has 128 cores. The average expected performance of the Pascal GPU is about 2.5x better than the Maxwell GPU.

The second option, the Jetson Nano, has a 128 core NVIDIA Maxwell GPU. As discussed in the previous paragraph, the Maxwell architecture is the predecessor to the Pascal architecture. While the newer Maxwell architecture was manufactured using a 16nm process, the older Pascal architecture was manufactured using a 28nm process. Since this option is using a less powerful, and older, GPU it should be expected that the capabilities will be limited. Our team will have to decide whether or not this option is viable with our current requirements.

The third option, the Compute Module 4, does not have a GPU. This option might not be possible to use since it is unknown how well it will perform with object recognition. It is possible however that our vehicle will not need to rely so heavily on object recognition, and can instead use various sensors, such as a proximity sensor, to aid in the process of detecting any objects in the vehicle's path. These factors make this option worth consideration despite not having a GPU.

18.4.4 Memory

The memory section in the tables above shows the amount of RAM each option has. The amount of RAM our AI machine has will determine how much data it can store. If our AI machine is running a lot of processes for object detecting, it may run out of available RAM, and this could lead to a system freeze, or failure. If this happens at any time during the operation of the vehicle, it would compromise the safety of anyone around, since the vehicle would essentially be blind. A system failure could also cascade to the other systems in our vehicle, and cause a crash.

The first option, the Jetson TX2 NX, has 4GB of 128-bit LPDDR4 RAM. This RAM has a read/write rate of 51.2GB/s. Since this is a 128-bit RAM, the data transfer rate is exactly twice as fast as the Jetson Nano. This could be an important factor, since our vehicle will need to react quickly, and will need the ability to read and write data quickly to do so.

The second option, the Jetson Nano, has 4GB of 64-bit LPDDR4 RAM. This RAM has a read/write rate of 25.6GB/s. Depending on the vehicle decision, this option could be the most beneficial. It is currently unclear how fast our final build

will travel, and if a slower vehicle is chosen it would be unnecessary for the AI machine to be the fastest option.

The last option, the Compute Module 4, has variable RAM depending on the model. It can have 1GB, 2GB, 4GB, or 8GB of LPDDR4 RAM. This option did not specify the read/write rate of the RAM. This is a risky design decision since the read and write speed of the RAM is important. Unless further research on this option leads to a read/write rate of this RAM, it may be taken out of consideration for the AI machine.

18.4.5 Display

The display section in the tables above shows the options the AI machines have for connecting to an external display. All options have the option of an HDMI display option, with the Compute Module 4 having 2 HDMI ports. The Jetson TX2 NX and the Jetson Nano also have a eDP option, which allows for multiple displays to be used through one cable. If our team needs to have multiple displays active at the same time, it would be worth having the eDP option. Some versions of eDP also have a higher bandwidth than HDMI.

If our team plans to have multiple displays on our vehicle, having a Jetson TX2 NX or a Jetson Nano would be best to choose since they both have the option of using an eDP connection. The Jetson TX2 NX would be able to have up to 5 displays on a single eDP port. This would be ideal if our team chooses to use multiple displays to display sensor information, gps mapping, and object recognition on different areas of the vehicle instead of using a single display for all of the information. The Jetson Nano would be able to have up to 2 displays on a single eDP port. This would allow for a display to show the front camera on the vehicle and display the object recognition software, while having another screen display gps mapping and sensor data.

If only one display is needed for our vehicle, the best option would depend on the vehicle. If our team chooses the Power Wheels vehicle, it wouldn't be a good choice to use the higher powered AI machines, such as the Jetson TX2 NX, or the Jetson Nano. Instead, the Compute Module 4 would be able to control the object recognition, since the Power Wheels is the slowest option. However if our team chooses a single display and one of the faster vehicles, it would be better to choose the higher powered options.

18.4.6 Total Cost

The total cost section in the tables above shows the total cost of buying the AI machine for each option. As with every part of this project, cost is an important factor when determining which AI machine to choose. Our team has a limited budget, and we will not be able to meet the goals of this project without managing every dollar we spend, and cutting costs where able.

The cost of the options are relative to their performance. This is to be expected with electronics. If our team needs an AI machine that has a lot of power, we are going to have to spend more money, which will impact the quality of the other parts we need. It would also be an unwise choice to pick a high powered AI machine and not have the budget to buy sensors capable of matching its performance, or having to buy a slower vehicle that cannot utilize the high GPU processing.

The option with the highest cost is the Jetson TX2 NX at \$200. This option will be best if our team decides to choose the retrofitting option for our vehicle. We will need the extra processing power from the GPU in order to quickly and accurately identify any objects that move in front of the vehicle while it's travelling at its max speed of 8kph. The extra cost of this option paired with the high cost of the vehicle would mean we would have to take a portion of the budget from either the sensors or the solar panel in order to stay within our total budget.

The option with the second highest cost is the Jetson Nano at \$100. This option will be best if our team decides to choose either the retrofitting option or the hand-built option for our vehicle. If we choose this option we may have to redefine the maximum autonomous speed the vehicles can drive at, however. With this option it will take roughly 2.5x as long as the Jetson TX2 NX to identify objects. This option would be best to choose if we don't want to reallocate the budget for other parts but still stay within our total budget.

The option with the lowest cost is the Compute Module 4 at \$71. This option would be best if our team decides to choose the Power Wheels option for the vehicle. A Power Wheels vehicle is limited to 8kph, and we would have to lower that even further for autonomous driving to give the vehicle ample time to stop if needed. One of the more powerful, and more expensive, options would be wasted on a Power Wheels, and would take away from budget we could use elsewhere. This option would be best if we need to buy more expensive parts such as sensors, batteries, motors, or solar panels, since we would be able to use some of the AI machine budget for those purposes.

18.5 Final Decision

After considering all the available options our team will use the NVIDIA Jetson Nano in our project. The AI performance of the Jetson Nano is capable of fulfilling the requirements, and while it may not be as powerful as the Jetson TX2 NX, it won't necessitate redefining our requirements. This choice may impact safety since the Jetson Nano may not be able to identify objects that quickly move into the path of the vehicle, however we will have backup measures that will work with the AI machine to prevent injuries, such as a proximity sensor placed in the front of the vehicle. This Jetson Nano is also the best option for budget concerns. With this option we will have money left over that we can distribute between other higher cost parts. Our team has decided to only use the AI machine to run object detection software, so the CPU and memory limitations shouldn't impact the performance of the software, and there won't be a risk of freezing or crashing the Jetson Nano. We will also only be using one display in our vehicle, and the Jetson Nano will be able to meet this need.

19 Sensors

Our vehicle takes inspiration from other sensor heavy vehicles such as the Mars Rover and modern day electric cars. As such, our vehicle will be equipped with many different kinds of sensors. These sensors will gather measurement and information of the surrounding area and that data will be shown on the vehicles display. Some of this information is to gather data, and similar to how modern

day cars do, show the occupants a real time measurement of the data. Others will have a specific function such as the light sensor and the proximity sensor.

19.1 Camera - Kinect

Our vehicle will have an autopilot feature as well as be able to detect objects, as such, we will need the vehicle to be equipped with a camera. We decided to go for the kinect sensor as our camera because we would not have to order one since we have one already. The kinect sensor has an infrared camera, a normal RGB camera, an infrared projector as well as some microphones. We will only be using the RGB camera for the auto pilot and object detection.



Figure 42: Xbox 360 Kinect Sensor

19.2 NFC Sensor

NFC stands for Near-Field-Communication. It is a form of wireless communication that allows devices that are very close together to communicate and send data between one another. NFC devices are separated into two different categories, passive NFC and active NFC. A passive NFC device is one that does not require any kind of power source to send data although this also means that it is unable to receive data as well. An active NFC device is one that is able to send and receive data to other NFC devices.

19.2.1 DFR0231-H

Developed by DFRobot, the DFR0231-H is an NFC sensor that is based around the NXP PN532 which is an integrated circuit dedicated for NFC. This module supports both I2C as well as UART and can be toggled between the two with a simple flick of a switch. The DFR0231-H has three different modes that it can use: a reader/writer mode for reading and writing to tags and other similar devices; a card emulation mode when using things like smartphone pay; as well as a peer to peer mode for exchanging data between devices.

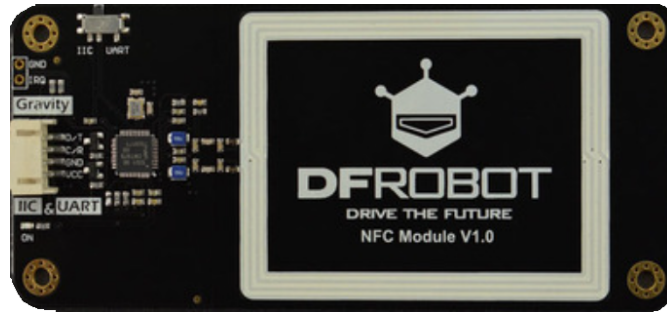


Figure 43: DFRobot DFR0231-H NFC Module

The maximum distance that the antenna can be away from another device is 10mm (1cm) and it has a communication frequency of 13.56MHz (mega hertz).

19.2.2 Grove NFC

The Grove NFC module is intended for the Seeedstudio Grove but is capable of communicating with other devices. It has the NFC antenna connected to the grove module using a small IPX cable. The Grove NFC module is based around the NXP PN532 integrated circuit which are dedicated for NFC. It has two different modes: a reader/writer mode for reading and writing to tags and other similar devices; as well as a peer to peer mode for exchanging data between devices. Can also communicate using either UART or I2C. The maximum distance that the antenna can be away from another device is 28mm (2.8cm) and has a communication frequency of 13.56MHz (mega hertz).

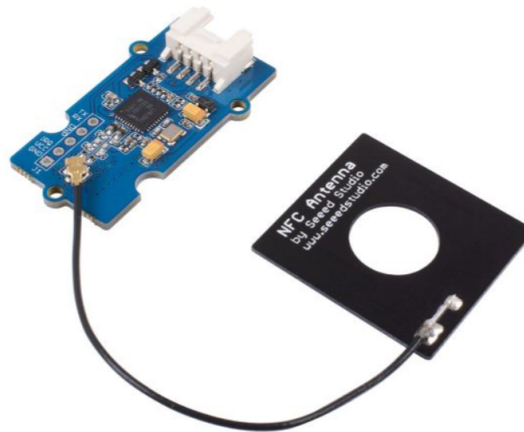


Figure 44: SeeedStudio Grove NFC Module

19.2.3 Comparison and Decision

3.3V to 5.5V	3.3V	Working Voltage
I2C / UART	I2C / UART	Communication
10mm (1cm)	28mm (2.8cm)	Max Communication Distance
13.56MHz	13.56MHz	Communication Frequency
Read/Write/Card/P2P	Read/Write/P2P	Protocol Support
110×50 mm	25.43mm x 20.35mm	Dimension
\$29.90	\$23.70	Price
DFR0231-H	Grove NFC	

Table 26: NFC Sensor Comparison Table

Both the DFR0231-H and the Grove NFC both run off of the NXP PN532 integrated circuit so in terms of max capability, both sensor are equal. That said, both sensors dont utilize the PN532 to its full potential, an example being the lack of an SPI interface even though the PN532 has pins for it.

With the Grove NFC, you dont get the card emulation that is available with the DFR0231-H, so the grove is unable to be used for applications where purchasing will be needed. The DFR0231-H needs the device to be at a slightly closer range, at 1cm as a opposed to 2.8cm for the Grove NFC. One very important difference between the two is that the Grove NFC has the antenna attached using a cable. So there is more freedom as to where the antenna can be placed on our vehicle.

Our team has decided to go with the Grove NFC module. It capable of doing what we need which is to have our phones connect with the module using NFC to determine if the user is authorized to unlock the door. On top of that it sits a slightly lower price point.

19.3 Temperature Sensor

A temperature sensor is used to determine the temperature of the environment around it. There are different ways to determine temperature such as reading the voltage level across diode terminals, using a thermistor which changes the amount of resistance based on the temperature which can then be read using an ohmmeter, or measuring the vibration in a metal wire.

19.3.1 Adafruit MCP9808

Adafruit produces multiple kinds of temperature sensor modules, one of which focuses around the MCP9808 temperature sensor. It is an eight pin sensor and the module itself is also eight pins so there is no extra functionality minus a voltage regulator for the input voltage. The pins on the module are: the input voltage (V_{in}), ground pin (GND), I2C communication pins (SDA and SCL), an interrupt pin if the temperature goes above or below a certain amount (Alert), and 3 pins for setting the I2C address (A0-A2). The three address pins correspond to the least significant bits of the address with A0 representing the lowest of three bits.

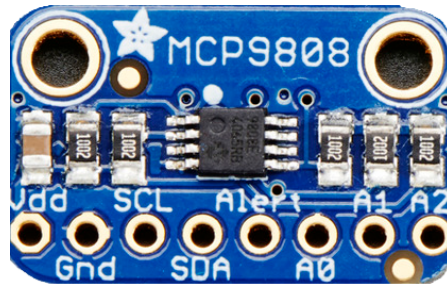


Figure 45: MCP9808

The MCP9808 can measure temperature from as low as -40 degrees Celsius up to 125 degrees Celsius. Adafruit produces another version of the MCP9808 that includes the STEMMA QT connectors on the side but since not all of our sensor will have this kind of connection available, we will not be considering that particular version of the MCP9808.

19.3.2 SEN0206

The SEN0206 is an infrared temperature sensor that is developed by DFRobot. When the lens collects infrared radiation energy, it is converted to electrical signals that are processed which then are converted to a temperature value. The module communicates with I2C and as such has the SDA and SCL pins, along side the V_{cc} (voltage input) and GND (ground) pins. It can provide a value for temperature ranges between -70.01 degrees Celsius to 382.19 degree Celsius.

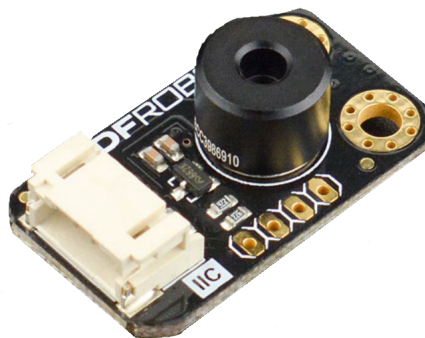


Figure 46: SEN0206

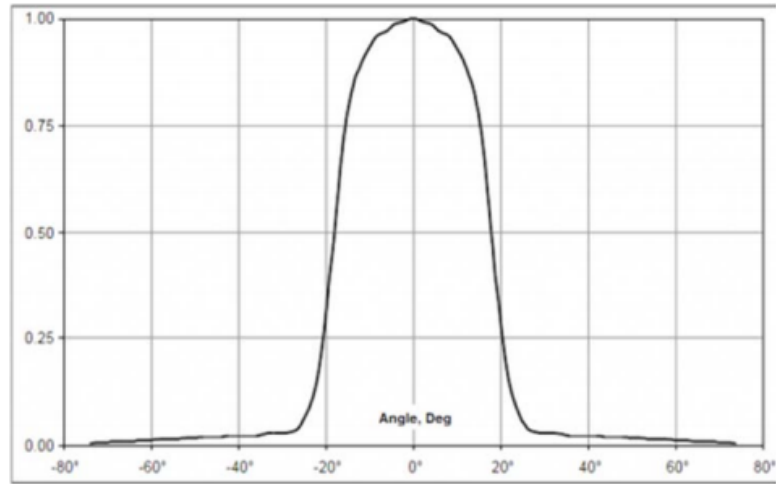


Figure 47: SEN0206 Field of View Graph

In order to ensure accurate measurements with the SEN0206, we must make sure that the object we are trying to measure the temperature of is within the sensor Field of View. From the graph above, we can see that a small change in the angle can cause a large change in accuracy with the temperature value.

19.3.3 DS18B20

The DS18B20 temperature sensor is a small three pin sensor developed by Maxim. You can buy just the sensor itself but we will be looking at the waterproof version. It has wires connected to each of the three pins of the DS18B20, and places the sensor inside of a metal tube casing and the wires are 36" (inches) long. The open end of the tube sealed shut using heat shrink. It can measure temperature ranges between -55 degrees Celsius up to 125 degrees Celsius. As mentioned earlier it is a three pin sensor: one pin for voltage input (Vcc), one pin for ground (GND), and a digital pin. That digital pin is what Maxim calls the 1-Wire Bus System.

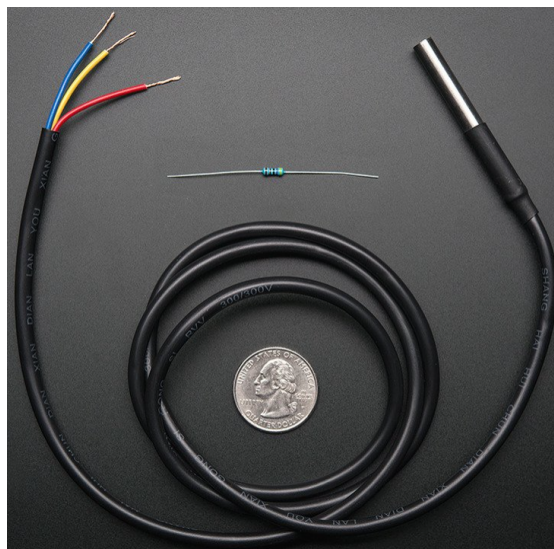


Figure 48: Waterproof DS18B20

19.3.4 AM2320

The AM2320 is a temperature and humidity sensor in one small package. It has 4 pins and uses I2C as its communications interface. Its range of temperature measurements are from -40 degrees Celsius to 80 degrees Celsius, with an accuracy between +/- 0.5 degrees Celsius.

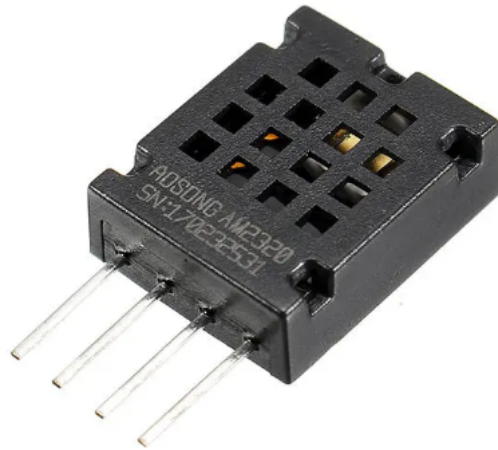


Figure 49: AM2320

19.3.5 Comparison and Decision

In this section we will be comparing the three temperature sensors mentioned above and make a final decision as to which sensor will be used.

2.5V to 3.6V	3.3V to 5V	3V to 5V	3.1V to 5V	Working Voltage
I2C	I2C	Digital	I2C	Communication
125 degrees Celsius	382.19 degrees Celsius	125 degrees Celsius	80 degrees Celsius	Max Range
-40 degrees Celsius	-70.01 degrees Celsius	-55 degrees Celsius	-40 degrees Celsius	Min Range
16.6*16.5*4.0mm	17.78*25.4*10.6mm	36" long wires	22.5*12*4.7mm	Dimension
\$4.95	\$16.00	\$9.95	\$3.95	Price
MCP9808	SEN0206	DS18B20	AM2320	

Table 27: Temperature Sensor Comparison Table

Both the MCP9808, DS18B20 and the AM2320 can measure almost the same minimum range of temperature, with the latter being able to measure an extra

-15 degrees Celsius compared to the former. The AM2320 cannot measure temperatures as high as the other three sensors. The SEN0206 on the other hand has a much wider range of values that it can measure, with its maximum temperature measurement being more than three times hotter than both the MCP9808 and the DS18B20. Although the SEN0206 has a wider range of values, its is not necessary to have such a wide range for the purpose of our vehicle. The 125 degrees Celsius of the MCP9808 and the DS18B20 is 257 degrees Fahrenheit which is a temperature that our vehicle will likely not ever experience. It would be much wiser to save a few dollars and stick to the MCP9808, DS18B20, or AM2320.

It terms of communication, the only sensor that is not I2C is the DS18B20; instead it uses a 1-Wire Bus System that is developed by its manufacturer Maxim. This one wire system is similar to I2C in that it allows multiple devices to be connected to a single bus, but its main drawback is that it is much more difficult to parse out the communication. I2C is more well known, has more examples to base off as well as standards to be followed.

Our team has decided to go with the AM2320 which offers a sufficient temperature range while also being the most affordable of the four. Although it is not waterproof like the DS18B20, since the top speed of our vehicle will be in the single digit range, there isnt as much threat of water reaching the module in the first place. The AM2320 also doubling as a humidity sensor is the main reason we chose it as we would not need a separate sensor to measure relative humidity.

19.4 Humidity Sensor

A humidity sensor is a sensor that measures the relative humidity surrounding it. The way that sensors work is by detecting changes that may cause change in the electrical current. Most humidity sensors fall into one of the three types: thermal, capacitive, and resistive. Our vehicle will be equipped with one of these sensor to allow accurate measurements to be displayed on the center console.

19.4.1 AM2320

As mentioned earlier in section 16.3.4, the AM2320 is a temperature sensor and a humidity sensor. For measuring humidity, it has a range between 0% rH (relative humidity) and 99.9% rH, with an accuracy between +/- 3% rH. It comes in a very small, four pin package and uses I2C for its communication. An image of the sensor is shown in section 16.3.4.

19.4.2 HTS221

Manufactured by Adafruit, the HTS221 is a humidity sensor that also operates as a temperature sensor as well. As with many other Adafruit boards, the the HTS221 is equipped with the STEMMA QT connections for an easy I2C connection. It also has 7 pins: the voltage input (V_{in}), ground (GND), a 3.3 voltage output pin ($3V_o$), SCL and SDA for I2C communications, a data ready pin (DRDY), and a chip select pin (CS). The SCL and SDA pins can also be used for SPI communications in conjunction with the CS pin.

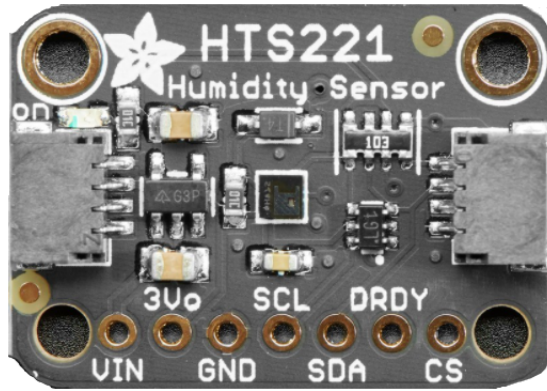


Figure 50: Adafruit HTS221

The HTS221 can measure from 0% to 100% rH (relative Humidity) with the temperature sensor being able to measure temperature from 15 degrees Celsius to 40 degrees Celsius. Compared to the temperature sensors from section 16.3, the HTS221 has a much smaller range of measurements and would only be useful as a temperature sensor in warm climates. Between 0% to 100% the humidity sensor is accurate to +/- 5% and between 20% to 80% it is accurate to +/- 3.5%.

19.4.3 Comparison and Decision

In this section we will be comparing the two humidity sensors mentioned above and make a final decision as to which sensor will be used

3.1V to 5.5V	1.7V to 3.6V	Working Voltage
I2C	I2C	Communication
99.9% rH	100% rH	Max Range
0% rH	0% rH	Min Range
22.5*12*4.7mm	17.78*25.4*10.6mm	Dimension
\$3.95	\$9.95	Price
AM2302	HTS221	

Table 28: Humidity Sensor Comparison Table

In terms of measuring humidity the AM2302 and the HTS221 are roughly the same. They both can measure from 0% to 100% humidity but the AM2302 has a slightly better range of accuracy by roughly 2%. The added feature of STEMMA QT on the HTS221 is nice but it is not necessary. The AM2302 is also much

smaller and much cheaper than the HTS221 with a price of \$3.95 instead of \$9.95 for the HTS221. On top of that it also has a wider range measurable temperature values compared to the HTS221. For those reasons, our team has decided to go with the AM2320 as our temperature and humidity sensor.

19.5 Oxygen Sensor

The air that we breath is not 100% oxygen and is a mix of oxygen and multiple other gases. By volume, the amount of oxygen that we breath in is roughly around 21% with about 78% being nitrogen and the final 1% being a mix of other gases. Our vehicle will be equipped with an oxygen sensor to determine the percentage of oxygen around the vehicle and be able to display that value onto our center console.[12]

19.5.1 MIX8410

Developed by Seeedstudio, the MIX8410 is an oxygen sensor intended for the Grove Arduino kit. It is an electrochemical oxygen sensor and what that means is that it measures a chemical reaction that occurs within the sensor and that reaction creates an electrical output that is proportional to the oxygen level. The connections on the grove modules do not have an I2C interface and as such uses regular analog to communicate. The MIX8410 is capable of measuring the percentage of oxygen in the air up to 25%.

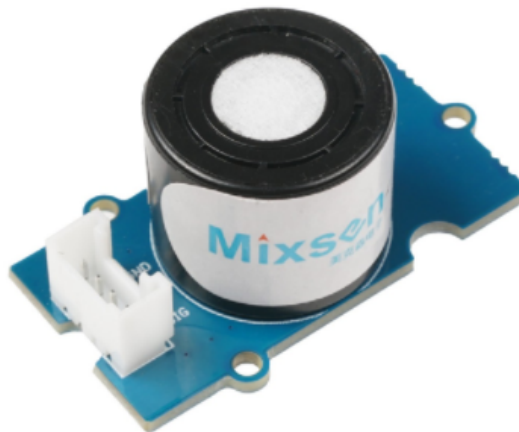


Figure 51: Seeedstudio MIX8410

19.5.2 SEN0322

Developed by DFRobot, the SEN0322 is part of their Gravity line of sensors. The SEN0322 is an electrochemical oxygen sensor so it creates an electrical output proportional to the oxygen level in the air. It is capable of measuring the percentage of oxygen in the air up to 25%. The SEN0322 also comes with an I2C interface for easy communication.



Figure 52: DFRobot SEN0322

19.5.3 Comparison and Decision

3.3V to 5.5V	3.3V to 5.5V	Working Voltage
Analog	I2C	Communication
25%	25%	Max Range
0%	0%	Min Range
\$49.90	\$53.90	Price
MIX8410	SEN0322	

Table 29: Oxygen Sensor Comparison Table

Both the MIX8410 and the SEN0322 have incredibly similar documented specifications. Input voltage is nearly the same, the percent oxygen range that they can measure are identical to one another, and even the price has only a few dollars in difference. The main difference between them is the fact that the MIX8410 uses analog for its communications while the SEN0322 is has an I2C interface for communications.

Our team has decided to go with the SEN0322 to be our vehicles oxygen sensor. Although it sits at a slightly higher price point than the MIX8410, the added feature of I2C communication is worth the few dollars extra the device costs.

19.6 Carbon Dioxide Sensor

Similar to the oxygen sensor mentioned in the section above, our vehicle will also have an carbon dioxide sensor. According to the United States Department of Agriculture, it is an OSHA violation to be exposed to more than 0.5% CO₂ in the air over an 8 hour exposure period. They also have a list of effects depending on the percentage of CO₂ in the air. At 1% there is possible drowsiness; at 3% a person will likely have an increased heart rate, and high blood pressure; at 4% and above there is an immediate danger to life or health. Our Carbon Dioxide sensor will measure the amount of CO₂ in the area and display that value on the center console of the vehicle.[13]

19.6.1 T6703-5k

The first Carbon Dioxide Sensor we will be looking at is the T6703-5k. It Nondispersive Infrared technology to measure the amount of CO₂ in the air around. According to co2meter.com, these sensor work by having a IR lamp that directs waves of light through a tube that contains samples of air. The CO₂ absorbs these light waves then and at the other side of the tube the remaining waves that were not adsorbed the CO₂ are read.

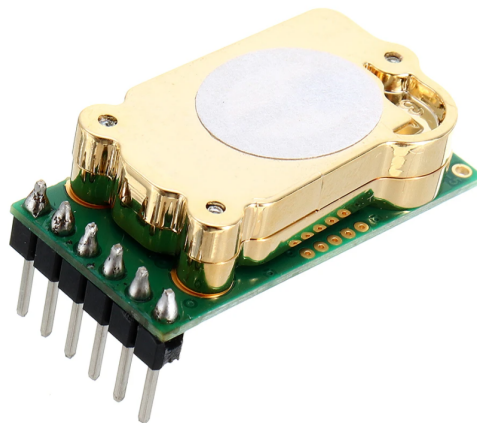


Figure 53: T6703-5k CO₂ sensor

The T6703-5k is capable of measuring the Carbon Dioxide in the area with a range from 0 ppm up to 5000 ppm; which is roughly 0% to 0.5% Carbon dioxide per volume. It has an accuracy of around ± 75 ppm. For communications, the T6703 is able to communicate using both I²C as well UART.

19.6.2 SEN0219

Designed by DFRobot, the SEN0219 is Carbon Dioxide Sensor that falls onto their Gravity line of sensor. It is capable of measuring the Carbon Dioxide in the area with a range from 0 ppm up to 5000p ppm; which is roughly 0% to 0.5%. It has an measurment accuracy that ranges between ± 100 ppm. Unlike many other sensors from DFRobots Gravity line, the SEN0219 does not have an I²C interface, instead it uses regular analog output for its measurements. The range for the analog output will range from 0.4V up to 2V.



Figure 54: DFRobot SEN0219

19.6.3 Comparison and Decision

In this section we will be Comparing the two Carbon Dioxide sensors that were mentioned above.

While the T6703-5k and the SEN0219 are similar in that they have the same range of measurable values, from 0 ppm to 5000 ppm, they are mainly different in price point and in how they communicate to a micro controller. The T6703-5k has an I2C interface so it does any calculations on the board itself then sends it to the micro controller, while the SEN0219 uses analog communication, so the the micro controller will have to read the voltage output from the SEN0219 to determine the the value. Unlike most other sensors mentioned in the sections above and below, the SEN0219 needs to be calibrated for the first use while the T6703-5k does not. Lastly the SEN0219 sits at a higher price point compared to the T6703, by around \$20.

Our team has decided to go with the T6703-5k for its I2C or UART communication as well as its lower price point.

4.5V to 5.5V	4.5V to 5.5V	Working Voltage
I2C or UART	Analag	Communication
5000 ppm	5000 ppm	Max Range
0 ppm	0 ppm	Min Range
\$37.99	\$58.00	Price
No	Yes	Calibration Required?
T6703-5k	SEN0219	

Table 30: Carbon Dioxide Sensor Comparison Table

19.7 Ultraviolet Sensor

Ultraviolet light is a form of electromagnetic radiation with a specific wavelength. It has a shorter wavelength than visible light and a longer wavelength than X-Ray. Too much exposure to UV light can cause damage to your health such as sunburn or even increase your risk of Skin Cancer. Our vehicle will be equipped with a Ultraviolet sensor to measure the amount of the intensity of Ultraviolet light which will then be shown on the center console.[14]

19.7.1 VEML6070

The first sensor we will be taking a look at is the VEML6070 from Adafruit. This sensor takes the amount of UV that is being detected and converts it to number. The value that the VEML6070 is unitless and does not correspond to any kind of UV light index value. This means that research will need to be done to figure out how much the sensor needs to be calibrated to provide accurate readings. The value that gets created is a 16 bit number, which means its maximum value is 65536. The VEML6070 comes with seven pins: the voltage input (Vcc), ground (GND), the two I2C communication pins (SDA and SCL), as well as an interrupt/alert pin (ACK).

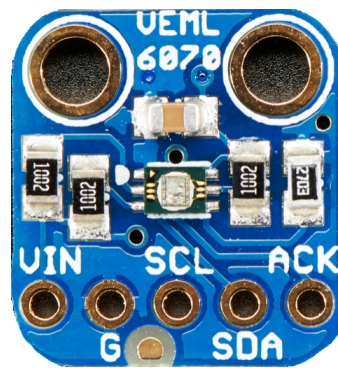


Figure 55: Adafruit VEML6070

19.7.2 ML8511

The ML8511 is a UltraViolet light sensor that is developed from DFRobot. It sensitive for UltraViolet wave lengths from 280 nm up to 400 nm. This is an analog sensor and does not give an actual unit for its UltraViolet intensity measurements and because of this it is necessary to calibrate the sensor to a known value in order for the ML8511 to produce accurate results. Since it is an analog sensor, it only has three pins: the input voltage (Vcc), ground (GND), and the analog output pin (OUT).

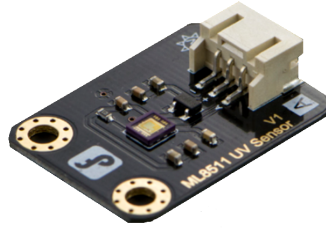


Figure 56: DFRobot ML8511

The output voltage for the analog will range somewhere between 1V and 3V. The amount of UV light is measured in milliwatts per squared centimeter (mW/cm^2) which also corresponds to the wavelength of the UV light. The more intense the UV light gets, the smaller the wave length and the greater the intensity. The figure below shows what the resulting analog output voltage when measuring UltraViolet intensities during operation at different ambient temperatures

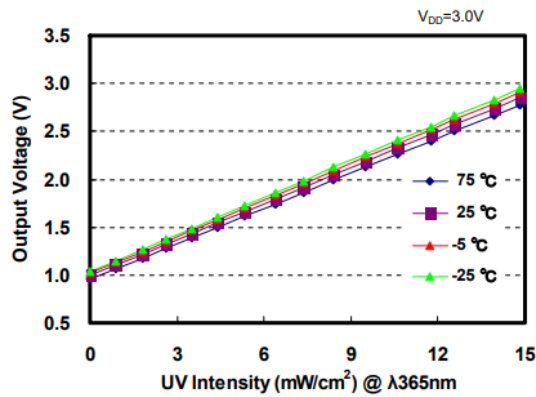


Figure 57: ML8511 Analog Voltage Output Graph

19.7.3 Comparison and Decision

When it comes to the VEML6070 and ML8511, there is not much of a difference between the two. Both take measurement of UltraViolet intensities and converts that to a output voltage. For both sensors, the output value does not have a unit so it would need to be tested with some sort of known value to continue getting accurate readings.

4.5V to 5.5V	4.5V to 5.5V	Working Voltage
I2C or UART	Analag	Communication
5000 ppm	5000 ppm	Max Range
0 ppm	0 ppm	Min output Voltage
\$5.90	\$9.90	Price
VEML6070	ML8511	

Table 31: Ultra Violet light Sensor Comparison Table

Our team has decided to go with the VEML6070 since the i2c interface would be easier to handle than the analog communication. It is also cheaper than the ML8511 which is great since our vehicle will be equipped with lots of sensors, we do not want it to be too costly.

19.8 Particulate Matter Sensor

A particulate Matter sensor, or PM sensor, is a device that can measure the amount of small particles in the air. These particles are similar to dust, dirt, or even soot and can be so small that they are unseen to the naked eye. These sensors are capable detecting incredibly small particles in the air, these sizes can range from up to 10um (micrometers) small or even up to 2.5um (micrometers). Our vehicle will be equipped with a Particulate Matter sensor to measure the general amount of small particles in the air, then this data will be sent to and displayed on the vehicles center console.

19.8.1 PMSA003I

The PMSA003I is a particulate matter sensor developed by Plantower. The sensor has a 1p pin connector with some of the pins not being used. The different kinds of pins are: Vcc and GND for the input voltage and ground respectively, SDA and SCL for the data and clock of the I2C interface, a reset pin for resetting the sensor (RST), and a SET pin which can be set to either High or low that will set the sensor to working or sleeping respectively.

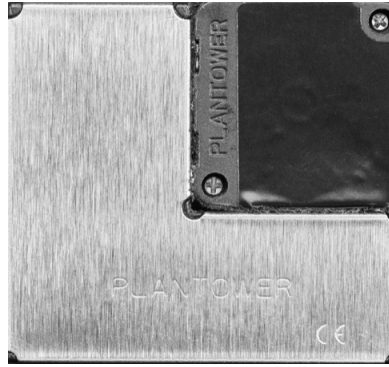


Figure 58: PMSA003I Sensor from Plantower

The PMSA003I takes in around 0.1L of air and will categorize the measurements into bins. These bins will reflect the size of the measured particles and will range from 0.3um, 0.5um, 1.0um, 2.5um, 5.0um and 10um. The information on each bin will be transmitted to the micro controller using I2C protocol. The data will be given in the units ug/m3 (micro gram per cubed meter). It is a small sensor, sitting at 38×35×12mm so roughly the size of a quarter

19.9 Volatile Organic Compounds Sensor

A Volatile Organic Compounds sensor, or VOC sensor, is a type of sensor that is capable of detecting a broad range of reducing gases. These gases have a low oxidation number and some examples of them include: alcohols, aldehydes, ketones, organic acids, amines, organic chloramines, aliphatic and aromatic hydrocarbons.

19.9.1 SGP40

The SGP40 is the one of the VOC modules from Adafruit. It is capable of measuring the VOC index in the air. The VOC index ranges from 0 to 500 with the average VOC in the air being somewhere around 100. The higher the VOC index gets, the more dangerous.

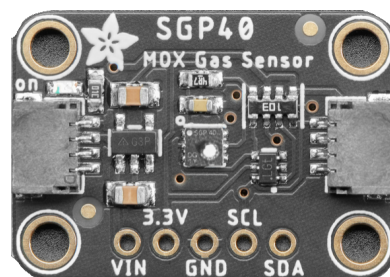


Figure 59: SGP40 from Adafruit

Humidity affects the output value of the SGP40 so a humidity sensor such as the ones in section 16.4 can allow the processor to adjust the value of the VOC index to be more accurate depending on the output of the humidity sensor. The SGP40 has an I2C interface with the STEMA QT connectors.

19.10 Light sensor

In this section, we will be discussing a couple different light sensors to be purchased. Most modern day cars are equipped with an auto headlight feature which ensures that the driver does not need to manually toggle on or off the headlights. This is done with a light sensor that is embedded somewhere on the vehicle. The sensor turns the amount of light it receives into an electrical signal then determines if the headlights need to be turned on or off. Our vehicle will have one light sensor for the auto headlight feature. The headlights are to be turned on when the ambient lighting outside fall below a certain level. This prevents driving in the dark with the headlights off by turning it on automatically.

19.10.1 Adafruit VEML7700

The VEML7700 is an ambient light sensor with an I2C interface. The sensor has four pins: the input voltage (V_{in}), ground pin (GND), and the I2C communication pins (SDA and SCL). It is connected to a break out board that has an extra pin called ($3V_o$) which acts 3.3V output if needed. Unlike other light sensors which give a number to represent how bright or dark the ambient lighting is; the VEML7700 calculates the lux, which is the SI unit for light. It provides an Ambient light range from 0 to 120,000 lx (lux).

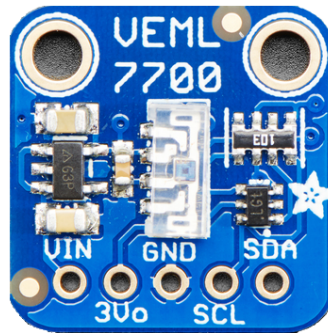


Figure 60: VEML7700

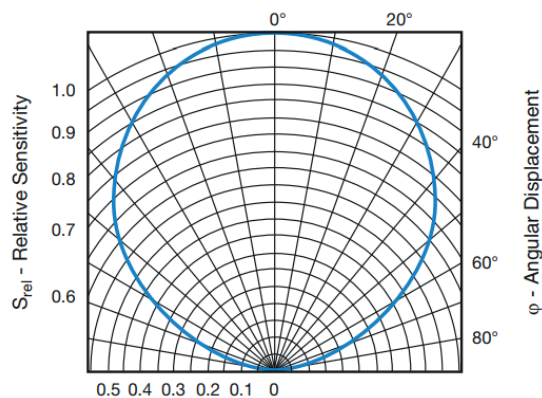


Figure 61: VEML7700 Relative Radiant Sensitivity vs. Angular Displacement

The figure above describes how sensitive the VEML7700 is with relation to the angle at which light is falling onto the sensor. Looking at the figure we can see

that the sensor is most sensitive when the light is hitting the sensor directly with a 0° angle. We can also see that the sensitivity begins to decrease as the angle at which light is hitting the sensor increases.

19.10.2 Adafruit TSL2591

The TSL2591 is another ambient light sensor from Adafruit that can provide a value to the amount of ambient lighting. The TSL2591 comes with infrared and full spectrum diodes. With the infrared diode, the sensor can be used to measure the amount of heat emitting from an object, similar to how a thermal camera works, and measure the ambient lighting with the full spectrum diode. It is a 6 pin sensor: the input voltage (V_{in}), ground pin (GND), the I2C communication pins (SDA and SCL), a 3.3V output pin ($3V_o$), as well as an interrupt pin (INT). The TSL2591 can measure light ranges from $188 \mu\text{lx}$ to 88,000 lux.

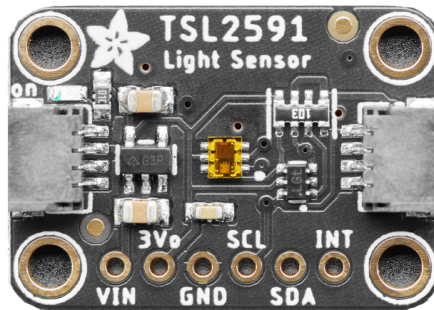


Figure 62: TSL2591

Another feature of the Adafruit TSL2591 is the addition of the STEMMA QT connectors on the sides of the module. STEMMA connectors allow for an easy, solderless, way to connect to other devices with I2C. Many Adafruit devices, come with a STEMMA connectors on board so there would be no need to have solder connectors to each individual pin.

19.10.3 Comparison and Decision

Both the VEML7700 and the TSL2591 are developed by Adafruit and measure ambient lighting in the same way. The TSL2591 has the added feature of being able to detect heat using the IR diode, but is not a feature that would be useful for our vehicle. Both sensors can communicate with a microcontroller using I2C interface. In terms of measurements, the VEML7700 is able to measure a higher range of lux than the TSL2591 by over 30,000 lux while being a smaller module.

2.5V to 3.6V	3.3V to 5V	Working Voltage
I2C	I2C	Communication
120 klux	88 klux	Max Range
0 lux	180 ulux	Min Range
16.6*16.5*4.0mm	17.78*25.4*10.6mm	Dimension
\$4.95	\$6.95	Price
VEML7700	TSL2591	

Table 32: Light Sensor Comparison Table

While the VEML7700 and the TSL2591 are very similar, our team will use the VEML7700 in our project. The TSL2591 does have some great features that are not on the VEML7700 such as the STEMMA QT connectors and the IR diode but those features will not be needed for the vehicles' automatic headlights. It is also slightly more economical to choose the VEML7700 over the TSL2591 as it is a few dollars cheaper and it is also slightly smaller.

19.11 Proximity sensor

Our vehicle will have four proximity sensors, one on each side of the vehicle. These sensors will provide distance measurements that will allow the on board computer to determine if the vehicle is too close another object. There are multiple different kinds of proximity sensors available for purchase, each with their own price points and method for measuring distances.

19.11.1 HC-SR04

The HC-SR04 module is 4-pin ultrasonic distance sensor. These sensors emit ultrasonic sound waves which would then bounce off an object if it is within range and be returned to the receiver. To determine how far an object is, the module measure how long it takes for the transmitted sound wave to be returned to the receiver. According to the data sheet, to calculate the range, divide the amount of time in micro seconds by 58 to get the range in centimeters or divide the amount of time in microseconds by 148 to get the range in inches. The 4 pins on the HC-SR04 are the input voltage pin (Vcc), the sound wave transmitter pin (trig), the receiver pin (echo) and the ground pin (Gnd).

This sensor can be bought with the common GPIO interface or with the I2C interface. The version of the HC-SR04 with the GPIO interface does not calculate

directly on the module itself, instead, when the transmitter emits the sound wave, the echo pin is set to high then back to low once the sound wave returns. From there it is up to the processor to do the calculations to find distance. The version of the HC-SR04 with the I2C interface does the calculations directly and passes that value to the processor.



Figure 63: HC-SR04 Ultrasonic Distance Sensor

Working Voltage	DC 5 V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2cm
MeasuringAngle	15 degree
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion
Dimension	45*20*15mm

Table 33: HC-SR04 Specifications

19.11.2 MB1040 LV-MaxSonar-EZ4

The MB1040 LV-MaxSonar-EZ4 works similarly to the HC-SR04, as it also uses ultrasonic sound waves that have been reflected back to determine distance from an object. It measure the time it takes for the reflected sound wave to received to determine the distance. Range calculations on done directly from the sensor and the result is passed on to the processor.



Figure 64: MB1040 LZ-MaxSonar-EZ4

Working Voltage	2.2V to 5.5V
Working Current	2mA
Working Frequency	42KHz
Max Range	645.16 cm
Min Range	15.24 cm
Dimension	22.1*19.9*15.5mm

Table 34: MB1040 LV-MaxSonar-EZ4 Specs

19.11.3 Comparisons and Decision

In this section, we will be comparing the two proximity sensors discussed in the previous sections.

Both are ultrasonic Proximity sensors so they both use sound waves to determine the distance of an object. The MB1040 LZ-MaxSonar-EZ4 is not only smaller than the HC-SR04 but it is able to detect objects that are farther by around 200cm compared to the HC-SR04. The HC-SR04 and the MB1040 communicate using I2C but the HC-SR04 has different versions that are developed by different manufacturers that have regular GPIO pins if necessary for cheaper. The largest drawback with the MB1040 LZ-MaxSonar-EZ4 sensor is its price. For a single sensor, it cost a little more than \$30 while the HC-SR04 is much more cost effective as four sensors would cost around the same as one of the MB1040 sensors.

5V	2.2V to 5V	Working Voltage
15mA	2mA	Working Current
400cm	645.16 cm	Max Range
2cm	15.24 cm	Min Range
45*20*15mm	22.1*19.9*15.5mm	Dimension
\$7.95	\$32.95	Price
HC-SR04	MB1040 LZ-MaxSonar-EZ4	

Table 35: Proximity Sensor Comparison Table

Our team will use the HC-SR04 in our project. We chose this sensor because for our vehicle, we will be needing four of these sensors so spending around \$120 as opposed to \$32 is much more friendly on our wallets. The extra 200cm of distance that can be measured with the MB1040 would not be necessary for our vehicle since the purpose of the proximity sensors is to detect nearby object.

20 Testing and Design

20.1 User Interface

In this section, we will be discussing different options in implementing a GUI interface that can interact with our micro-controller unit and display environmental information which will be measured by the different sensors in our design. The user interface is one of the essential components of our vehicle's design as it can display environmental information which is important if we're trying to design a vehicle similar to a Mars Rover. We will not be utilizing all sensors in the interface, as they will have different functions that we feel wouldn't be necessary to display, such as the light sensor which will only be used for automatic headlights.

20.1.1 Options for User Interface

Below are the different options we will be considering between a HTML 5 web application, a Python application, and a Windows Form application. Each platform provides it's own unique set of features and challenges when designing an interface. The benefit of each platform, is that they each have different libraries available online to help design an interface. HTML 5 offers a rich amount of libraries for different components that will be discussed in more detail below. Python also features extensive support with different libraries that we can use. Finally our

third option, Windows Form Application, could work. Although we have the least experience working with it, designing a GUI might actually be the easiest to do as the graphical components are already built-in to the IDE and even supports SERIAL connection which can allow us to directly connect to an MCU. For a more detailed summary of the frameworks, we will elaborate on these options further below.

One thing to note is that all three applications will require some form of internet connectivity. The reason for this is that depending on the option we choose we need a solution for data to be received from the MCU. How we will solve this is by hosting a web server that will be hosted on the Pi and receive data from the MCU, from there the Pi will upload that data to a web server which will then be shown on the front end GUI.

20.1.2 HTML5 Web Application

One way we can implement a GUI is through an HTML5 web application. Application development with this framework wouldn't be too hard as well, as HTML5 is known for its compatibility and modularity with different libraries such as Google Charts and jQuery. The amount of support for the language and libraries is rich as well, which can help solve any problems we might face during development.

The web app will connect to our micro-controller and display the info through different web components such as the many charts that the Google Charts library provides. How we will host the application is by having a separate computing device such as a Raspberry Pi running the app locally or host it over a domain on the Internet. How we will accomplish this is by using a separate component that can allow us to connect to the Internet such as the Arduino Ethernet Shield. The Arduino Ethernet Shield will be acting as the back-end that will be sending data and receiving requests. Whereas, the Raspberry Pi will be hosting the HTML5 front-end application that will receive and display the data and send requests for certain actions. If we choose to do a local host, the Arduino Ethernet Shield would be sufficient enough as we can directly connect it to the Pi. If we choose to host the web app on a domain, then we need to make sure that the Pi is able to connect to the Internet in the first place.

This will be done either through a WIFI network or LAN network. The MCU needs to utilize a separate module to be able to communicate via WIFI, and the same goes for LAN connection which add to our overall costs. This is already a requirement for our design which is listed as under the GSM chip. We also need to purchase a SIM card as well as the monthly service for it to operate. Besides that, connection to an API wouldn't be too hard as well with all the resources and documentation out there.

20.1.3 Python Application

Another option we can consider is utilizing one of the many packages available for Python that help in building GUI's. The main package we are considering is tkinter which can be used to design different components and link different functionalities from the MCU, such as LED toggling and data plotting. Since this is a Python package, the Raspberry Pi would work well with this as it comes with pre-installed with the device. Other packages that could be beneficial in

our design would PyQT, PyGUI, and Kivy. The main reason we are considering using Python is because it gives us the option of showing a real-time feed that will be captured by our camera and allow for users to see what the vehicle is seeing. Although this isn't a requirement for our design, this could be a great addition after we finish completing the prototype for it.

As stated previously, this framework also allows us to connect to an API which will be necessary since we will be retrieving data from a server that is speaking to the MCU. This works via a library called Python Requests that allows the code to send HTTP requests. This way the MCU sends data to a web server, either local or hosted through the internet. Then the data is held in the server, which is then accessed via the HTTP requests from our Python code.

20.1.4 Windows Form Application

Our final option to create our GUI is through Windows Forms. This UI framework is based around .NET and allows users to generate events based on actions they do with the interface. Events can include mouse-click, key-press, etc., and the code portion processes the event and returns different output. A benefit to using a Windows Form application is that creating graphical components will probably be the easiest to do compared to the other frameworks. Graphics can simply be added via drag-and-drop which means we could designate team members for designing the UI components and for implementing the functionality of each. Another benefit of using Windows Forms is that we do not have to rely on using other libraries for creating graphics such as charts and graphs as they are already built into the platform.

The Windows Form application has the option to send and receive HTTP requests which we can then use to display information onto charts and the like, but we also have the option of being able to connect the Pi directly to the MCU via a serial port. The benefit of this is that we can forgo the initial server setup and simply pull data from it and not worry about any server problems we may face in the long run. Although if we choose to do serial communication we have to decide between using I2C, SPI, or UART. This is further elaborated in a previous section.

20.1.5 Comparison

In comparison, all three frameworks are all trying to solve the same issue of how we will be implementing the GUI for our device. For our final decision this had to come down to what we were most comfortable using. The frameworks we are most comfortable with are listed in this order (from most comfortable to least): HTML5, Python, and Windows Form. The first two technologies have a lot of documentation as well as external libraries that can solve or even extend our GUI's functionality. In the next section, we will need to make sure we can decipher and send the data to a database so that it can be accessed through a front end. We can do so by using Python, so this means that we will have to utilize this in our design too.

20.1.6 Final Decision

For our final decision, we have decided that we would need to use two of these frameworks to accomplish the GUI design. The reason being is that having a front-end portion in our design is vital, but we need to make sure that the front-end can receive that data in the first place. The way we can do this is using another technology, Python, to receive and send data to a database to use this information. Therefore, we decided that for our Front-End framework we will be using HTML5, and for our translation we will be using Python. In the next section, we will explain how exactly we will be using Python to handle the data.

20.2 Backend Connection

In this section, we will discuss how we will be sending data from the Arduino to the Raspberry Pi. The way we can connect the two is via serial communication. There are multiple options to choose from to do this type of connection whether it be through UART, SPI, or I2C. Overall, communication between the two devices will not be a problem. The main challenge here is saving the data that's being received from the Arduino into a database that can then be accessed through a web server which then displays onto a front-end GUI. Two solutions to this would be using serial communication to connect the two devices. Then we would need a separate program that will send this data to a MySQL database. Python can accomplish this with external libraries such as MySQLdb.

The second option we have considered is having the MCU send the data directly to the database with use of an external module such as the ESP8266. The ESP8266 is a Wi-Fi microchip that can enable external MCU's to connect to wireless networks. With this, you can then further connect to a database that will "hold" the data. Since the data will be stored and updated in the database, we can just create a simple back-end server that can access the data.

20.2.1 Serial Communication

Serial communication is a broad term of how we'll be connecting our sensors to our MCU and other devices. There are three popular types of serial communication such as: UART, SPI, and I2C. We are not limiting ourselves to using one type of communication as each sensor has their own method of communication as well so we need to be familiar with all three types. The way we will be connecting the MCU to the Pi will be through UART. We can connect the two either: directly with USB, or through GPIO pins.

Once connected, we just need to design a program that can use the data that's received from the MCU and then send that data to the MySQL database. As stated earlier, it won't be too hard to implement this as there are dedicated libraries that both allow Python to interface devices connected through serial communication, as well as connect directly to a database.

20.2.2 Wireless Communication

Wireless communication is essentially have devices communicate with each other over a network to send different types of data. How this will work with our design is by using an external module that extends the MCU's networking capabilities. The device in question, the ESP8266, let's us do so as long as there is an Internet connection. With this, we can upload information from anywhere that has connection and be able to use that data as we see fit. Since we are just sending data directly to a mySQL database. One major problem we could face is the issue of being connected to a network. Since the design is meant to survey and analyze remote areas, we can face the issue of not having Internet connection. We have two solutions for this.

Firstly, since we are going to be utilizing a GSM module for our Pi, we can have the ESP module connect to this which gives us access to different cellular networks essentially providing Internet connection on the go. Another solution would be to have the Pi host a local database that will receive the information from the MCU. This can be done using a development environment called XAMPP. Essentially what this does is that it allows us to host a local database that will receive the data.

20.2.3 Comparison

In comparison, serial communication and wireless communication both aim to solve a problem in our design, sending data to the computing device. Serial communication is advantageous in a way that we just need to connect the two devices with either jumper cables, or USB and we set up a connection easily, and then create a python program to send the data to a database. With wireless communication we have to configure the ESP8266 and create a local database with XAMPP and then connect both to each other. However, once the connection is done we don't have to worry about much else.

A disadvantage with serial communication is that it can result in slower data transmission. Whereas with wireless it wouldn't be an issue so long as the two devices are communicating locally. If we relied on an external network then it would entirely depend on if there is service in a particular area.

20.2.4 Final Decision

In order to decide on which technology to use for our design we need to consider two things: where will the MCU and Raspberry Pi be located, and what environments it will be traversing. Since both devices are going to be placed into the same vehicle, it might be overkill to setup wireless communication to send data, when a simple USB connection can solve that easily. Not only that, if we consider the different environments we want to operate in there is the issue of not having any Internet service due to how remote an area is. Although this aspect affects both types of communication since we will be using a GSM module in our design. The added price of having to buy the ESP8266 also hurts our overall budget. Therefore, our team have decided that we will be using serial communication going forward in implementing this connection to the GUI.

20.3 Jetson Nano Test

Initial testing of the Jetson Nano is shown in the image below. This image is an object recognition image that accurately identified all of the people in the image. The process was easy to learn, and only took about an hour from the initial start up to the final result. This object recognition image was the result of NVIDIA's Jetpack "Hello World AI" Tutorial. This tutorial uses pretrained models for image classification and object detection. NVIDIA has great tutorials for object recognition with the NVIDIA Jetson Nano. NVIDIA tutorials, along with community support forums, will help our team progress with a complex object recognition system that will assist our vehicle when autonomously navigating.

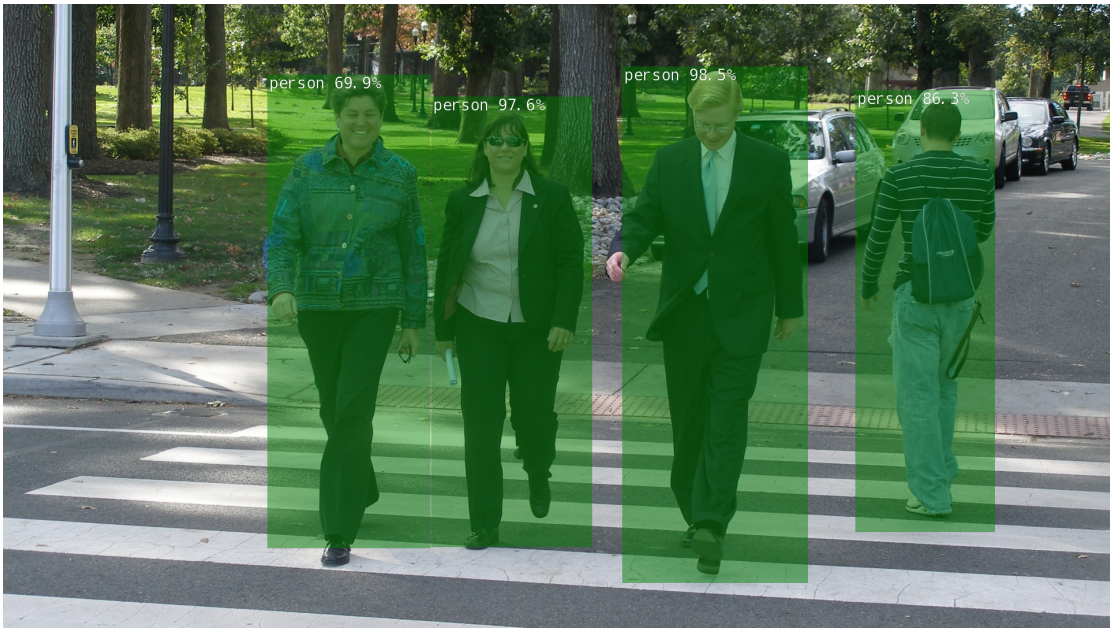


Figure 65: Jetson Nano Object Recognition Test

20.4 ATMEGA2560 & ATMEGA328p Test

Initial testing of the ATMEGA2560 and ATMEGA328p is shown in the image below. For this test an Ardiuno Mega and Arduino Uno were used for testing the microcontrollers. This test was done to get comfortable with the microcontrollers, as well as test I2C serial communication. To do this, a temperature and humidity sensor was used to transfer data over I2C from the Arduino Uno to the Arduino Mega.

The Arduino Uno controlled the temperature and humidity sensor, and relayed the data to the Arduino Mega. This was done once a second, and the Arduino Mega requested the data from the Arudino Uno. The Arduino Mega would then display the temperature and humidity on an LCD display, and would light up an LED depending on the temperature. The green LED would light up if the temperature was below 75°F, yellow if the temperature was between 75°F and 85°F, and red if the temperature was above 85°F. Since our vehicle will be transferring all of our sensor data to a master controller that will store the data, it's important to get

a good understanding of this process, and this test helped our team understand how this will be done.

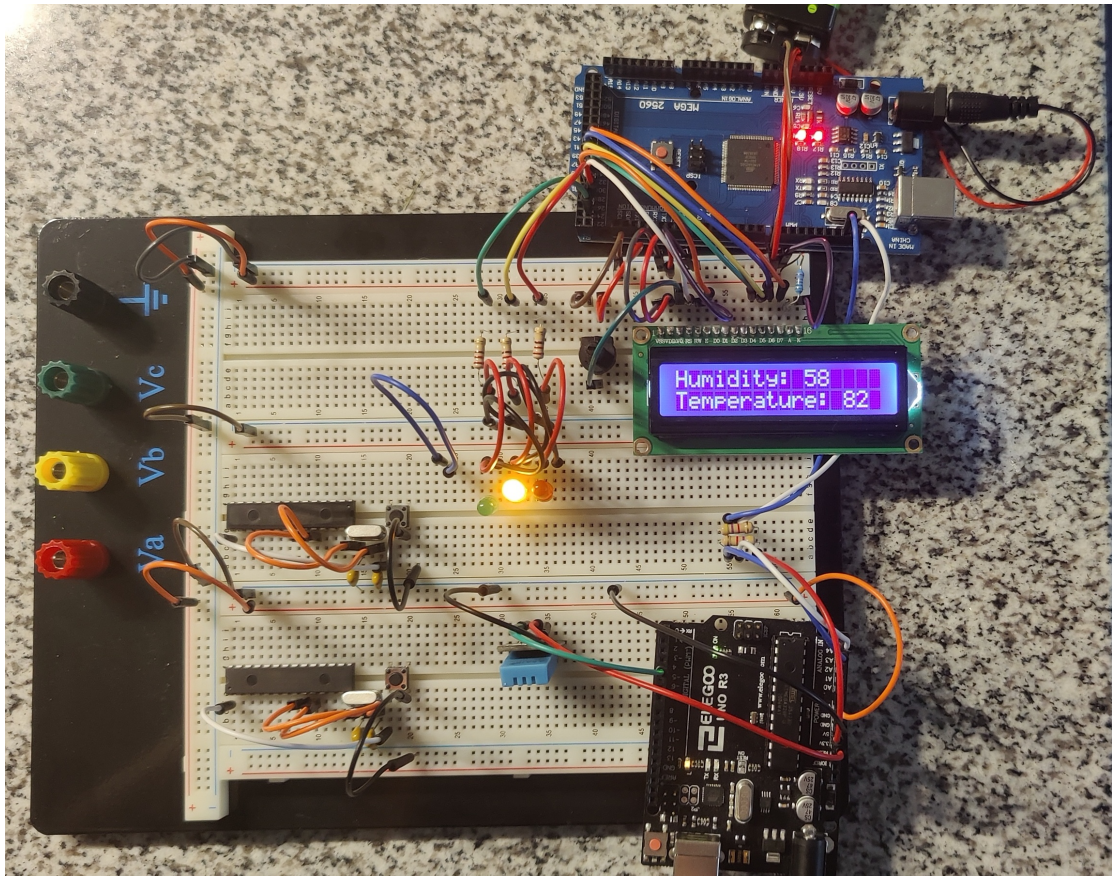



Figure 66: ATMEGA2560 & ATMEGA328p I2C Test

References

- [1] O.-R. A. D. O. committee, *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, apr 2021.
- [2] J. S, “18650 dual battery holder.” <https://www.thingiverse.com/thing:2821437>.
- [3] “Particle sizes.” https://www.engineeringtoolbox.com/particle-sizes-d_934.html.
- [4] S. Campbell, “Basics of uart communication,” Apr 2017.
- [5] S. Campbell, “Basics of the spi communication protocol,” May 2018.
- [6] S. Campbell, “Basics of the i2c communication protocol,” Apr 2017.
- [7] S. Inc., “Introduction of inr18650-30q.” <https://eu.nkon.nl/sk/k/30q.pdf>.
- [8] E. P. Inc, “Exp1280 12v8ah sealed lead acid battery.” <http://www.expertpower.us/exp1280>.
- [9] “Fram or flash: How to select the right mcu for your application,” 2021. https://e2e.ti.com/blogs_/b/process/posts/fram-or-flash-how-to-select-the-right-mcu-for-your-application.
- [10] M. S. Shawn Hickey, David Coulter, “Ux checklist for desktop applications,” 2020.
- [11] “Jetson tx2 nx module,” Feb 2021. <https://developer.nvidia.com/embedded/jetson-tx2-nx>, journal=NVIDIA Developer.
- [12] “10 interesting things about air – climate change: Vital signs of the planet,” Nov 2016.
- [13] CO2Meter, “How does an ndir co2 sensor work?.”
- [14] C. for Devices and R. Health, “Ultraviolet (uv) radiation.”

A Permissions

 Dustin Franklin <dustinf@nvidia.com>
Sat 7/17/2021 10:39 AM
To: Devon Wilkerson

Hi Devon,

No problem - good luck with your paper!

Best regards,
Dustin

Are the suggestions above helpful? Yes No

[Reply](#) | [Forward](#)


From: wilkersonrdevon@knights.ucf.edu <wilkersonrdevon@knights.ucf.edu> on behalf of NVIDIA Developer <noreply@nvidiadeveloper.com>
Sent: Saturday, July 17, 2021, 10:03 AM
To: Embedded Contact Form
Subject: [Jetson & Embedded Computing] Usage Permission of Pictures on Website

External email: Use caution opening links or attachments

Devon Wilkerson (wilkersonrdevon@knights.ucf.edu) sent a message using the contact form at <https://developer.nvidia.com/contact>.

Hello, I am a student at the University of Central Florida, and I am currently writing a senior design paper which includes discussion about a couple of your products. These products include the Jetson TX2 NX and the Jetson Nano Development Kit. I was wondering if I could have permission to use the pictures on the product page of the to include in my paper for reference? Thank you for your time.

Figure 67: NVIDIA Permission

 Elo Touch Support <support@elotouch.com>
Sat 7/31/2021 11:14 AM
To: Kris Choudhury

Hi Kris,

I apologize for the delay in this. Our legal team said it was fine for you to use the image from our website.

Thanks,
Byron

Figure 68: ELO TouchSystems Permission

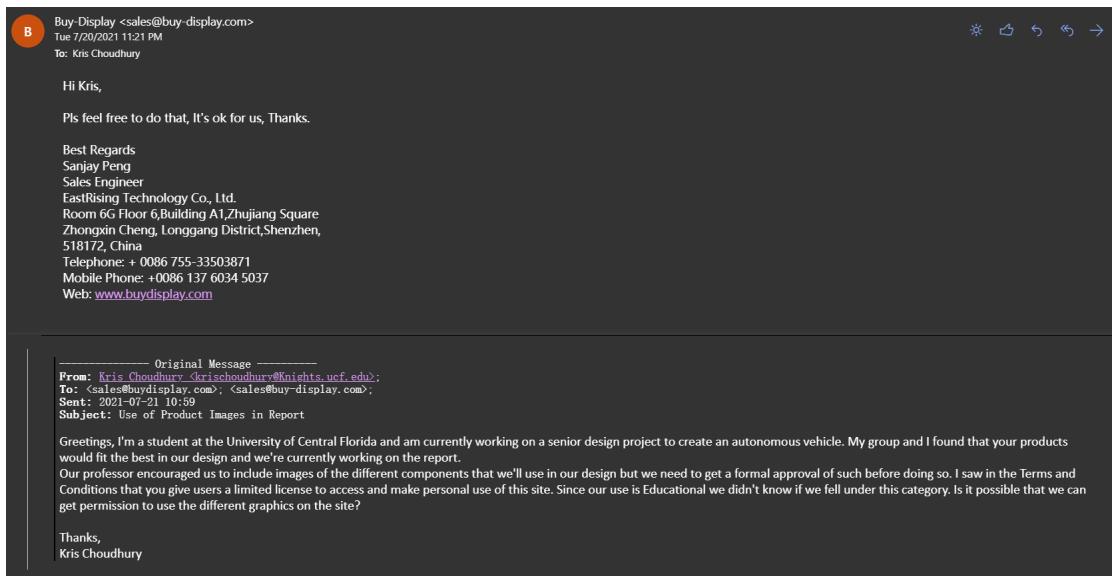


Figure 69: BuyDisplay Permission

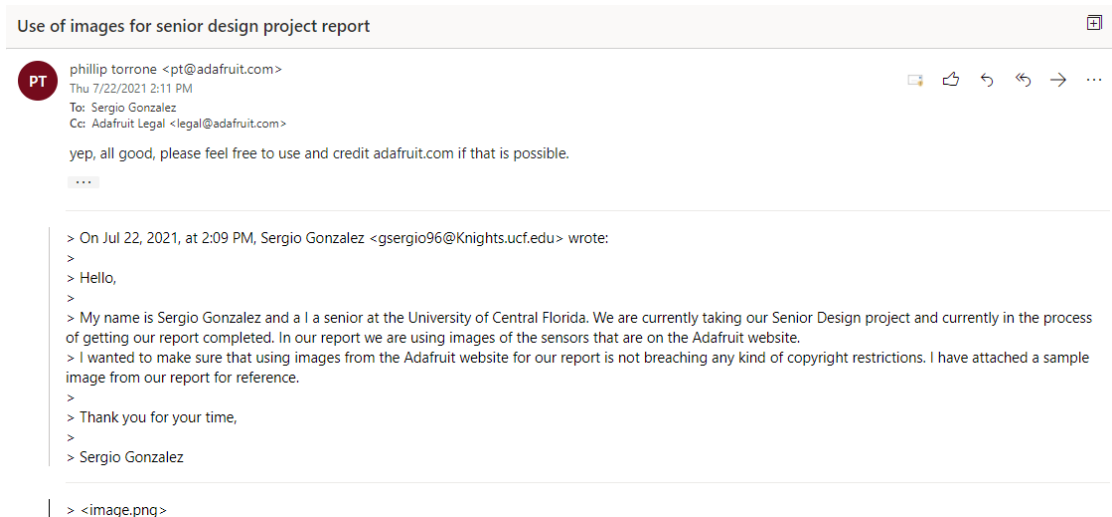


Figure 70: AdaFruit Permission

New Message From Circuit Basics



Scott C <circuitbasics@gmail.com>

Fri 7/23/2021 9:27 PM

To: Sergio Gonzalez



Hi Sergio,

We love it when students want to use our images in their projects. Please feel free to use any of the images on Circuit Basics as long as it's for educational purposes. Just a link citation to the source page would be greatly appreciated. Thanks and good luck with your project!

Thanks,

Scott
Circuit Basics
www.circuitbasics.com

Thursday, July 22, 2021, 11:27 AM -0700 from mail@circuitbasics.com <mail@circuitbasics.com>:

Hello, I am a senior at the University of Central Florida and im currently working on a report for our senior project. I was wondering would be any kind of breach in copyright to use images such the ones on this page: <https://www.circuitbasics.com/basics-uart-communication/> to be used in our report. Thank you for your time, Sergio Gonzalez

I have read, understand, and agree to the Privacy Policy: I have read, understand, and agree to the Privacy Policy

Reply | Forward

Figure 71: Circuit Basics Permission

21:45



this one is ok for you?

yes great. thanks! could I use the listings you posted on your store as well?



Read

21:52



ok,friend,you could use our picture and specification

Follow Store



Type your message...



Figure 72: Solar Panels & Controller Permission



New message from: [prowmotor](#) (2)

That's my pleasure. Just take them. Thank you!

[Reply](#) [Make an offer](#)

Your previous message

Hello, I am a student at the University of Central Florida writing a research paper on electrical vehicles and wanted to use the image of the go-kart rear axle kit. Could I have your permission to include them in the paper, strictly for educational use?

Thanks,
Ben

Figure 73: Go Kart Axle Permission



Wing <cs03@omc-stepperonline.com>

9:04 PM

To: Benjamin Goerd

Dear Ben,

Good day!

This is Wing from Stepperonline.

Glad to receive your enquiry.

Yes, you can use the image of the NEMA 34 Stepper in the paper for educational use.

If you have any further questions, please feel free to contact us.

Best Regards

Wing

STEPPERONLINE

Tel: +86-25-87156578 x059

cs03@omc-stepperonline.com

www.omc-stepperonline.com

Figure 74: Nema 34 Stepper Permission

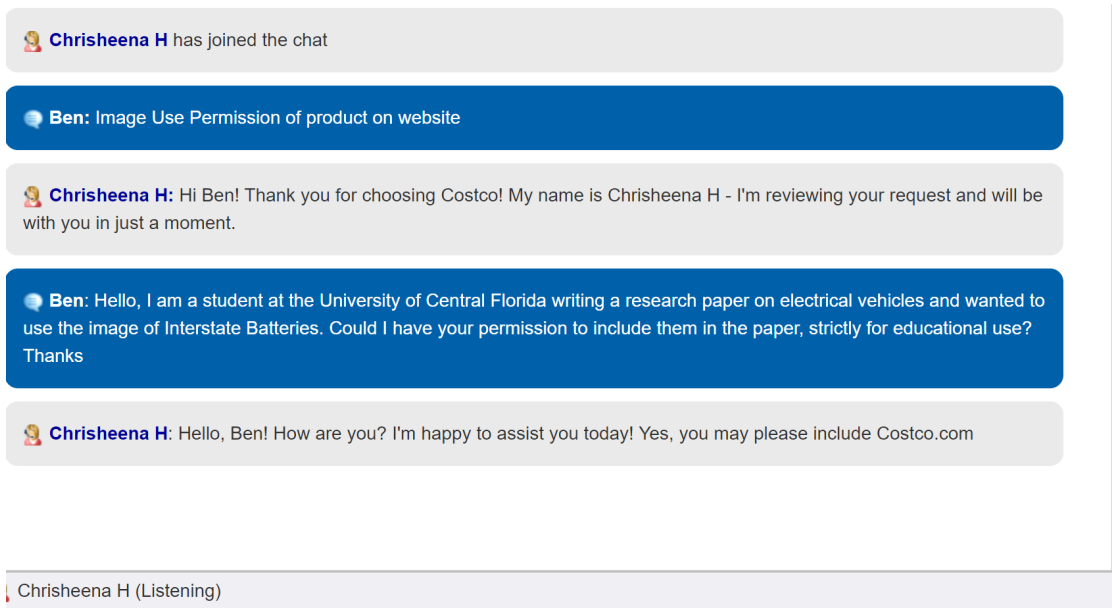


Figure 75: Interstate Battery Permission

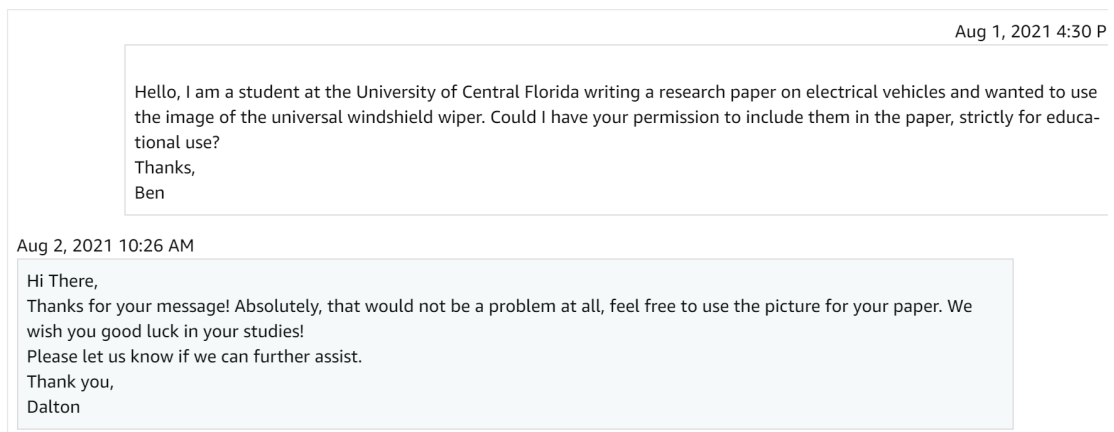


Figure 76: Universal Windshield Motor Permission

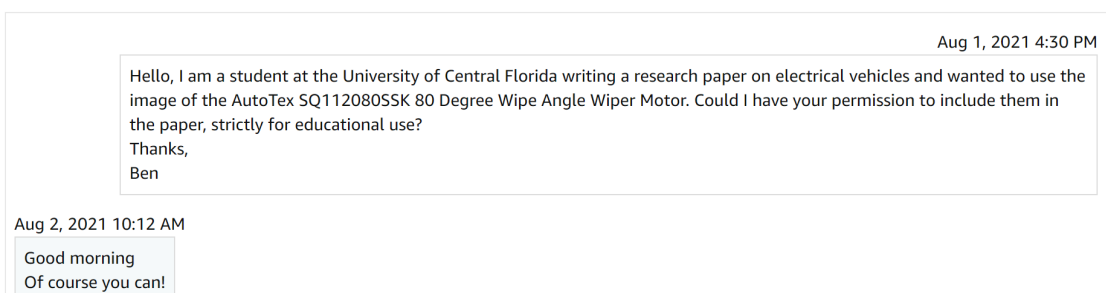


Figure 77: AutoTex Motor Permission



Sales - D&D motor Systems <sales@ddmotorsystems.com>

4:02 PM



To: Benjamin Goerd

Sure

From: [Benjamin Goerd](#)
Sent: Monday, August 02, 2021 3:42 PM
To: sales@ddmotorsystems.com
Subject: Image Permission Use

Hello, I am a student at the University of Central Florida writing a research paper on electrical vehicles and wanted to use the image of the Golf Cart Motor. Could I have your permission to include them in the paper, strictly for educational use?

Thanks,
Ben

Figure 78: Golf Cart Motor Permission



Hobby Club <hobbyclub@earthlink.net>

4:30 PM



To: Benjamin Goerd

Ben:

Yes, no problem.
Please e-mail us a copy of your paper,

Alberto

-----Original Message-----

From: Ben Goerd <bengoerd@knights.ucf.edu>
Sent: Aug 2, 2021 12:55 PM
To: Support <hobbyclub@earthlink.net>
Subject: Website Inquiry from Hobby Club

From: Ben Goerd
Mail: bengoerd@knights.ucf.edu
Telephone: [REDACTED]

Hello, I am a student at the University of Central Florida writing a research paper on electrical vehicles and wanted to use the image of the a few motors, including a Tonegawa 050. Could I have your permission to include them in the paper, strictly for educational use?

Thanks,
Ben

Figure 79: RC Motor, Servo, Actuator Permission



Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)

This is a human-readable summary of (and not a substitute for) the [license](#). [Disclaimer](#).

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material

The licensor cannot revoke these freedoms as long as you follow the license terms.

Figure 80: 18650 Cell Holder Permission