Solar Powered Beach Buggy Challenge

Group #1

Team Members:
Jared Cozart
Jose Rosales
Robinson Charles
Tony Jimogaon

Summer 2018
June 30, 2018
Sponsored by Duke Energy

Table of Contents

List of Figures

List of Tables

1. Executive Summary

In recent years, there has a been a growing interest in both the public and commercial sectors with regards to the viability of automated motor vehicles. While there has been much research and prototyping done in the realm of converting traditional, on-road vehicles into self-driving cars, our team was unable to find anything concerning the automation of cars for non-standard terrain, such as sand. As a result, we have decided to create an automated, single passenger buggy for use on a populated beach. It will also be entirely operated on solar power, in order to demonstrate the viability and low power consumption of environmental detection and recognition systems.

The buggy will consist of a chassis, able to carry a passenger weighing a minimum of 120 lbs., across the beach at approximately 3 miles an hour. Through a combination of stereoscopic camera vision and Lidar, it will be able to conclusively detect and identify objects, communicate with an onboard Arduino unit, and through there affect the rotational speed of the wheels in order to turn the buggy away from a collision course with the object. Through the use of solar panels, as well as two onboard batteries, it will be able to operate for a minimum of 10 hours, assuming ideal weather conditions and functional components.

Lidar will be used as a primary method of surveying the landscape. Due to its basis on time-of-flight technology, it is an ideal method for quick and precise distance measurements to within one centimeter. It can generate a 'point cloud', or a series of data points with differing distances and angles from the origin, that are within its field of view and are constantly updated, and these data points can be fed into the Arduino in order to identify objects. The system will also utilize two stereoscopic cameras on the front of the buggy, giving us a rudimentary three-dimensional image, which can be analyzed by image recognition software running on the Arduino unit. Through the use of both systems in unison, we can be certain of any objects that could potentially obstruct the path of the buggy, and program the buggy to move around them.

In an effort to emphasize the general independent nature of the buggy, and with consideration to the bright environment it will be operating in, it will be powered by a solar panel placed on a platform located at the rear of the buggy. The panel selected will be chosen for its ability to operate at a maximum efficiency in the bright sunlight, and for the capability to output a sufficient level of wattage, so as to properly run the motors, Arduino, location and navigational systems onboard. The solar cell will also turn slightly, using two smaller solar cells in a wedge pattern in order to detect the general location of the sun and maximize the amount of sunlight incident on the surface at any given point in time.

The primary goal of this project is to demonstrate the feasibility of both solar energy and automated vehicles, while simultaneously aiding beachgoers during their visit. The costliest aspects of our design will be the motors operating the wheels, and as

a result we hope to showcase an efficient and cost-effective way to provide self-driving capabilities to conventional vehicles.

This project is of course being done in conjunction with a group of mechanical engineers. Their role in the project will be in designing the chassis we will use as a basis for our electronic components, and in helping select the motors we will use to operate the buggy.

2. Project Description

The following section provides a general description of the project, listing all of the constraints and engineering requirements we were given by our sponsor. It will also describe the limitations on the components we will choose as we prototype our design.

2.1 Block Diagram

The diagram below in Figure 1 is a block diagram illustrating the general layout of the operating components, as well as their inputs and outputs.

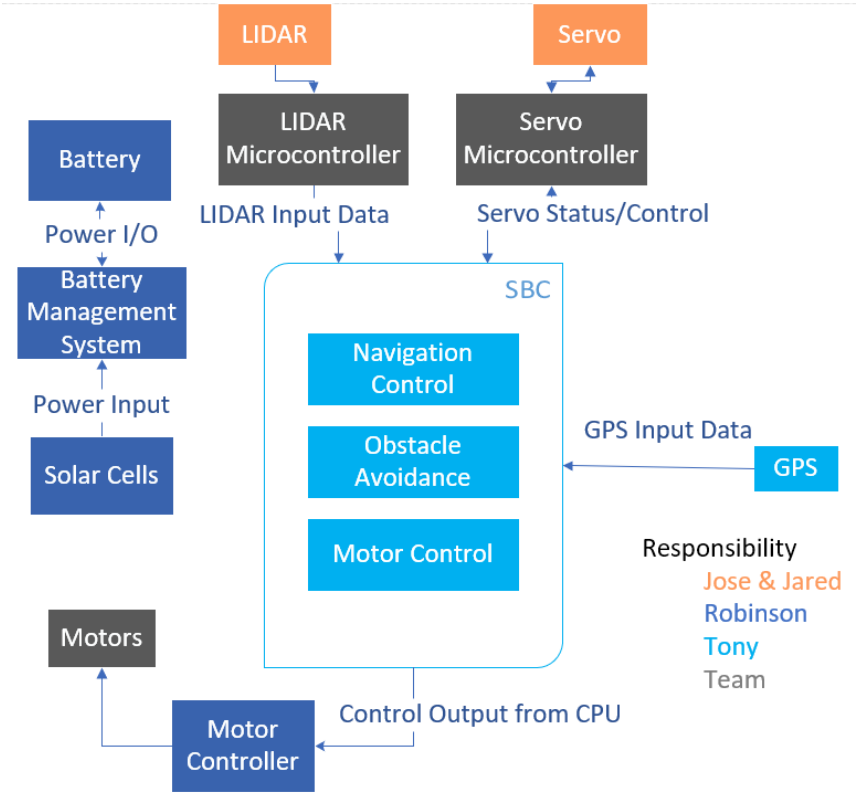


Figure 1 - The block diagram of the electrical system for the buggy.

In an attempt to organize the workflow for this project, a list detailing the individual components of the project and the respective members who are responsible for that component's successful development is included below in Table 1.

Component Description	Team Member(s) Responsible	Component Status
Lidar	Jose Rosales, Jared Cozart	Complete
Lidar Microcontroller	Jose Rosales, Jared Cozart	Complete
Battery Management Systems	Robinson Charles	Complete
Solar Cells	Robinson Charles	Complete
Obstacle Avoidance	Tony Jimogaon	Complete
Motor Controller	Tony Jimogaon	Complete
Navigation Controller	Tony Jimogaon	Complete

Table 1 - A list of each of the components of the block diagram, the team member or members responsible, and their status.

2.2 Requirement Specifications

Our buggy has a well-defined set of criteria it must meet in order to be deemed a success. In terms of its power source, it must be able to run entirely on solar power for the entirety of time spent in operation. While in operation, it must not exceed speeds of 3 miles per hour, and it must be able to undergo a 10-mile trip along the beach, and successfully make the return trip. It must be designed to be able to withstand a carrying capacity of a 120-pound passenger. It must be able to operate without causing any harm to the environment, local wildlife or beachgoers, and it must be constructed on a budget that does not exceed our established budget of $2000.

2.3 Project Constraints

The constraints herein are obstacles encountered when designing the buggy. These prove to be challenges that has to be overcome in order for the buggy to operate successfully.

- Ability to operate under suboptimal weather conditions, such as a cloudy sky
- Ability of solar panels to adequately power all necessary subsystems for the duration of the trip
- Must be able to identify and steer away from ocean

2.4 Economic and Time Constraints

Given the scale of this project, there are significant economic restraints that need to be taken into consideration. We were given a budget of approximately $2000. Despite appearing as a sizeable amount of money, this budget is then split between the electrical and mechanical engineering teams, to be allocated on the basis of importance to the overall function of the system. For reference, typical automated vehicles using LIDAR spend between $12000 and $75000 for a single LIDAR system operating on a single vehicle. As a result, many difficult decisions concerning the allocation of the budget had to be made, with the costliest components appearing to be the tires and motors for the buggy.

As for our time constraint, significant consideration for the time requirement is a must. Our group alone will be working on the second semester in realizing the buggy. That means that we will not have the Mechanical or Computer Science teams working with us. While the absence of the Mechanical team can be remedied with the purchase of a subsystem that has most of our Mechanical requirements (such as a Power Wheels children's vehicle), the subsystem handled by the Computer Science team will still need to be handled by our team, as it is considered a vital component of the system. Without it, the buggy will not be an autonomous buggy.

Another consideration is that our second semester is a summer semester. Therefore, it has considerably less break time allotted to us than normal. Most teams will have the break between Fall-Spring or the summer between Spring-Fall to polish their plan, order their parts, and start assembly while our team will have only one week in between Spring-Summer.

And lastly, the summer semester itself is only about 12 weeks--4 weeks less than a Spring or Fall semester. As such, our team will have to design our system in such a way that it can be quickly assembled and tested once it is finalized. Specifically, the stereoscopic vision, LIDAR, and solar cells need to be able to be fine tuned and improved in a very short amount of time if the need arises.

2.5 Ethical, Health, and Safety Constraints

Given the relatively innocuous nature of our project, we see no serious ethical issues which need to be taken into consideration prior to development of this project, or which would be likely to present themselves in the course of the buggy's construction. Given the slow speed limited by both the specifications and the hardware, our buggy holds no real capability to be used in a military capacity, or as a weapon of any kind, as it is explicitly designed to avoid collision with objects. If anything, our buggy's primary use lies in its utility as a tool of leisure by allowing people to travel along the beach without any strenuous physical activity, which would be especially helpful for the elderly or the physically disabled. It could also be used to assist people who have been injured on the beach by allowing them to be transported to distant medical professionals. In both cases, there are no serious ethical considerations that would complicate the design of the buggy.

A major concern of this project is safety, specifically with regards to the people on the beach and the wildlife. As the vehicle will be driving autonomously, it is paramount that the image recognition systems be able to detect any moving object from a safe distance and steer well clear of it. As a result, we will incorporate both a remote and a manual 'kill switch' into our design, in order to ensure that the vehicle can be stopped in the event that either an error occurs with our prototyping, or a passenger riding in the vehicle feels that they are unsafe and need to bring the buggy to a stop.

Given the populated nature of the beach, and the electronics on board, it's also paramount that the buggy does not drive into the water. This is a massive potential safety hazard, and in order to counteract this we intend to program our LIDAR system to identify the scattered, haphazard point cloud it would likely receive from a large body of water like the ocean as a positive identification for a body of water, and automatically steer clear of it, providing that there is nothing in its way. This same logic will be used in order to stop the buggy from travelling away from the beach, into the mainland.

Bystanders should be made aware that the buggy is autonomous and should not be blocked or disturbed to prevent any unforeseen consequences. The buggy should also consider pedestrians that are not aware of the presence of the buggy. The buggy might trip or run over (albeit slowly) unaware pedestrians. A safe way to ensure that the buggy maintains a safe operating environment is to post watchers that can monitor the buggy and alert and pedestrians that may endanger the pedestrian, the buggy, or both.

Health-wise, the buggy does not emit any harmful elements (radioactive energy or carbon emissions). Health considerations were researched, and our team has found none.

2.6 Environmental, Social, and Political Constraints

In much the same way that we were unable to identify any legitimate ethical concerns with regards to the design of our buggy, we find no political issues which would impact our design process. Our buggy is meant as a tool to assist beachgoers, and as a result has a very apolitical purpose. The only potential effect that politics could have on our designs are the country's use of tariffs on other powers which may or may not make certain components of our design more expensive.

The buggy, powered by solar cells, has a zero-direct carbon footprint. Global environmental considerations are thus not a concern for this project. However, considerations are made for the small flora and fauna that might cross the buggy's path. Prior observation of the buggy's path of travel should be done to ensure that no endangered species or any flora or fauna are at risk of being trampled or hit by the buggy. Should there be environmental concerns that may cross the buggy's path, steps will be done to ensure that the buggy does not endanger them by either avoiding them or by the manual intervention of the team.

As the buggy is designed from well-accepted standards and methods, the buggy poses no threat to the fabric of society and does not have any opposition from any social group. Social considerations were researched, and our team has found none.

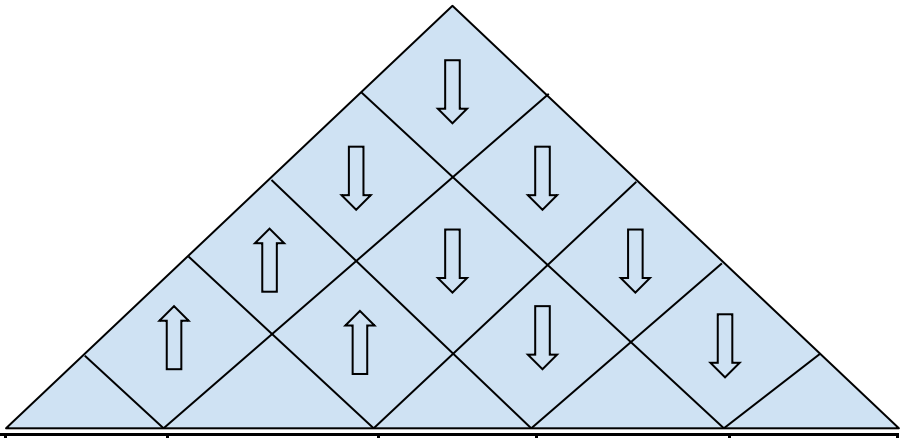
2.7 Manufacturability and Sustainability constraints

Manufacturability constraints limit our procurement of materials and parts to ones that can be manufactured either by ourselves or through a third-party. Should a part prove to be something that we cannot manufacture ourselves or if it is more efficient to be purchased, it will be purchased. Our designs use parts that are commonly and easily obtainable and thus pose no manufacturability constraint.

Our parts also use commonly available materials and poses no sustainability issues in both manufacturing and the sustainability of ecological balance.

2.8 House of Quality

After considering the engineering requirements we were tasked to work around, we constructed a house of quality. In constructing the House of Quality, we first had to determine which areas of our design were the most vital, and which aspects of the design conflicted with each other the most. We then added costs and rewards based on these factors, as shown in Figure 2.



		Durab-ility	Structural Integrity	Speed	Power Use	Cost
		+	+	+	-	-
Object Detector	+			⇊	↑↑	↑↑
Cost	-	⇊	↑↑	↑↑	↑	↑↑
Safety	+	↑↑	↑	⇊	↑	↑↑
Weight	+	⇊	↑↑	⇊	↑↑	↑
Targets for Engineering Requirements		Able to travel for 20 miles	Can carry a minimum of 120 lb	Max of 3 MPH	Self sufficient for 8 hours min.	Max of $2,000

Figure 2 - House of Quality

2.9 Objectives

This project has many objectives set forth by the client, Duke Energy. Namely, they are: To create a vehicle capable of traversing a beach while autonomously detecting moving and inanimate objects; To have the buggy travel at a maximum speed of 3 miles per hour, travel 10 miles, and carry a weight of approximately 120 lbs; To have it run entirely on solar power; To do no harm to the environment, or any beachgoers; and to be produced with a budget of $2000. These design goals ensure that the buggy can be a positive addition to the beach going experience and can demonstrate the viability of low-cost Lidar and solar systems for autonomous transportation purposes.

A key feature of the buggy is its ability to automatically detect objects and move around them, and for that purpose we chose to incorporate a Lidar system into our design. While other autonomous vehicles chose to incorporate traditional rangefinders into their designs as a way of detecting objects, the ability of Lidar to generate a point cloud and recognize individual objects makes it a superior, if more complex, choice. It also allows us to continuously scan an entire field of view, as opposed to simply measuring distances along a single vector. In the interest of ensuring safety, we have also opted to incorporate a redundant form of image recognition in the form of stereoscopic vision, by way of two cameras placed at the front of the buggy. Through the use of both the Lidar and the stereoscopic vision, we can be certain that the buggy will not collide with any people or objects, assuming all components are functional.

In the interest of safety, several design decisions have been implemented. The primary safety feature will be several ‘kill switches’ placed in easily accessible areas of the buggy. One will be reachable by the passenger, another on the side of the buggy so as to be available to a passerby, and a remote ‘kill switch’ available to us as we monitor the buggy. For safety reasons, the maximum speed of the buggy is also limited to 3 miles per hour, so as to give us time to react in the event of some unforeseen design failure.

The primary purpose of this project is to create a novel mode of transportation that will stoke the public interest in the capabilities of both solar power and autonomous navigation. To this end, we intend to have a large solar panel on the rear of the buggy, which will provide power into one of two on-board batteries. The batteries will power the motor and electrical systems and will each alternate between powering the buggy and receiving a charge from the solar cells. The solar panel will also be outfitted with servos, which will slightly adjust it’s pan and tilt until it is maximizing the exposure of the solar cells to the sun. This will be done by taking two smaller solar cells and placing them in a wedge formation, allowing light to be incident on both faces. When both cells report an equal amount of light, that will indicate an optimal angle for the larger panel.

3. Initial Research

The following section contains a description of the research each of the team members has done in preparation for the design phase of the project. This includes a discussion on the fundamental, conceptual aspects of some key features of the project, such as solar power, Lidar, stereoscopic vision, electric motors, and so on.

3.1 Mechanical Components

This section details the research done into the mechanical engineering aspects of the buggy, such as the materials used to create the chassis and the suspension system. These components are as vital to the successful operation of the buggy as the electrical components, and so an extensive amount of research was done in order to determine the most effective solution to each individual concern.

3.1.1 Chassis

There are a couple main concerns when designing this chassis: the structure and the material properties. The structure itself has multiple components that must be designed around. The chassis will need to hold all of the components, its cargo, withstand the forces coming from the driving surface, and withstand the torque from the drivetrain. Another design topic is arrangement of the chassis members. The chassis must hold photovoltaics, wiring, computers, sensors, motor, drivetrain, suspension, and seat a 120 lb. human within reasonable size. As long as all goals are met, smaller and lighter will typically be better. When a vehicle is made more compact, transport is easier, but serviceability maybe sacrificed. With tighter arrangements of components, to access one-part others might need to be removed in the process. When making a vehicle lighter, it will put less stress on mechanical components and the power of the motor will be used more efficiently.

The electrical properties of the chassis material come into play when electrical components come into contact with the chassis. Most vehicles use a conductive chassis to utilize it as an electrical ground around the vehicle. High end vehicles might use some fibrous composite, in which case a different grounding system must be used. However, a composite would not be useful for this buggy because it is not conductive and expensive. So, for this buggy steel or aluminum are being considered. Steel is inexpensive, works as a ground, and has high strength. On the other hand, steel is very heavy which require stronger motors. Aluminum is light, even higher conductivity than steel, but more expensive.

Most chassis, whether it is a buggy or automobile, do not typically fail without being in a crash. Manufacturing a chassis is almost always welded together. Machining is not feasible because chassis are too large. When a weld is done properly, it should retain the strength of the member material from one member -- through the weld -- and to the other member. Welding provides strength in all directions instead

of bolts, which vary in yield strength depending on the direction of forces act upon them. There are uncommon cases where the top mounting areas, of the rear subframe and/or spring and strut assemblies, will fatigue causing cracks or mushrooming. Depending on region and type of use, rust and corrosion can form. This is a very common occurrence in snowy regions because the salt on the roads corrode not only the chassis but any ferrous components on the vehicle. In the buggy's case, there will be salt and moisture which accelerate corrosion. Fortunately, this is a negligible issue for such short run time.

3.1.2 Suspension

The suspension will include control arms, swing arm, shocks, and springs. A spring will mount between a lower control arm and the chassis. The weight of the buggy partially compresses the springs when static. If the buggy encounters a bump under one tire while in motion, the spring will compress, and the chassis will remain more level than if there was no suspension. If the buggy encounters a hole under one tire while in motion, the tire will dip inside as the spring extends providing the chassis with stability. The shocks will prevent excessive oscillation of the springs. While this design would be ideal for performance, the budget could be used in more important areas. So, one design in the works for the rear is a monoshock, swing arm, and solid-rear-axle setup. Figure 3 below shows the UFP Baja from Brazil with the same setup that might be implemented [1]. As it seems, it is very simple and cost effective. Considering the 3 MPH speed limit and relatively smooth terrain, there is not much of a performance loss. So, this design is a good candidate.

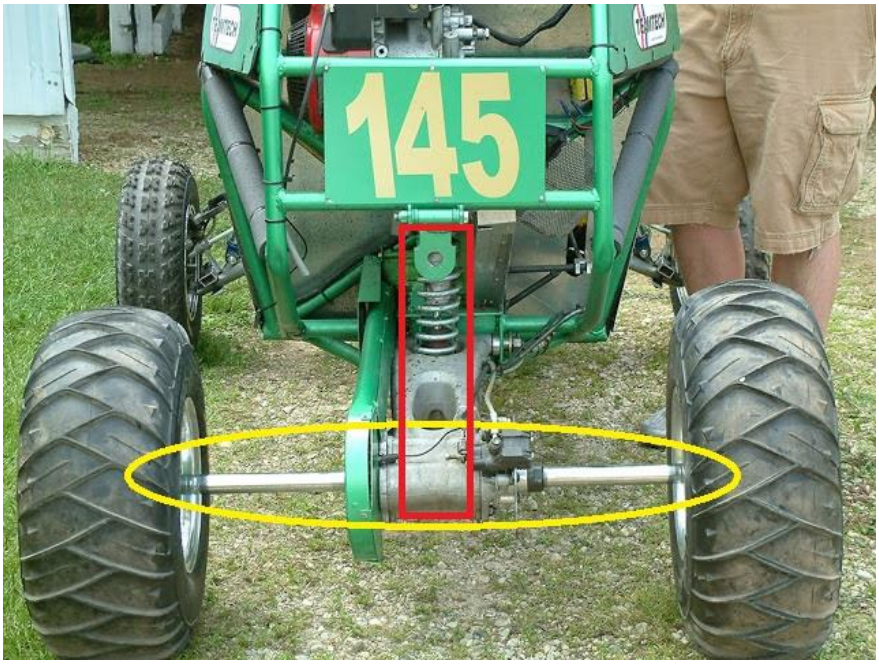


Figure 3 - Solid-rear-axle (yellow oval), monoshock (red rectangle), and swingarm design. *Courtesy of Pedro_UFPBaja, forums.bajasae.net.*

3.1.3 Drivetrain

Initially, the drivetrain would be a belt or chain running from the shaft of the motor to the rear axle(s). Instead of the price and weight of an independent rear end, a solid-rear-axle (SRA) might be used. A chain will likely be used over a belt because the chain is much cheaper. The belt would last longer with sand exposure, but the duration of service is short and one time. So, endurance of the chain will likely perform as needed. Figure 4 below shows the independent axles which can be compared to the SRA shown in Figure 3. If the SRA is used, there will have to be design and analysis comparing a steel versus an aluminum axle and diameters. This is done to assure torque from the motor can be withstood. Given the beach as the surface being driven on, it would be helpful to not have a slipping differential to keep from getting stuck. The looseness of the sand will almost entirely ease the binding of tires from a locked differential. Since there is no longer a significant need for a conventional or limited-slip differential, the buggy does not need a differential at all. Now, the buggy will be simpler, lighter, and cheaper. Another possible design is differential steering where typical axles are not necessary. In this case, each drive wheel is electronically controlled, providing power as well as the ability to steer. Just an axle beam as part of the swingarm setup will suffice to tie components together. Since electric motors have a very broad range of RPM of usable torque, a transmission is not necessary.

Figure 4 - Baja with independent axles (yellow circles). *Courtesy of Pedro_UFPBaja, forums.bajasae.net.*

3.1.4 Frame Structure and Material Selection

To begin with, a trend that is observed when scrutinizing different low budget beach buggy models is that the frame merely consists of a few metal tubes arranged to resemble a truss and welded together to successfully encase and protect the internal components, such as the engine, and the passengers aboard. The frames composing these specialized sand vehicles constituted the exterior appearance, possessing virtually no body panels to enhance the vehicle's aesthetic. While it is possible to locate elegant beach, buggies possessing decorative body panels, convenient windows, and functional doors, these amenities are not essential to building a competent beach roaming vehicle, adding unnecessary costs and mass to the buggy. A prime example of a beach buggy showcasing this simple, yet functional patterned metal frame is shown below in Figure 5 for reference.

Figure 5 - A vehicle composed of a simple strutted frame. [2]

A popular frame, shown in the image above, utilized in the construction of a sand buggy is known as a space frame, which is a three-dimensional frame structure formed from multiple struts arranged in a triangular pattern that effectively converts bending moments into tension and compression forces due to its specific geometric arrangement [3]. The mechanical function of the triangular pattern of a space frame is demonstrated and elaborated upon below in Figure 6.

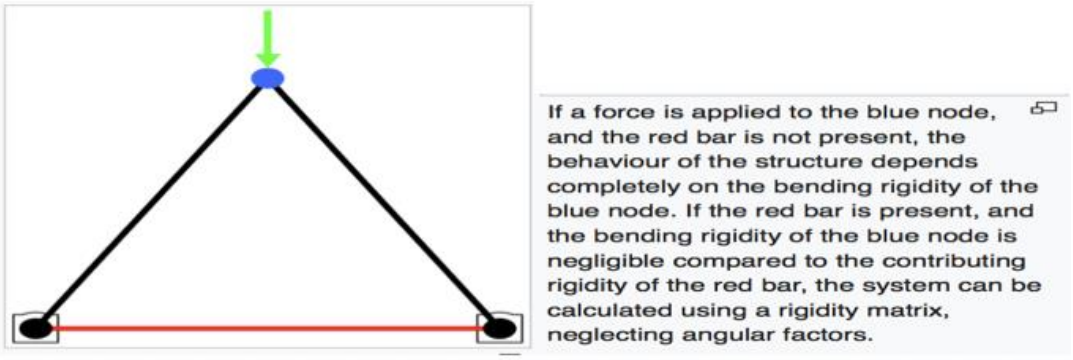


Figure 6 - The function of a space frame explained. [4]

A space frame structure contributes the necessary rigidity to ensure the safety and sturdiness required by a beach buggy while also remaining lightweight and cost effective, two qualities desired when recalling the relatively sparing budget and the requirement that this rover remain completely powered by solar energy. Without a doubt, the students must consider this form of patterned frame, and similar arrangements, when constructing the beach buggy.

Furthermore, the metal frame must be composed of a sturdy, lightweight, and inexpensive metal that can withstand the loads generated by the passengers, required to be specifically 120 pounds for this task, and the internal components. Common properties that describe a metal's suitability for this task include:

- Density- the ratio of a metal's mass to its volume
- Modulus of Elasticity- the ratio of a given tensile stress applied to the produced stress
- Yield Strength- the maximum stress a metal can withstand before plastically deforming
- Ultimate Tensile Strength- the maximum stress a metal can withstand before fracturing

The students researched these specific properties of common metals utilized in the formation of metal frames and the data is displayed in Table 2 below [5].

Metal	Density (g/cc)	Modulus of Elasticity (Gpa)	Yield Strength (Mpa)	Ultimate Tensile Strength (Mpa)
Aluminum	2.6989	69	95	110
Steel	7.75-8.05	180	502	860
Iron	7.87	210	120-150	180-210
Copper	8.96	117	70	220

Table 2: A table showing various properties of common metals

The table shown above states that aluminum is the lightest metal of the group, but steel possesses a higher Modulus of Elasticity. The students must analyze the data carefully and select the metal best suited to form the frame of the beach buggy.

3.1.5 Beach Buggy Tires

Moreover, an additional defining characteristic of a beach buggy that the students noticed are their massive, uniquely shaped wheels. A beach buggy needs to be equipped with special blade tires in the front, which possess a blade profile in the center of the wheel to enhance flotation above the sandy terrain and increases the maneuverability and traction of the beach buggy. In addition, the beach vehicle requires wide paddle tires, named after the paddle shaped rubber pattern that they possess, in order to provide elite traction [6]. These blade and paddle tires are shown below in Figure 7. The students must utilize these specialized tires to produce an efficient beach vehicle.

Figure 7 - A blade tire, left, and a paddle tire, right, shown [6]

3.1.6 Motors

When it comes to electric motors, two main types will be considered: DC (direct current) and AC (alternating current). First though, it is best to understand the parts of a simple motor. These include:

- Armature or rotor (electromagnet)
- Commutator
- Brushes
- Axle
- Field Magnet (permanent magnet)
- DC power supply

Magnets and magnetism is the driving factor in electric motors. The attracting and repelling forces are utilized to create rotational motion.

The guidelines of this challenge involve the team to be able to transport 120 lbs. a distance of 10 miles at a max speed of 3mph. Due to these requirements, there are certain aspects of the vehicle that should be calculated to determine the size of the motor that will be needed; such as torque required, rpm, voltage and amps.

1. Figure out the number of revolutions of the tire per mile. Circumference equals distance per 1 revolution. Convert tire circumference to revolutions per mile.
2. Calculate wheel rpm for desired speed.
3. Calculate estimated weight of vehicle and determine power required to operate at desired speed continuously.
4. Determine the power needed for initial acceleration.
5. Sum continuous power with acceleration power to get peak power.
6. Calculate wheel torque.
7. Choose the gearbox/differential ratios then divide the torques by gear ratio to get the required motor torques. Multiply the wheel rpm by the gear ratio to get required motor speed.
   [8]

The two options for electric motors are either direct current (DC) or alternating current (AC). Alternating current describes the flow of charge that changes directions periodically, whereas DC provides a constant voltage or current (Most electronics that use batteries relies on DC). [9]

Type of Motor	Advantages
AC	● Faster acceleration ● Higher RPM rates, generate higher top speed ● More power for hill climbing ● Lower maintenance costs ● Improved downhill braking power
DC	● Widespread availability ● Easy installation ● Cheaper ● Offers all of its toque from standstill ● Ability to overdrive

Table 3 - A comparison of AC and DC motors

The option to use AC motors is there for the team, but their advantages do not apply to our project. Opting out of AC leaves us with the DC motor as the preferred choice. Within the DC family though, there are two main motor categories: Brush and brushless. Brushless motors are usually much better than their counterpart. They are more efficient, have a higher power output, quieter, have a longer lifespan, and are overall more stable. The optimal choice, due to the apparent advantages and also previous controlling knowledge, will be the brushed DC motor. The characteristics of a Brushless DC motor is displayed below in Figure 8.

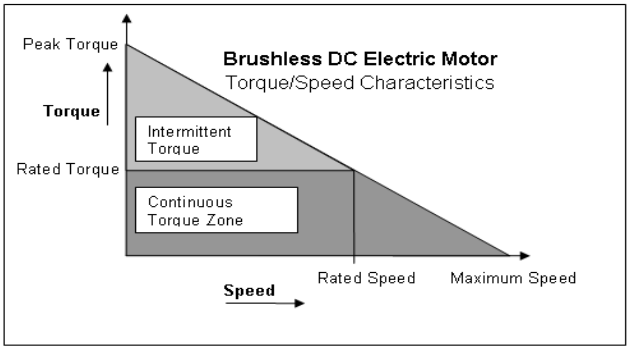


Figure 8 - Brushless DC Motor Characteristics [7]

When considering obtainable options for the buggy's motor, it appears that Go-Cart motors will be the easiest to purchase. Depending on possible car designs, like whether the buggy will have a differential or not, will determine the amount of motors needed for the vehicle. The average cost of motors ranges from $100-$400.

Table 4 below shows a comparison between a brush and brushless motor.

	AmpFlow E30-400	MY-1020	NPC4200	Motenergy ME-0909
Motor Type	Brushed	Brushed	Brushed	Brushed
Weight	5.9 lbs.	9.03 lbs.	15.7 lbs.	24 lbs.
Voltage	12-24V	12-24V	24-36V	24-48V
Peak current	266A	26.7A	470A	300A
Peak Torque	11Nm	1.91Nm	N/A	N/A
Peak Horsepower	2.1hp	0.7hp	3.8hp	15hp
RPM	5700rpm @24V	2500rpm @24V	3400rpm	4850rpm
Price	$109	$79.99	$339	$385
Efficiency	79%	75%	N/A	95%

Table 4 - Brush and brushless motor comparison

Although a brushless motor is more efficient and requires less maintenance, a brushed motor will be the most economical choice for this application. As a result, the MY-1020 will be the motor that will power the solar beach buggy.

3.1.7 Batteries

In the process for designing an autonomous solar powered beach buggy, it is of utmost importance to develop great understanding of batteries and the uses they can have in the development of designing a beach buggy. This beach buggy will need to traverse a 10-mile stretch, so it will be necessary to have the optimal battery for the journey. This technical memo will be used to explain how batteries can be used to store energy collected from the sun and how batteries can be used to power motors.

In order to understand how a vehicle can be run using the power from the sun, it is first necessary to know how to collect the energy from the sun and converting that energy into electricity. This process of collecting the sun's energy is accomplished using photovoltaic cells or PVCs. PVCs are the components in solar paneling that convert the sun's energy into electricity. PVCs are made up of semiconductors that are absorb the light and are usually made out of silicon. The energy from the sun then frees the electrons in the semiconductors, when then creates the flow of electrons. The flow of electrons then generates the electricity that powers the battery or powers the motor directly.

A key element in this Solar Beach Buggy Challenge is understanding batteries. The primary functions of batteries in a PV system are to store electrical energy when it is produced by the PV array, to supply power to electrical loads at stable voltages and currents, and to supply high peak operating current to appliances.

Each battery type has its individual strengths and weaknesses. Table 5 below compares some of the characteristics of the battery types in consideration. In PV systems, lead-acid batteries are most common due to their wide availability in many sizes, low cost and well understood performance characteristics. Nickel-cadmium cells are used in low temperature applications, but their high costs limit their use in most PV systems.

Battery Type	Cost	Deep Cycle Performance	Maintenance
Flooded Lead-Acid			
Lead-Antimony	low	good	high
Lead-Calcium Open Vent	low	poor	medium
Lead-Calcium Sealed Vent	low	poor	low
Lead Antimony/Calcium Hybrid	medium	good	medium
Captive Electrolyte Lead-Acid (VRLA)			
Gelled	medium	fair	low
Absorbed Glass Mat	medium	fair	low
Nickel-Cadmium			
Sintered-Plate	high	good	none
Pocket-Plate	high	good	medium

Table 5 - Battery Types: Cost, Deep Cycle Performance, Maintenance

Lead-Antimony batteries are a type of lead acid battery which use antimony (Sb) as a primary alloying element with lead in the plate grids. The advantages of these types of batteries include providing greater mechanical strength than pure lead grids, and excellent discharge and high discharge rate performance. Lead-antimony grids also limit the shedding of active material and have better lifetime than lead-calcium batteries when operated at higher temperatures. The disadvantages of lead antimony are a high self-discharge rate, and as the result of necessary overcharge, require frequent water additions depending on the temperature and amount of overcharge. Most lead-antimony batteries are flooded, open vent types with removable caps to permit water additions. They are well suited to application in PV systems due to their deep cycle capability and ability to take abuse, however they do require periodic water additions. The frequency of water additions can be minimized by the use of battery designs with excess electrolyte reservoirs.

Lead-calcium batteries are a type of lead-acid battery which use calcium (Ca) as the primary element with lead in the plate grids. The advantages to these types of batteries include providing greater mechanical strength than pure lead grids, low self-discharge rate, and reduced gassing results in lower water loss and lower maintenance requirements than for lead antimony batteries. The disadvantages include poor charge acceptance after deep discharges and a shortened battery life at higher operating temperatures.

Lead-Antimony/Lead-Calcium hybrid are typically flooded batteries, with capacity rating of over 200 ampere-hours. This design combines and advantages of both lead-calcium and lead-antimony design, including good deep cycle performance, low water loss and long life.

Captive electrolyte lead-acid batteries are another type of lead-acid battery, and as the name implies, the electrolyte is immobilized in some manner and the battery is sealed under normal operating conditions. Electrolyte cannot be replenished in these battery designs; therefore, they are intolerant of excessive overcharge. Captive electrolyte lead-acid batteries are popular for PV applications because they are spill proof and easily transported, and they require no water additions making them ideal for remote applications where maintenance in infrequent or unavailable. However, a common failure mode for these batteries in PV systems is excessive overcharge and loss of electrolyte, which is accelerated in warm climates. For this reason, it is essential that the battery charge controller regulation set points are adjusted properly to prevent overcharging. The benefit of captive or immobilized electrolyte designs is that they are less susceptible to freezing compared to flooded batteries. In the US, about one half of the small remote PV systems being installed use a captive electrolyte or sealed batteries. The two most common captive electrolyte batteries are the gelled electrolyte and absorbed glass mat designs (AGM). The advantages and disadvantages of battery types are further displayed in Table 6.

Battery Type	Advantages	Disadvantages
Flooded Lead-Acid		
Lead-Antimony	low cost, wide availability, good deep cycle and high temperature performance, can replenish electrolyte	high water loss and maintenance
Lead-Calcium Open Vent	low cost, wide availability, low water loss, can replenish electrolyte	poor deep cycle performance, intolerant to high temperatures and overcharge
Lead-Calcium Sealed Vent	low cost, wide availability, low water loss	poor deep cycle performance, intolerant to high temperatures and overcharge, can not replenish electrolyte
Lead Antimony/Calcium Hybrid	medium cost, low water loss	limited availability, potential for stratification
Captive Electrolyte Lead-Acid		
Gelled	medium cost, little or no maintenance, less susceptible to freezing, install in any orientation	fair deep cycle performance, intolerant to overcharge and high temperatures, limited availability
Absorbed Glass Mat	medium cost, little or no maintenance, less susceptible to freezing, install in any orientation	fair deep cycle performance, intolerant to overcharge and high temperatures, limited availability
Nickel-Cadmium		
Sealed Sintered-Plate	wide availability, excellent low and high temperature performance, maintenance free	only available in low capacities, high cost, suffer from 'memory' effect
Flooded Pocket-Plate	excellent deep cycle and low and high temperature performance, tolerance to overcharge	limited availability, high cost, water additions required

Table 6 - Battery Type: Advantages and Disadvantages

Table 7 below shows a list of batteries of different type that is being considered.

Type	Brand	Weight (lbs.)	Dimensions LxWxH(in)	Capacity (AH)	Voltage (V)	Price ($USD)
SLA	ExpertPower	23.2	7.7 x 5.2 x 6.3	33	12V	$64.99
FLA	USBattery US12VXC	86	13.13 x 7.07 x 11.38	155	12V	$178.87
FLA	Trojan T1275	82	12.96x7.13x11.25	150	12V	$189.95
AGM	Universal Battery	69.9	12.12 x 6.61 x 9.16	100	12V	$164.99
Gel	Renogy	66	13 x 6.8 x 9.0	100	12V	$229.99
SLA	NPP	59.5	12.1 x 6.7 x 8.2	90	12V	$162.99

Table 7 - Lead acid battery comparison

As part of the requirement, the solar beach buggy must have the ability to travel up to 10 miles; therefore, the battery must have the capacity to store a lot of charge it receives from the solar panel. The ExpertPower is the best choice for this application mainly because it is fairly inexpensive.

Once a particular type of battery has been selected, it is then best to consider how to configure and maintain the battery for optimal performance; considerations in battery subsystem design include the number of batteries in series and parallel, over-current and disconnect requirements, and selection of the proper wire sizes and types.

Batteries connected in a series circuit have only one path for the current to flow. Batteries are arranged in series by connecting the negative of the first battery to the positive terminal of the second battery. The negative of the second battery to the positive of the third battery, and so on for as many cells in the series string. The total voltage is the sum of the individual battery voltages, and the total capacity is the same as for one battery. If batteries or cells with different capacities are connected in series, the capacity of the string is limited to the lower battery capacity.

Batteries connected in parallel have more than one path for current to flow, depending on the number of parallel branches. Batteries are arranged in parallel by connecting all the positive terminals to one conductor and all the negative terminals to another conductor. For similar batteries connected in parallel, the voltage across the entire circuit is the same as the voltage across the individual parallel branches, and the overall capacity is sum of the parallel branch capacities.

Battery manufacturers recommend that their batteries be operated in as few parallel strings as possible. If too many parallel connections are made in a battery bank, slight voltage differences between the parallel strings will occur due to length, resistance and integrity of the connections. The results of these voltage differences can lead to inconsistencies in the treatment received by each battery in the bank, potential causing unequal capacities within the bank.

3.1.8 Frame Materials

To build a solar powered vehicle, a desert buggy needs to be purchased. The weight of the buggy must be kept in mind thought the entire design process. Whether it's made from plastic, wood, different alloys, mild steel or manufactured types of steel durability and safety is the top priority. As seen in the Figure below, HREW has a yield strength of 30,000 to 45,000 psi and an ultimate strength of 42,000 to 62,000 psi. [10] DOM ranges from 60,000 to 70,000 yields and 70,000 to 80,000 ultimate strength. Most of the metals used to build the frame of the buggy are formed and welded by using a tube bender and tig/mig welder. Any unnecessary parts can be removed from buggy leaving just a unibody shell that fused body, fenders and frame. Figure 9 below shows the differences between

potential frame materials. Heavy wheels, batteries location, electric motor, and solar panels must all be considered when designing a buggy. The chassis composes a major issue because of its metal construction, narrow front, and limited spacing for batteries, charger and controller.

Figure 9 - Car Frame Materials

3.1.9 Steering

In the rack-and-pinion system, a small pinion (gear wheel) is inside a housing at the base of the steering column. Its teeth mesh with a straight row of teeth on a rack - a long transverse bar. Turning the pinion makes the rack move from side to side. The ends of the rack are coupled to the road wheels by track rods. This system is simple, with few moving parts to become worn or displaced, so its action is precise. A universal joint in the steering column allows it to connect with the rack without angling the steering wheel awkwardly sideways.

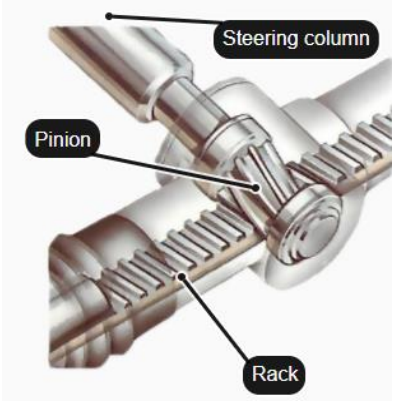


Figure 10 - Rack and Pinion System

Differential steering allows a vehicle to move on tracks by increasing or decreasing the speed at which one side of the tracks moves. This method may be more cost/labor efficient as it only requires two motors which can be used for both driving and steering compared to a rack-and-pinion system which requires a motor to drive and a separate system for steering.

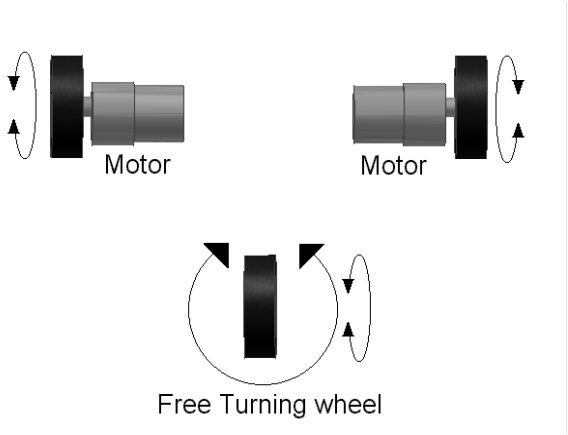


Figure 11 - Differential Steering

3.2 Electrical Components

In this section we will describe the research conducted into the electrical components which will be considered for our final design. Much like the mechanical components, combining the proper electrical components in their proper configuration is vital to the successful construction and completion of this project. This section will contain a discussion of the systems which will be used in order to survey the environment, communicate the correct signals to the appropriate devices, and interact with the motors in order to move the buggy in the necessary manner.

3.2.1 Solar Cells

In this section we will discuss some of the operating characteristics of solar cells, as well as the primary distinctions between different types of solar cells, in order to be able to select the appropriate cell for our needs. Solar cells form a central tenet of our design, and as such it is necessary to select a cell that most adequately aligns with our goals in regard to power output, efficiency, heat tolerance, durability, and price.

The basic operating principle of a solar cell relies on the use of a photodiode to capture light emanating from the sun and transform it into a current. This is possible due to a scientific phenomenon known as the photoelectric effect, which describes the emission of electrons or other free carriers when light shines upon an object. This occurs because the light passes through an area of the material known as the depletion region, wherein a photon of light is absorbed by an electron, which

causes it to escape the depletion region and escape from the semiconductor as current.

A photodiode is a semiconductor device, meaning that they are typically made of materials such as silicon or gallium. These materials are then doped by other materials such as boron or phosphorus, giving them excess amount carriers, which refers to either electrons or holes. These materials are then referred to as either n-type or p-type, respectively. When placed in contact with each other, these materials create a 'depletion region' where a built in electric field causes the movement of excess carriers. As such, when a photon of sufficiently high energy is incident on the depletion region, it creates an electron-hole pair, which are then carried off in different directions by the electric field, generating a photocurrent. Depending on how the photodiode is biased, it can also generate a voltage. However, a solar cell is operated under zero bias conditions.

There are three different types of solar panels that are typically used for commercial use: Monocrystalline silicon, polycrystalline silicon, and thin-film solar panels. There are also more niche types of solar panels, such as translucent cells and high efficiency cells, which are capable of reaching efficiency rates as high as 40%. Although research is constantly being done on these panels and their price is steadily decreasing, these solar cells are not yet commercially viable, and so will not factor into our designs as a serious option.

Monocrystalline silicon cells are created using sheets or cylindrical wafers of silicon ingots. It is produced via a manufacturing technique called the Czochralski process, which is a long, technical process which gives the monocrystalline silicon cell a higher price than some alternative panels. As a result, however, they boast a relatively high efficiency rate, in the range of 15-20%. They are also space efficient and boast a better performance than polycrystalline silicon cells in low light conditions, as well as generally low failure rates. In particular, they suffer noticeably less degradation in efficiency when exposed to high temperatures than other types of solar cells. This makes them a viable candidate for our design.

Polycrystalline silicon cells do not use silicon ingots or the Czochralski process, instead being produced by pouring molten silicon into square molds which are then cooled and made into wafers. As a result, they prove to be a cheaper, more cost-effective alternative to monocrystalline silicon cells, although they prove to have a slightly lower average efficiency, at around 15%. This also makes them less space efficient than some alternatives.

Thin-film solar cells are created by depositing multiple layers of photovoltaic materials onto a substrate. They can also be further classified by the photovoltaic material deposited onto the substrate, namely: Amorphous Silicon (a-Si), Cadmium Telluride (CdTe), Copper Indium Gallium Selenide (CIS/CIGS), and organic photovoltaic cells (OPC). Thin-film solar cells are a fairly new, rapidly developing form of solar cell, and as a result they have among the lowest operating efficiency, achieving a maximum efficiency of around 7%. However, the production

process is relatively simple, meaning that thin-film solar cells have a significantly lower cost than more traditional solar cells. They are also more flexible and suffer less negative effects as a result of extreme temperatures.

From our research, it was determined that a monocrystalline solar cell would be the most appropriate for our needs. Despite being among the more expensive options, it proved to have the highest efficiency in converting incident sunlight into useful energy, which is of vital importance to our project, as the solar cells are responsible for powering the entire buggy. This efficiency also makes it the most space-efficient, which is also a key aspect of our decision as every foot of space on the panel needs to be accounted for in the design of the buggy's chassis. Monocrystalline solar cells have also demonstrated relatively little drop in efficiency in response to changes in temperature, which is of special concern to us, seeing as the buggy needs to operate continuously on a hot, humid, and crowded Florida beach.

3.2.1.1 Advantages

Solar energy boasts many significant advantages over forms of energy creation such as fossil fuels. It's a renewable, clean energy source that is readily available from anywhere in the globe and can be harnessed quickly and easily, using equipment that requires relatively little in the way of maintenance. As research progresses and energy needs grow, solar energy continues to look more and more like a viable, and in many ways necessary, source of power.

Solar energy can prove to be a useful energy source in volatile areas where complicated, high maintenance machinery is not viable, due to its ability to be decentralized, off-grid and operate with next to no management. It's especially useful for areas near the equator, where sunlight is both prevalent for most of the year and especially intense. The standalone nature of a solar panel station also insulates it from difficulties involving changing economic conditions, such as might heavily affect energy production through the use of fossil fuels. Not only are solar cells useful for small scale power generation, they're also proving to be a viable source of large scale energy production, with solar farms generating over 50 GW of power in the U.S. alone.

With regards to our needs, a monocrystalline solar cell has many advantages over systems utilizing traditional battery power or fossil fuels. The nature of our buggy makes it ideal for emergency situations, wherein it could allow beach patrons to reach safety when they would otherwise be rendered unable to move. In such a crisis, it would be catastrophic if the buggy were to suddenly and unexpectedly stop moving due to a lack of sufficient battery charge or an empty gas tank. A total reliance on solar energy ensures that, provided sufficiently sunny weather conditions and regular maintenance to ensure all parts of the buggy are in proper working order, our buggy would be able to operate from first light at dawn to sunset without ever having to slow down or recharge.

3.2.1.2 Disadvantages

Solar energy faces a few limitations. For one, it is extremely limited by location, as only regions which experience large amounts of unobstructed sunlight on a regular basis can generate a useful amount of power. It's also limited by the fact that it is only capable of generating power during the day, and night time activities need to either be conducted on battery power or through the use of other energy sources. It also requires a significant allocation of land which could potentially be used for other purposes. Moreover, although it is advancing rapidly, solar panels remain a costly investment. These are issues that will disappear with time as solar cells become more efficient, both in terms of cost and power generation, allowing them to occupy less space.

For our purposes, these are both reasonable but not insurmountable obstacles. While the buggy is heavily dependent on optimal weather conditions, Florida is known for an abundant amount of sunshine, particularly in the summer seasons when attendance at a beach would be at their highest points. As such, the buggy would be able to operate, without incident, more often than not. This design also takes advantage of the fact that beach attendance is traditionally down during periods of unfavorable weather, meaning that, with less beachgoers in general on days of overcast or rain, the buggy is less likely to be needed during the periods in which it would have difficulty operating. As far as cost of investment goes, the system we're proposing is fairly cheap when the total amount of functionality is considered, and a minimal amount of regular maintenance would be required in order for the buggy to remain functional. The solar cells themselves have guarantees in excess of 25 years, ensuring that they could provide a massive return on investment.

As global demand for energy rises, and a public call for more renewable energy becomes lauder, solar energy is poised to become a primary source for the world's power needs. For our purposes, understanding the strengths and limitations of solar energy is vital to creating an efficient, functional design. As our buggy will be operating on a beach, ideally on a sunny day with no overcast clouds, we foresee no issue in acquiring sufficient light to power our design, only in selecting the appropriate solar cell and power configuration to make it viable for the duration of our journey.

3.2.2 Inverters

This section will discuss the various types of inverters that are commercially available, as well as their properties, benefits, and drawbacks. Inverters are a vital component for any system using photovoltaic cells. As our solar cell is only capable of putting out DC current, and we need it to operate many different, highly complex electrical components with as much efficiency as is possible, it is necessary for us to incorporate an inverter in order to convert the DC current to AC current.

Inverters are also useful in other respects. By utilizing a maximum power point tracking system, the inverter can analyze the non-linear DC current from the solar cells and output a steady, maximized power for the system. Because the solar cells follow a characteristic I-V curve for their output, the inverter is able to sample the I-V curve at the location of the maximum of the current/voltage product, in order to output the maximum power.

3.2.2.1 String Inverters

This section will discuss the basic properties of a string inverter, and how they relate to our project. Inverters can prove to be a key component to the success and efficiency of system involving solar panels, and so it is necessary that we examine all the various types that exist in order to determine which type will be able to most satisfactorily meet the needs of our system.

String inverters are among the most cost-effective inverter options available, as they have been in use for many years in solar systems. They are typically used in systems involving multiple solar panels. They essentially function by taking the combined output of many different solar cells in series and inverting the current to AC. This has major implications in terms of lowering costs and complexity of the system. However, it can also cause problems if one or more of the panels in the system becomes shaded and ceases to work, drastically reducing the amount of power produced by the solar cells. Due to this limitation, and our inability to predict weather patterns once the buggy is in motion, we will seek out an alternative type of inverter.

3.2.2.2 Micro Inverters

This section will detail the key features of the micro inverter, it's typical applications and how it compares to other types of inverters.

A micro inverter differs from the string inverter in one key area. Rather than providing a single inverter to convert the entire output of a solar system into AC current, an individual micro inverter is connected to each solar panel in the system, and is responsible for converting the output of that panel alone into AC. Due to this independence, the total output of the system is no longer capable of being severely diminished by the effects of optical blocking on one or more of the solar panels, leading to greater overall efficiency. However, as a result of the extra parts required in setting up multiple micro inverters, this can prove to be a costlier solution. As we are only interested in one or two panels, this is a non-issue, but the increase in efficiency makes it a viable choice over a string inverter.

3.2.2.3 Power Optimizers

This section will detail the construction and implementation of power optimizers in a system of solar cells, and how it compares in terms of cost, efficiency, and practicality for our purposes with the other types of inverters available to us.

A power optimizer is a device that monitors the output of a solar cell and tracks its maximum point in order to ensure that the cell is outputting at the largest amount of power it can. It does this through a process called maximum power point tracking, wherein it monitors the output of the array and adjusts the load it presents to the system in order to keep the system operating at its peak efficiency point. They are fairly inexpensive, and so can be considered as viable solutions in our design process.

A power optimizer setup can almost be thought of as a combination of a string inverter and a micro inverter. It functions by placing a power optimizer at the output of each solar panel, and then connecting all the outputs of the power optimizers in series into a central string inverter. In this way it is able to easily convert all the DC current produced by the solar panels into AC current without suffering the severe decreases in efficiency due to partial shading of one or more solar panels. Due to these properties, it can be thought of as a happy medium between the two previous inverters and sits somewhere between the two in terms of cost. For our purposes, this type of inverter seems to be ideal, as cost is a significant issue in our designs, as well as maximum efficiency in all foreseeable conditions. It would likely be a much cheaper solution to purchase two power optimizers than it would be to purchase two micro inverters, and we will only suffer a negligible amount of power loss as a result.

Table 8 below compares the different solar panels under consideration.

	Ramsond SP100	Renogy RNG-100P	Renogy	Grape Solar	Nature Power
Maximum Power (W)	100	100	100	100	100
Peak Voltage (V)	18.5	17.79	18.9	18	17.85
Peak Current (A)	5.41	5.62	5.29	5.56	5.6
Efficiency (%)	17	-	-	17.4	-
Cell Technology	Mono	Poly	Mono	Poly	Poly

Number of cells		36	36	36	36	36
Weight (lbs.)		16.5	16.5	16.5	18.11	20.4
Dimensions (in)	L	47	39.7	47	40.16	41
	W	21.63	26.7	21.3	26.37	27
	T	1.56	1.4	1.4	1.37	2
Cost (USD)		140	114.99	124.99	89.99	179.99

Table 8 - A comparison table of different Solar panels

The Grape Solar is the best candidate for this project mainly because of its price. Table 8 shows it has similar specification compared to the other panels in considerations. However, the UCF College of Engineering and computer Science has provided a free solar panel for this project. It will be used to power the beach buggy instead of purchasing one.

3.2.3 Single Board Computer

In selecting the proper hardware system in providing a backend for the software required to operate the robot, the following must be considered: its weight, its power draw, its processing power, and its architectural capabilities. The hardware should be powerful enough to be able to control and process the different sensors and imaging devices on the buggy without having too much overhead, allowing a real-time autonomous operation of the vehicle but at the same time, its power draw should be conservative in such a way as to not overdraw from the solar panels and the batteries as power might be needed elsewhere.

For the buggy, a single-board computer provides the ability to satisfy our requirements as it has the ability to have an operating system installed, along with the ability to use the programs considered in operating our robot (elaborated in the 3.3 Software Components section).

As opposed to microcontrollers that tend to be specific to only one application, A single-board computer contains the necessary, more powerful, components (especially a more powerful CPU and CPU architecture; a GPU, however, is not required) to run the software that will manage the buggy as well as the architecture that can support installing an operating system that will be the backend to our image processing and robot-control software.

Table 9 compares the different single-board computers in consideration which are: the Asus Tinker Board S (Figure 12a), the NVidia Jetson TX1 (Figure 12b), and the Raspberry PI 3 Model B+ (Figure 12c).

Figure 12a: Asus Tinker Board S

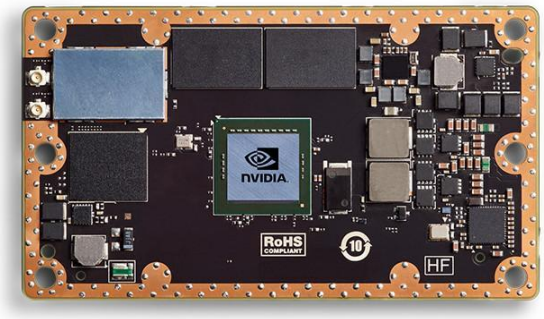


Figure 12b: nVidia Jetson TX1

Figure 12c: Raspberry Pi 3 Model B+

Single-board Computer	Asus Tinker Board S	nVidia Jetson TX1	Raspberry PI 3 Model B+
CPU	Quad core 32-bit 1.8 GHz ARM Cortex-A17	64-bit quad-core ARM A57 CPUs	1.4GHz 64-bit quad-core ARM Cortex-A53 CPU
GPU	600 MHz Mali-T760 MP4 GPU	1 TFLOP/s 256-core Maxwell	
RAM	2GB dual channel LPDDR3	4 GB LPDDR4 (25.6 GB/s)	1GB LPDDR2 SDRAM
Storage	16GB eMMC + removable MicroSD slot	16 GB eMMC	Micro SD
Networking	Gigabit LAN (not shared with USB bus)	10/100/1000Mbit Ethernet	Gigabit Ethernet over USB 2.0 (max 300 Mbps)

Wireless	Bluetooth V4.0 + EDR, 802.11 b/g/n Wi-Fi, with IPEX antenna header	802.11ac wireless LAN, Bluetooth-enabled	Dual-band 802.11ac wireless LAN (2.4GHz and 5GHz) and Bluetooth 4.2
Power Input	5V @ 2~2.5A	5.5 V-19.6 V DC (6.5 W-15 W)	5V @ 2.5A
Weight	55g	88g	
Other IO	GPIOs, SPI, I2C, UART, PWM, PCM/I2S	GPIOs, I2C, I2S, SPI	GPIO, I2C, I2S
Price	$80	$299	$35

Table 9 - A comparison of some of the single-board computers being considered

The most promising candidate for selection is the Raspberry Pi 3 Model B+ mainly because of its price and its ability to still decently run the required software implementations for the buggy. The nVidia Jetson TX1, while impressively powerful, has an MSRP of ~$300. Compared to the Asus Tinker Board S, the Raspberry Pi 3 Model B+ has been in the market longer and has more support from the manufacturer and the community, making computational hardware and software implementation less time and resource consuming.

3.2.4 GPS Tracking Unit

While Image and sensor processing alone can allow a robot to operate safely by avoiding obstacles, true autonomy can only be achieved if the robot knows where exactly it is. A GPS tracking unit is a device that allows just that.

A GPS tracking unit uses the Global Positioning System (GPS) to track the device's movements at intervals to determine its location and, when attached to the vehicle, its carrier.

GPS currently consists of 31 satellites in semi-synchronous orbits around the earth. GPS satellites continuously transmit data about their current time and position. The satellites have atomic clocks on board to keep accurate time. General and Special Relativity predict that differences will appear between these clocks and an identical clock on Earth; that is, time will run faster on GPS satellites than that of clocks on earth due to General Relativity predicting that time appears to run slower under stronger gravity. Also, Special Relativity dictates that due to the speed of the satellites relative to a clock on earth, satellite clocks will appear to run slower.

A GPS tracking unit monitors multiple satellites and uses triangulation to determine its position along with its deviation from true time. It does this by receiving GPS signals (carrier wave with modulation) that includes:

- Pseudorandom bits that is known to the tracking unit. By time-aligning a receiver-generated version and the receiver-measured version of the code, the time of arrival of a defined point in the code sequence, called an epoch, can be found in the tracking unit clock time scale
- A message that includes the time of transmission of the code epoch (in GPS time scale) and the satellite position at that time

The tracking unit measures the times of arrival of at least four satellite signals. From the time of arrival and the time of transmission, the tracking unit forms four time of flight values which are approximately equivalent to the distance between the tracking unit and the satellite. The tracking unit then calculates its three-dimensional position from this.

Two GPS trackers are in consideration: the Adafruit Ultimate GPS HAT for Raspberry Pi (Figure 13a) and the Adafruit Ultimate GPS Breakout (Figure 13b). Both of which are identical to each other except the HAT (Hardware Attached on Top) is an add-on board specifically for Raspberry Pi B+ that conforms to a specific set of rules that make life easier for users.

A significant feature of HATs is the inclusion of a system that allows the B+ to identify a connected HAT and automatically configure the GPIOs and drivers for the board. The disadvantage is that the GPS HAT takes over the Raspberry Pi's hardware UART to send/receive data to and from the GPS module so if you the RX/TX pins are used with a console cable; this HAT cannot be used in conjunction.

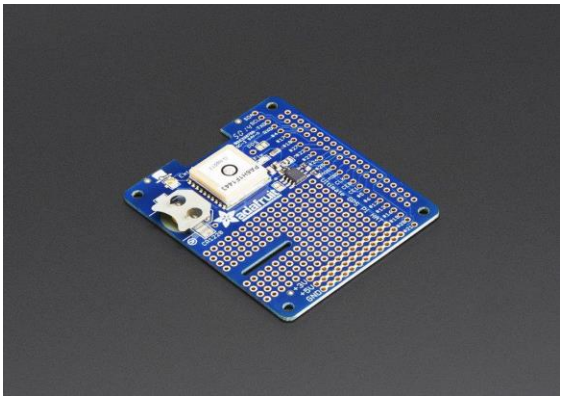


Figure 13a. Adafruit Ultimate GPS HAT Raspberry Pi

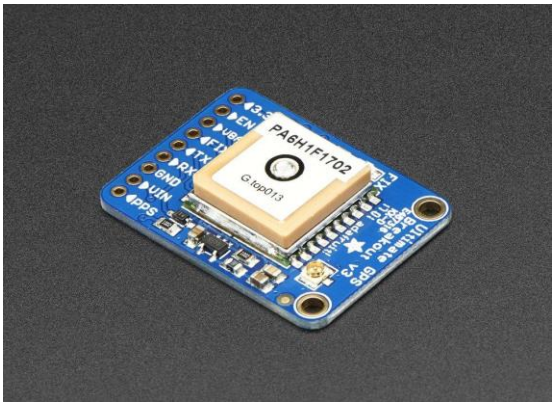


Figure 13b. Adafruit Ultimate GPS Breakout

Among their features, it includes:

- 165 dBm sensitivity, 10 Hz updates, 66 channels
- 5V, 20mA current draw
- Breadboard friendly + two mounting holes
- RTC battery-compatible
- Built-in datalogging
- PPS output on fix
- Internal patch antenna + u.FL connector for external active antenna
- Fix status LED

Table 10 lists detailed specifications for the GPS chipset.

GPS Solution	MTK MT3339
Frequency	L1, 1575.42MHz
Sensitivity[1]	Acquisition: -148dBm, cold start Reacquisition: -163dBm, Hot start Tracking: -165dBm
Channel	66 channels
TTFF	Hot start: 1 second typical Warm start: 33 seconds typical Cold start: 35 seconds typical (No. of SVs>4, C/N>40dB, PDop<1.5)
Position Accuracy	Without aid:3.0m (50% CEP) DGPS(SBAS(WAAS,EGNOS,MSAS)):2.5m (50% CEP)
Velocity Accuracy	Without aid : 0.1m/s DGPS(SBAS(WAAS,EGNOS,MSAS,GAGAN)):0.05 m/s
Timing Accuracy (1PPS Output)	10 ns(Typical)
Altitude	Maximum 18,000m (60,000 feet)
Velocity	Maximum 515m/s (1000 knots)
Acceleration	Maximum 4G
Update Rate	1Hz (default), maximum 10Hz
Baud Rate	9600 bps (default)
DGPS	SBAS(default) [WAAS, EGNOS, MSAS,GAGAN]

Power Supply	VCC： 3.0V to 4.3V； VBACKUP： 2.0V to 4.3V
Current Consumption	25mA acquisition, 20mA tracking

Table 10 - GPS Specification table

3.2.4.1 GPS Antenna

A GPS system work by receiving signal from a network of satellites, and an antenna makes it possible. Most GPS unit come with an antenna that is built-in or inside the case that works fine when they have clear view of the sky. Others come with the option to use an external antenna. While many units do not need an external GPS antenna, others, due to obstructions and interference, need an external antenna.

There are currently two types of GPS antennas today: passive and active. Passive GPS antennas simply receive the satellites signal and send it to the GPS navigation device. However, an active GPS antenna boost the power of the signal it receives with an amplifier circuit, which nearly doubles the GPS signal strength.

Active GPS antennas are expensive compared to the passive antenna and are better suited for use with a larger vehicle.

Table 11 compares the different antennas in consideration.

	ADA 2460	ADA 2461	ADA 960
Type	Passive	Passive	Active
Supply Voltage(V)	*	*	2.3~5.5
Supply Current(A)	*	*	0.0066~0.0166
Gain (dBi)	-2	1	28
Wire Length (mm)	50	50	5000
Weight (g)	2.4	5.5	*
Dimensions (mm)	9x9	15x15	41.2x38.5
Interface	IPX uFL	IPX uFL	SMA
Price (USD)	4.95	3.95	12.95

Table 11 - GPS antennas comparison table.

After careful consideration of the antennas in Table 11, the ADA2461 will be the antennas of choice for the project due to its price.

3.2.5 Wireless Networking

While the buggy is intended to operate autonomously, being able to communicate with it is a priority considering that it is important to gather the buggy's telemetry data such as its current position, speed, and energy status when not near its vicinity. It is also imperative that the buggy's operation can be aborted should it pose a threat to itself, to property, or to others.

The implementation details of which wireless technology to use depends on the bandwidth, range, and power consumption requirements for the buggy. Three technologies are considered: WiFi, GSM, and Bluetooth Low Energy.

Wi-Fi is a technology based on IEEE 802.11 standards, with 802.11ac as the latest standard as of writing. It is a trademark of the Wi-Fi alliance. Depending on standard implementation, Wi-Fi can reach a single-link theoretical throughput of at least 500 Mbit/s (802.11ac). Both 802.11n and 802.11ac have similar maximum ranges of around 230 feet. While 802.11ac and 802.11n do not differ much when it comes to range, 802.11ac provides superior throughput in longer ranges than 802.11n. Wi-Fi has a modest power consumption rate which averages around 0.5-2 W.

Global System for Mobile Communications, or GSM, provides a virtually unlimited range where the buggy can be communicated from anywhere around the world. However, GSM is subscription based and requires a subscriber identification module (SIM) for both the buggy and the human operator. Current devices primarily communicate on the subscriber network over the Long-Term Evolution (LTE) standard, which is based on GSM. LTE has a theoretical net bit rate capacity of up to 100 Mbit/s in the downlink and 50 Mbit/s in the uplink, with realistic bit rates of half of that. LTE has the most consumption rate of the technologies considered, which averages around 1 — 3.5W.

Bluetooth Low Energy, or Bluetooth LE, is a wireless technology designed and marketed by the Bluetooth Special Interest Group. Bluetooth LE is based on the "classic" Bluetooth but is not backwards compatible. Compared to Classic Bluetooth, however, Bluetooth LE is intended to provide considerably reduced power consumption and cost while maintaining a similar communication range. Bluetooth LE has a max theoretical range of 330 feet and has an over the air data rate of up to 0.27 MBit/s throughput. While its throughput is nowhere compared to Wi-Fi or LTE, it shines the most in its power consumption-- a mere 0.01-0.50 W with peak current consumption of 15 mA.

Table 12 compares the different wireless technologies discussed:

	WiFi	Bluetooth LE	GSM
Frequency	2.4 GHz or 5 GHz	2.4 GHz	600 MHz - 2300 Mhz

			(Depending on carrier and LTE band)
Throughput	500 Mbit/s (802.11ac)	0.27 MBit/s	100 Mbit/s downlink 50 Mbit/s uplink
Range	105 feet	330 feet (theoretical)	Unlimited over land
Power Consumption	0.5-2 W	0.01-0.50 W	1 — 3.5W

Table 12 - Wireless Technologies Comparison Table.

The ultimate choice in wireless technology will depend on which factor is important the most: throughput, range, or power consumption. There is an advantage of choosing Bluetooth or Wi-Fi over GSM as the former technologies are already built-in to the Single-board computers in consideration.

3.2.6 Voltage Regulator

A voltage regulator is an electronic circuit that provides a constant output voltage for an electronic design regardless of changes to its input voltage or load conditions. It is very important in any electronic circuits to provide stability and efficiency. Any fluctuation in output currents can damage sensitive electronic components. Some of the reasons why variations in output current may occur: appliances turning on and off, time of day, weather, to only name a few. In this project, many of the components in considerations require a specific voltage. Microcontrollers and sensors need to operate mainly on 3.3V and 5V while the motors require 12V to operate and current as high as 1000W.

3.2.7 Linear Voltage Regulator

Linear voltage regulators work by varying the resistance in accordance to the load which results in keeping the output voltage fixed. They are a good choice for low power applications. Even though they are simple, easy to use, and inexpensive, they are very inefficient.

From the equation 1 of the power dissipation below, the smaller the difference between the input and output voltage the better the linear regulator is. Therefore,

a linear regulator is not a good option for our application as high-power devices will be used.

Power dissipation = (input-output) Voltage * load current (equation 1)

3.2.8 Switching Regulators

In a switching regulator, a transistor is used to control the regulator. It works by rapidly switching a series of element on and off. When the transistor is off, there is no current flowing through it therefore no loss of energy; on the other hand, when the transistor is on, it is fully conducting and again there is little or no power dissipations.

Unlike linear voltage regulators, switching regulators are very efficient; whether they are conducting or not there is almost no power dissipation. They are also able to generate multiple output voltages from a single input or output higher voltages than they take in.

3.2.9 Voltage Regulator Circuits

In this design, a switching regulator will be used since it offers the advantages of higher power conversion efficiency that we are looking for. A boost converter will be used to lower the input voltage from the battery or solar panel to 5 volts and 3.3 volts to power the microcontrollers, IR sensors, LIDDAR, and cameras.

The following two tables (table 13 and 14) shows the 5V and 3.3V Switching Voltage Regulator in consideration:

	LM2596	**LM22670**	**LM43603**
Vin (Min) (V)	4.5	4.5	3.5
Vin (Max) (V)	40	42	36
Vout (Min) (V)	3.3	5	1
Vout (Max) (V)	37	5	28
Iout (Max) (A)	3	3	3
Switching Frequency (Max) (kHz)	173	1000	2200
Price	4.73	4.28	3.92

Table 13: 5V Switching Regulator Comparison Table.

	LM3940	**LM1117**	**TLV755P**
Vin (Min) (V)	4.5	15	1.45
Vin (Max) (V)	5.5	20	5.5
Vout (Min) (V)	3.2	1.8	0.6

Vout (Max) (V)	3.4	5	5
Iout (Max) (A)	1	0.8	0.5
Price	1.65	1.50	0.60

Table 14: 3.3V Voltage Regulator Comparison Table.

For the 5V, the LM2596 will be used. Although it is the most expensive of all the chips in consideration, it is very efficient and use only four components. The LM3940 will be used for the 3.3 V. Figures 14 and 15 below show the circuit diagrams for two voltage regulators which will lower the 12V solar output voltage to 5V and 3V, respectively.

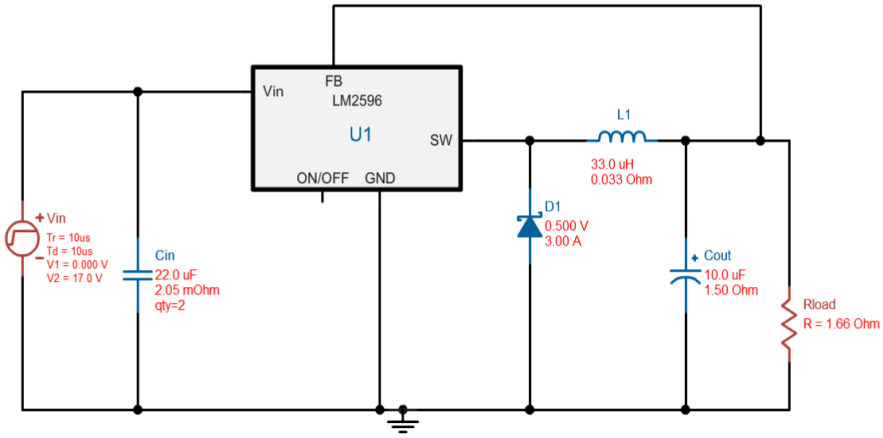


Figure 14 - Buck converter voltage regulator circuit with a 5V output. Reprinted with permission from Texas Instruments.

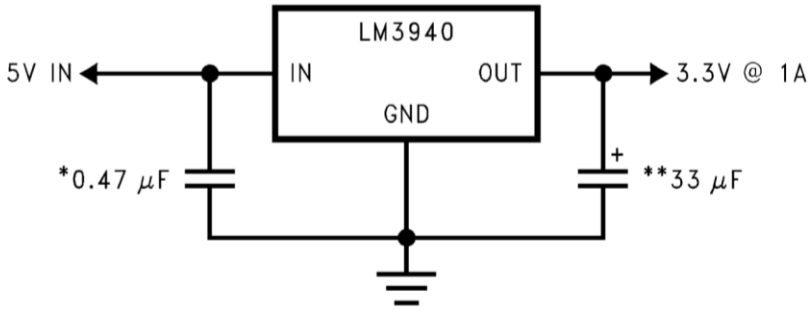


Figure 15 - Buck converter voltage regulator circuit with a 3.3V output. Reprinted with permission from Texas Instruments.

3.2.10 Motor Controller

A motor controller is an electronic device that controls the performance of a DC motor. The motor controller is necessary in this project because the microcontroller can only provide less than 1 A of current whereas the DC motors require a minimum of 32A continuous and 64A peak.

3.2.11 Motor Controller Selection

For this application, finding a motor controller capable of providing sufficient power to the motor was very important. First, it must have a dual driver motor; second, it has to be able to handle a motor with at least 500W of power. The following table will compare 3 motor controllers that can supply high current to a DC motor.

Table 15 compares the different motor controllers in consideration. Figure 16 illustrates the different motor controllers.

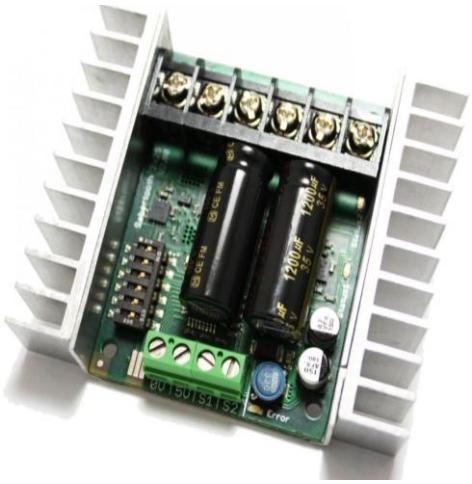


Figure 16a - Sabertooth Dual 25A

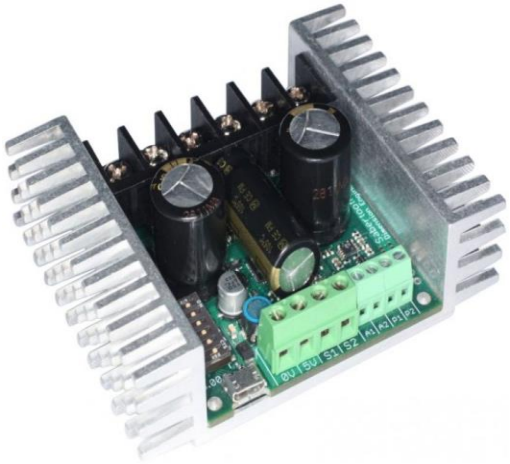


Figure 16b – Sabertooth Dual 32A

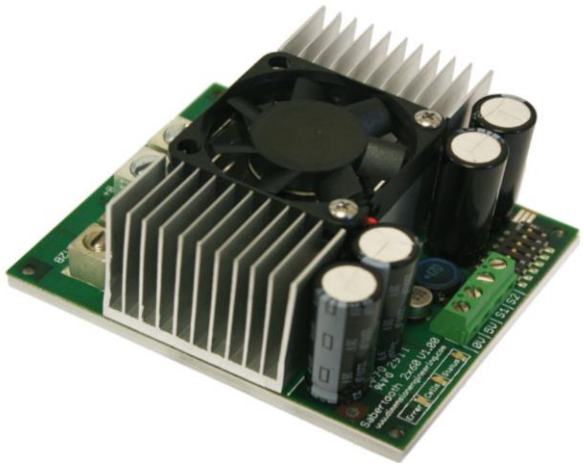


Figure 16c - Sabertooth Dual 60A Motor Driver.
Figure 16 a, b, and c are courtesy of Dimension Engineering.

	Sabertooth dual 25A	Sabertooth dual 32A	Sabertooth dual 60A
Price	$124.99	$124.99	$189.99
Weight(g)	90g	125g	240g
DC input(V)	6-30V	6-30V	6-30V
Weight Rating	300 lbs.	300 lbs.	1000 lbs.
Board Size(mm)	65 x 80 x 21 mm	70 x 90 x 25 mm	76 x 89 x 46 mm
Peak Current(A)	25A	32A	60A
Continuous Current(A)	50A	64A	120A

Table 15 - Specifications for 3 DC motor controller.

The Sabertooth dual 32A controller does not generate enough power to supply high current motors. On the other hand, the Sabertooth dual 60A is too powerful for this application. The Sabertooth dual 32A was selected because it offers the all the features needed to operate the Solar Beach Buggy. The dual 32A controller has a dual motor driver capable of supplying 32A to two motors, with peak currents up to 64A per motor. In addition to a dual motor driver, the controller comes with thermal and overcurrent protection. This is an important feature for a controller to have since an overheating board can have a negative effect on other electrical components. The motor controller allows the control of the two motors using analog, radio, serial and packetized serial control. The controller also has a build in switch mode converter that can supply power to a 5V DC source like microcontrollers, receivers, as well as up to 4 analog servos.

3.2.12 Solar Charge Controller

A battery bank is simply storing energy from the solar panel. When the battery reaches its storing capacity, it will be overcharged and get damaged. Because a solar panel is not producing energy constantly during the day, for that reason, a controller that regulates the amount of charges to the battery is needed.

A solar charge controller, as the name implies, is an electronic module which controls the amount of charges to and from the battery and regulates the performance of the battery. The charge controller offers many benefits to any systems. It charges the batteries at the correct voltage level. This helps preserves the life of the batteries. Other benefit of having a charge controller: it protects the

batteries from overcharging and over discharging; it also blocks reverse current and protect the system from electrical overload.

Solar charge controllers perform many major functions. They charge the battery bank. When the battery bank is fully Charge, they indicate it using either an LED indicator or arrays of LED's. They monitor the input and output of the battery bank; if the charge is below the minimum level, the supply to the load is cut off. However, if the battery bank is full, it will ensure the charging switch is in off position.

3.2.13 Types of Solar Charge Controller

There are two types of charge controller. One is Pulse Width Modulation (PWM) and the other is Maximum Power Point Tracking (MPPT).

3.2.13.1 PWM

PWM controllers work by lowering the input voltage they receive from the solar panel to charge the battery bank. Their charging features are very basic compared to the MPPT. When the battery voltage is low, the controller pulled down the voltage of the solar panel from 18 V to the battery voltage. Due to loss in power, PWM charge controllers are only 75-80% efficient. PWM is suitable for small systems and provide low cost solutions.

Figure 17 illustrates the wiring diagram of our design.

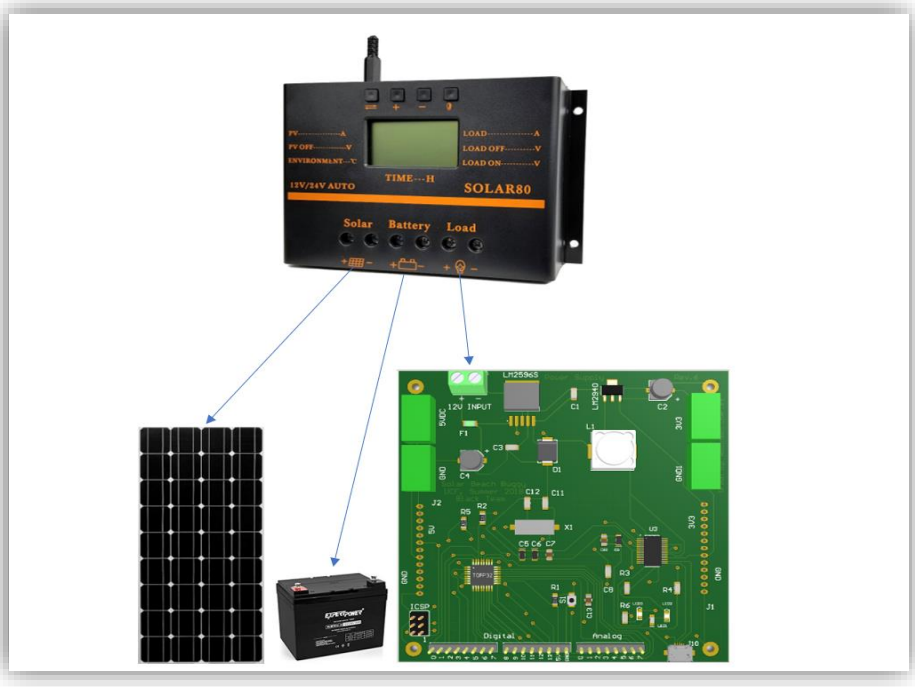


Figure 17 - PWM Solar Charge Controller Wiring Diagrams.

3.2.13.2 MPPT

MPPT controllers work by regulating the electrical module state in order to charge the battery bank with the maximum power output. Unlike PWM, MPPT controllers are very efficient. They can accept a higher input voltage than the battery can handle and step it down to match the battery bank voltage for a correct charge. The MPPT controller convert any excess voltage to current making the battery charges faster.

While MPPT provides the best solution for high power systems, it is very expensive. It is best to use it in a larger system.

In order to determine the best controller for this project, we will be looking at various charge controllers.

The very first one is TI charge controller known by TIDA00120.

The TIDA 00120 is a solar MPPT (Maximum Power Point Tracking) charge controller. It is a 20A MPPT solar charge controller that can take a solar charge inputs between 12-24V DC and outputs up to 20A current to either a 12 or 24V batteries. It is designed for small and medium solar systems. It is scalable to a 48V and up to 40A current by just replacing a few MOSFETs. The controller has lots of features built including reverse battery protection and software programmable alarms. It has an efficiency of over 97% at full load in a 24V systems and uses less than 10mA of standby currents while operating from battery.

The charge controller has a current sense and a cut-off circuit. The circuit will check the voltage in the battery; if it's at least 12.7V meaning it is fully charged, the circuit will prevent any more charge going to the battery. On the other hand, if it is less than fully charged, the circuit will send the appropriate charge to the battery until it reaches a fully charged state.

The operating efficiency of the design reaches well above 96% in a 12V and 24V systems, as illustrated in Figure 18.

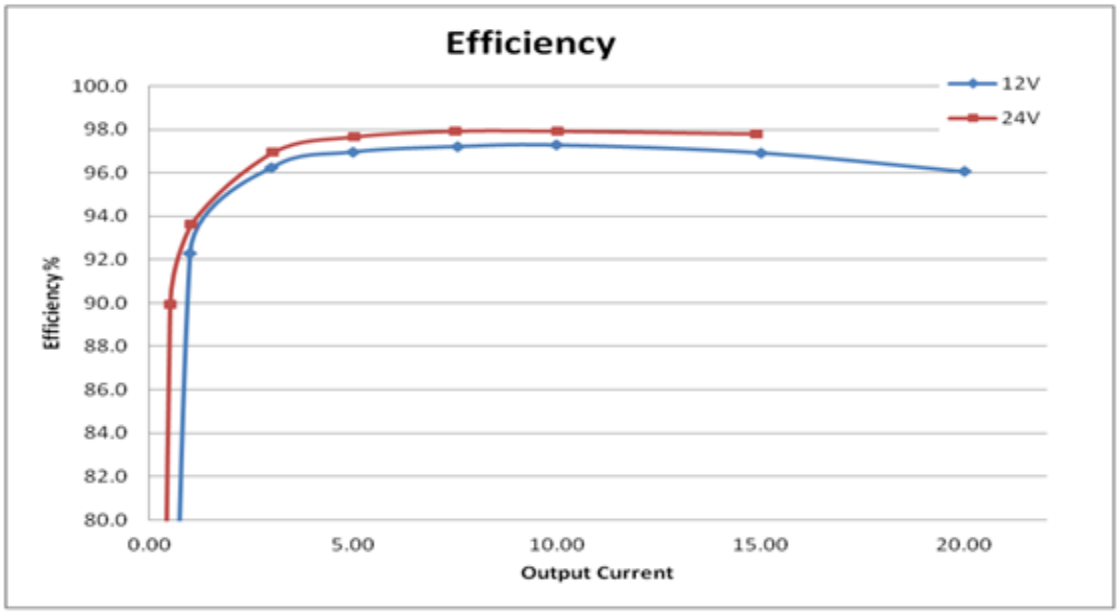


Figure 18 - Efficiency Curve Vs Output Current. Courtesy of Texas Instrument.

The second charge controller we will be looking at is the Rover, illustrated in Figure 23. The Rover is an MPPT charge controller; it has a 12-24V DC input and is available in 20A and 40A. it features a die-cast aluminum shell that allows heat to dissipate and provides protection for the device. It is an intelligent controller capable of tracking power efficiently up to 99%. It can also self-diagnose system faults.

Some of its key features:

- Automatically detects 12V or 24V DC system voltages
- Compatible with various Deep Cycle battery: Sealed, Gel, and Flooded
- Innovative MPPT technology with high tracking efficiency up to 99%
- Electronic protection against reverse polarity, overcharging, and overload
- LCD screen and multiple LED indicators for displaying error codes
- Features diverse load control
- Die-cast aluminum design allows for efficient heat dissipation

Figures 19 illustrates the Rover 20A.

Figure 19 - Rover 20A MPPT Solar Charge Controller.

In the following section, Table 16 compares the different solar charge controllers in consideration.

	Renogy Rover 20A	Renogy Rover 40A	Tristar TS-60	Tristar TS-60	ZHCSolar
Type of Regulation	MPPT	MPPT	PWM	MPPT	PWM
Load Current Rating	20A	40A	60A	60A	80A
System Voltage	12-24V	12-24V	12-24-48V	12-24-36-48V	12-24V
Output Voltage	12-24V	12-24V	*	8-72V	12-24V
Price	$129.99	$209.99	$220.00	$579.00	$73.00

Table 16 - Comparison table of solar charge controller.

The ZHCSolar is the likely candidate compared to the other choices. It is inexpensive and can provide four times as much current as the Rover 20A and twice as much current as the Rover 40A. Unlike the TIDA00120, the ZHCSolar is a fully assembled board with enclosure and is available to purchase. Although the TIDA00120 can be designed and assembled for less than $30, other readily available product on the market can be purchased for less and still provide more output current to the battery.

3.2.14 LIDAR

LIDAR stands for “light detection and ranging” and is often used for machine vision. It works by sending out laser pulses and recording the time it takes for the pulse to return. Sending out many quick pulses while moving the laser around allows the LIDAR to create a real-time 3D map of its environment. Figure 20 visualizes the parts of a typical LIDAR system.

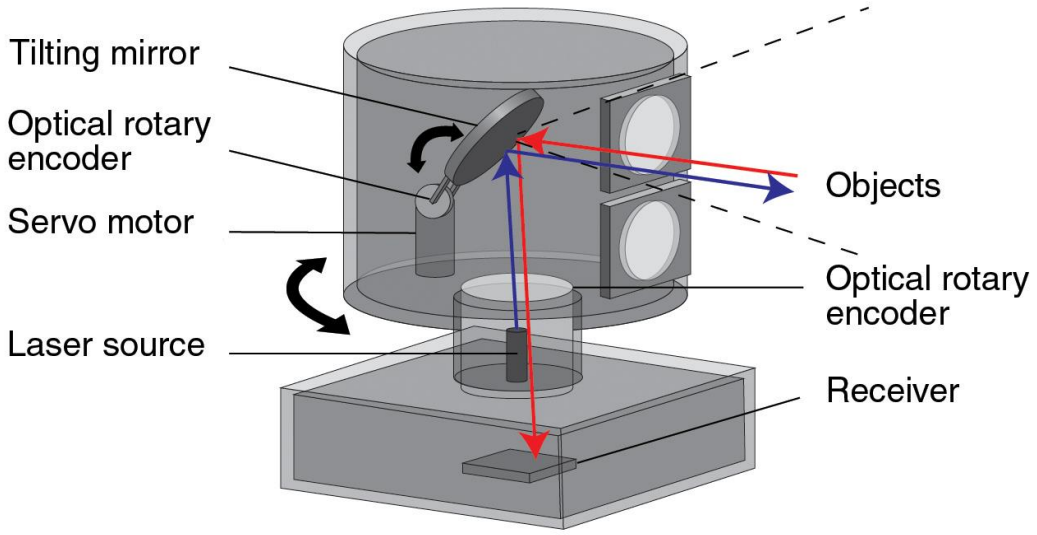


Figure 20 - Diagram of a typical LIDAR system. Courtesy of Renishaw LLC.

In principle, a lidar must consist of a transmitter and a receiver. Short light pulses with lengths of a few to several hundred nanoseconds and specific spectral properties are generated by the laser. Light pulses with lengths ranging from a few to several hundred nanoseconds and specific spectral properties are generated by the laser. Many systems also use a beam expander within the transmitter unit to reduce the divergence of the light beam. At the receiver end, optics are used to gather the returning signal. An optical analyzing system can be used which, depending on the application, selects specific wavelengths or polarization states out of the collected light.

Wavelengths used in lidar depend on the application and usually extend from about 250nm to 11μm. Ruby, nitrogen, copper-vapor, and CO2 lasers were mainly used in the early years, but high-power excimer and Nd:YAG lasers have been growing in use since the 1980s. Excimer lasers produce ultraviolet radiation, while the Nd:YAG crystal emits in the infrared spectral region at a wavelength of 1064nm. Frequency doubling and tripling with nonlinear crystals is often used to convert the wavelength of Nd: YAG radiation to 532 and 355nm. Quadrupling to 266nm is also utilized. The doping of crystalline lattices, e.g., yttrium aluminum garnet (YAG), yttrium lithium fluoride (YLF), lutetium aluminum garnet (LuAG), or of glasses with active ingredients such as Nd, Ho, Tm, Cr, Er, or Yb, creates a wide range of infrared wavelengths, some of which are particularly well suited for Doppler lidar. Presently, new laser types such as slab, microchip, waveguide, and solid-state Raman lasers are under investigation for their possible use in lidar.

Large systems have as many as 64 emitter/receiver pairs, also known as “channels”. Multiple channels enable the system to generate more than a million data points per second. However, 64 stationary channels aren’t enough to be able to map an entire environment. It would just give very clear resolution in focused areas. Adding more of these channels is expensive due to the precision required in the optics, so increasing the number of channels above 64 just increases cost faster. Instead, many Lidar systems use rotating assemblies, or rotating mirrors to enable the channels to sweep around the environment 360 degrees. Usually, each of the emitter and receiver pairs are angled above or below horizontal to blanket more of the environment in the field of view of the lasers.

Semiconductor lasers are favorable compared to other solid-state lasers in pulsed TOF (time-of-flight) laser radars because they are small, mechanically stable, relatively cheap, and have good efficiency. One of the most important requirements for a laser being used for LIDAR is that it must make short, powerful optical pulses. Laser Diodes are capable of the peak powers and pulse durations needed.

A quickly-switching circuit is needed to pulse the laser diode. A schematic diagram of a basic avalanche transistor laser pulser is shown in Figure 21.

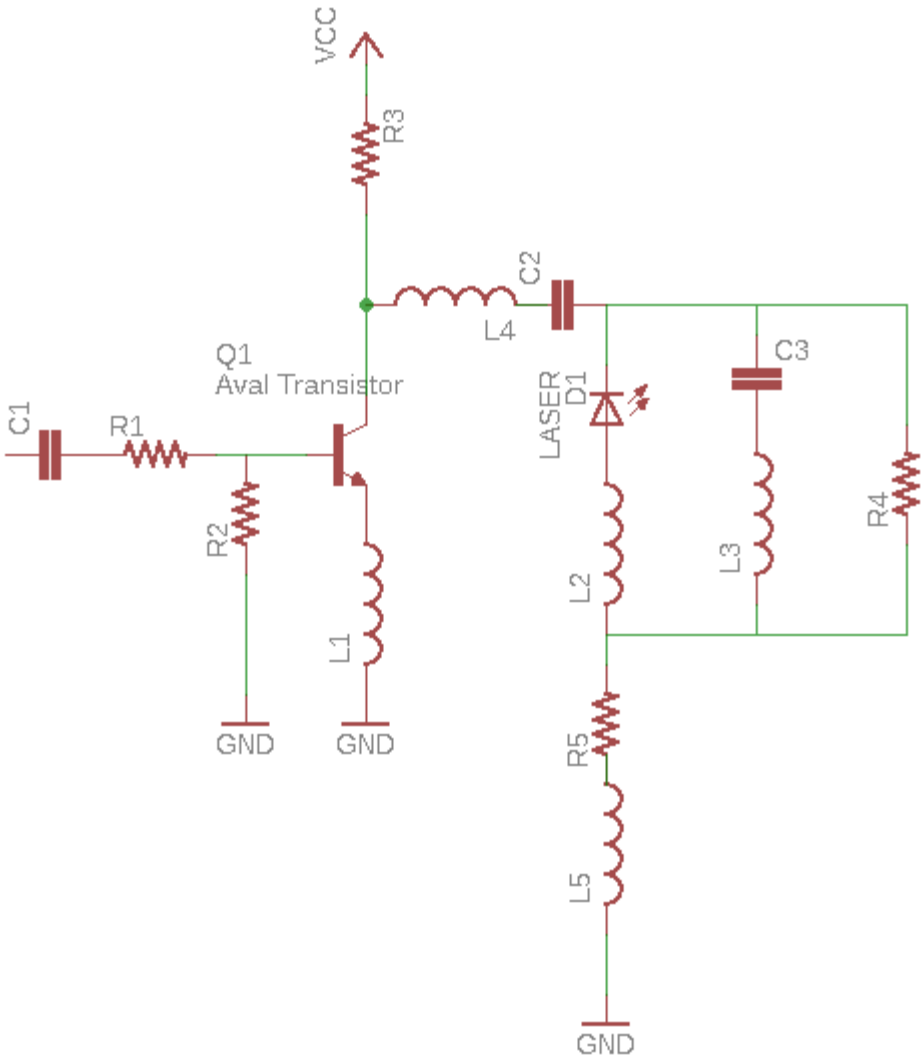


Figure 21 - Schematic diagram of a typical avalanche pulser.
Redesigned in Eagle for clarity.

A study performed by Ari Kilpelä and Juha Kostamovaara at the University of Oulu compares the parameters of different avalanche transistor circuits used for time of flight laser radars. Figure 22 below shows a comparison for the max peak current pulses that different avalanche transistors are capable of producing, while Figure 23 shows the output of laser diodes LD-65 and CVD-197 with full bandwidth of the system and from LD-65 with 100 MHz bandwidth.

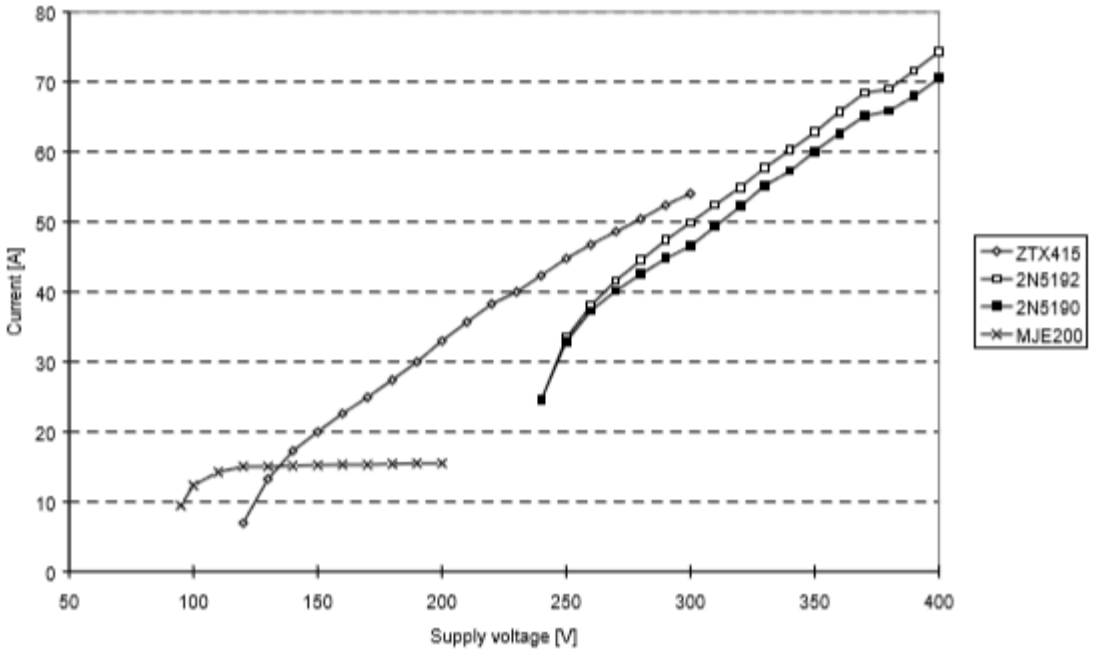


Figure 22 - The peak amplitudes of current pulses as a function of supply voltage.

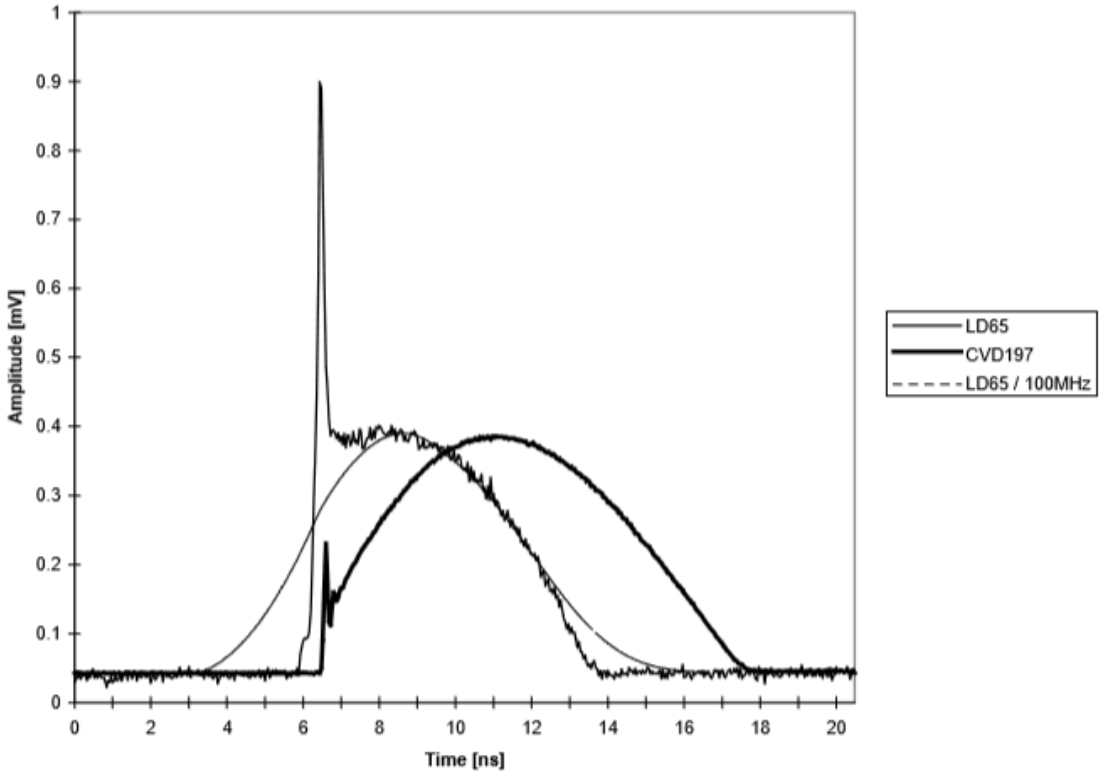


Figure 23 - Output of Laser Diodes at full and 100 MHz Bandwidth.

Table 17 shows measurement results of current and optical pulses from a circuit, in which a capacitor has been used in parallel with laser CVD197.

	Current of series resistor			Optical pulse	
Capacitance	I_{peak} (A)	t_{rise} (ns)	t_{width} (ns)	t_{width} (ns)	P_{peak} (W)
0 pF	29.6	3.8	9.2	8.9	57
100 pF	30.9	5.5	9.0	8.7	58
200 pF	26.5	2.7	10.5	7.2	72
300 pF	26.7	2.4	12.4	7.2	72
400 pF	28.3	2.5	13.8	7.5	70

Table 17 - Measurements of Current, Power and Width in nanoseconds of optical pulses from a circuit.

The data above shows that laser diodes with avalanche transistor circuits are capable of producing laser pulses with adequately short pulse durations and high enough peak intensities that they can be used for range finding.

3.2.15 Power Calculation

In order to meet the requirements of the design, it is important to calculate how much power the various components of the beach buggy will consume. Based on this calculation, a proper battery configuration can be selected to meet the project requirements. A proper solar cell selection can also be made, ensuring that the system is capable enough to be charged adequately to sustain the buggy for the duration required for its trek. Table 18 shows the power consumption of all electrical components used in the system as specified by each component's hardware datasheet.

Components	Supply Voltage (V)	Number of Components	Net Current (A)	Power Consumption (W)
Motors	12	2	26.7	640.8
Servo	5	1	0.15	0.75
Raspberry Pi3	5	1	2.5	12.5
GPS Antenna	3.3	1	negligible	~
LIDAR	5	1	0.275	1.375
GPS	5	1	negligible	~
TK1	12	1	5	60
ODROID	5	1	4	20
Total power consumption				735.4

Table 18 - Power consumption by parts.

3.3 Software Components

For hardware to work, it must have software. While the CPU provides the brain of the buggy and its other parts as its limbs, eyes, and hands, the software is its mind. The following software considerations were researched to provide the command and control of the buggy to make it the autonomous robot that it needs to be.

3.3.1 Programming Language

Two programming languages were considered: C and Python.

C is an imperative procedural language that provides low-level access to memory, provides language constructs that map efficiently to machine instructions, and requires minimal run-time support. By design, C provides constructs that map efficiently to typical machine instructions, and therefore it has found lasting use in applications that had formerly been coded in assembly language, including operating systems, as well as various application software for computers ranging from supercomputers to embedded systems.

Python is an interpreted high-level programming language for general-purpose programming. Python features a dynamic type system and automatic memory

management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

Due to the nature of our system, and the middleware required for robot control (section 3.3.4), Python is considered to be the most optimal choice in our programming design language. In addition, plenty of libraries have been written due to the popularity of Python, allowing some of the work to be offloaded from us due to simply including the libraries written.

3.3.2 Robot Control

Robot Operating System (ROS) is robotics middleware (i.e. collection of software frameworks for robot software development). It is not an operating system but it provides services designed for heterogeneous computer cluster such as hardware abstraction, low-level device control, implementation of commonly used functionality, message-passing between processes, and package management. Running sets of ROS-based processes are represented in a graph architecture where processing takes place in nodes that may receive, post and multiplex sensor, control, state, planning, actuator and other messages. Despite the importance of reactivity and low latency in robot control, ROS, itself, is not a real-time OS (RTOS), though it is possible to integrate ROS with real-time code.

ROS has extensive developer and community support, allowing the use of many libraries and packages that can be used to process the various data output from our sensors such as from our main sensor, LIDAR.

3.3.3 Operating System

The Robot Operating System (ROS), being middleware, requires an underlying Operating System to run. Its only supported Operating System is Ubuntu, an open source operating system.

Ubuntu is a Linux distribution based on the Debian architecture. It is usually run on personal computers, and is also popular on network servers, usually running the Ubuntu Server variant, with enterprise-class features. Ubuntu runs on the most popular architectures, including Intel, AMD, and ARM-based machines.

The specific distribution of Ubuntu used in our project is Linux Mint Cinammon 18.3. It is a Unix-like operating system running Ubuntu 16.04 (Xenial) at its core. One of the main advantages of Linux Mint is its similarity to Windows, allowing ease of use of the system and providing a very user-friendly environment.

Being a Unix-like operating system, it has access to most software available written specifically for Ubuntu and installing additional required software can be done easily with the use of Ubuntu's *apt-get* system.

4.0 Related Standards

Standards are very important in today's technological world as not only do they provide a blueprint for designers and manufacturers to follow, but it also allows the ease of use for consumers by allowing interoperability and interdependency. Standards also provide major safety guidelines that prevent any harmful effects from the use of products or from the implementation of technology.

Many organizations provide standards. Examples of such organizations are the American National Standards Institute (ANSI), a private non-profit organization whose mission is "to enhance U.S. global competitiveness and the American quality of life by promoting, facilitating, and safeguarding the integrity of the voluntary standardization and conformity assessment system"; the IPC (Association Connecting Electronics Industries), a trade association whose aim is to standardize the assembly and production requirements of electronic equipment and assemblies; and the IEEE (Institute of Electrical and Electronics Engineers), "The world's largest technical professional organization for the advancement of technology".

Standards from these organizations will be looked at and discussed in relation to this project.

4.1 PCB Standard

The IPC-2221B Generic Standard on Printed Board Design (PCB) establishes the generic requirements in designing organic printed boards and other forms of component mounting or interconnecting structures. The requirements in the standard establishes design principles and recommendations that shall be used together with the detailed requirements of another specific sectional standard.

The standard identified generic physical design principles and is supplemented by other various sectional standards that focuses on specific aspects of printed board technology that includes:

IPC-2222 Rigid organic printed board design
IPC-2223 Flexible printed board design
IPC-2225 Organic, MCM-L, printed board design
IPC-2226 High Density Interconnect (HDI) printed board design

The standard itself is not a performance specification for finished printed boards.

The standard allows us to create PCBs that are oriented properly, allowing an efficient and error-free soldering process; placed properly, preventing things such

as components on the solder side of a board resting behind plated through-hole components; organized, allowing the PCB to be efficient, look clean, and minimize assembly steps due to standards such as requiring that Surface Mount (SMT) components be placed on the same side of board, and all through-hole (TH) components on the top side the board; and minimized interference due to guidance such as separating the power ground and control ground for each power supply stage.

4.2 Software Testing Standard

The ISO/IEC/IEEE 29119 series consists of five standards that define an internationally agreed set of standards to support software testing. The standard allows developers to create software that meets industry specifications, as well as proper testing of functionalities to ensure proper operation of the system.

Two parts of this standard are discussed in relation to this project: ISO/IEC/IEEE 29119-2 (Part 2) - Test Processes and ISO/IEC/IEEE 29119-4 (Part 4) - Test Techniques.

4.2.1 ISO/IEC/IEEE 29119-2 (Part 2) - Test Processes

This part defines a generic process model for software testing that is intended for use by organizations when performing software testing. It comprises test process descriptions that define the software testing processes at the organizational level, test management level and dynamic test levels. Each testing process is described using a standard template.

The purpose of the organization level is to develop and maintain organizational test specifications, such as a test policy and organizational test strategy. “Organization” in our context, is our group members. This process allows us to implement a test specification based on stakeholder requirements.

The management level describes test planning, test monitoring and control, and test completion. Test planning involves understanding the context of the test and its scope, wherein one should then organize test plan development. Risks should then be identified and incorporated in to the design test strategy. Once a strategy is in place, staffing and scheduling should be determined, along with documenting the test plan. Once a test plan is documented and approved, the test plan is ready for test monitoring and control.

Once testing begins, test monitoring and control allows revisions or updates on previous test plans to allow for accounting of factors that may not have been considered previously. This allows the creation of new test plans that can be cycled again through test monitoring and control; repeating the process if necessary. As

with previous test plans, new test plans must confirm with organization test process guidelines. Once testing is complete, the test completion process can be executed.

The test completion process involves the dynamic test process which, similar to the test management process, involves test design and implementation. Additionally, the dynamic test process includes a processing for testing environment requirements, setting up the test environment, and then test execution. Once test execution is done, the results can be reviewed and, if any incidents arise, be reported so the rest of the group can be aware. Figure 24 illustrates the Test Management Process, as illustrated by ISO/IEC/IEEE 29119-2

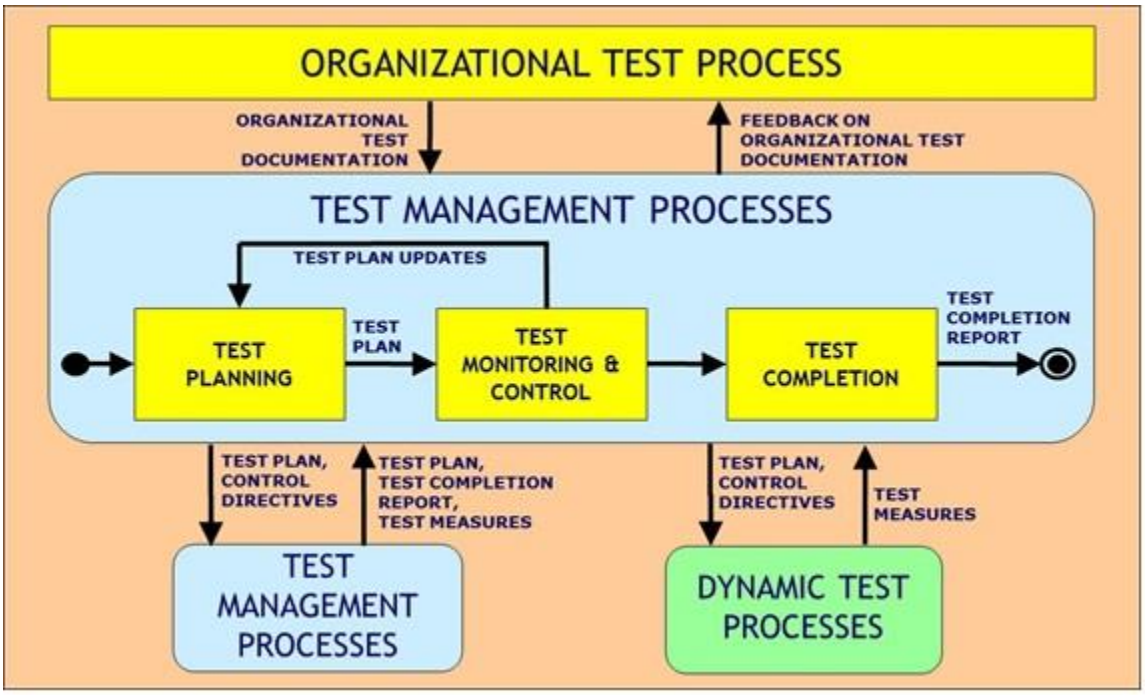


Figure 24. The Test Management Process.

4.2.1 ISO/IEC/IEEE 29119-4 (Part 4) - Test Techniques

This part defines standard definitions of software test design techniques (also known as test case design techniques or test methods) that can be used during the test design and implementation process that is defined in ISO/IEC/IEEE 29119-2. Techniques of part 4 are intended to support part 2 or can be used without part 2.

The techniques in the section are categorized into three categories: Specification-Based Test Design Techniques, Structure-Based Test Design Techniques, and Experience-Based Test Design Techniques. Each category has different techniques wherein the identification of relevant test aspects and the derivation of test inputs and test cases varies by technique used. Consideration of these

technique categories and the techniques within can allow our group to thoroughly, properly, and effectively test the requirements and operation of our buggy.

Specification-Based Test Design Techniques base on the (functional) specification of the system under test. They are also called black-box testing. These techniques examine the functionality of an application without knowing its internal structures or workings. The tester is aware of what the software is supposed to do but is not aware of how it does it.

Structure-Based Test Design Techniques base on the (internal) structure of the system under test. They are also called white-box testing. These techniques tests internal structures or workings of an application, as opposed to its functionality. These techniques are especially useful in knowing the flow of program code within the software so that any discrepancies between blocks or functions can be corrected.

Experience-Based Test Design Techniques rely on the experience of the human tester. The standard list only one technique in this category--Error guessing. It is a test method in which the test cases used are established based on experience in prior testing. Typical errors include divide by zero, null pointers, or invalid parameters. Error guessing has no explicit rules, and its scope can vary widely depending on the situation.

5.0 Project Hardware and Software Design Details

After an extensive discussion on the base elements which will be necessary to the successful operation of the buggy, this section will detail the current plans the team has with regards to the actual design of the electrical components which will be used on the beach buggy. This is obviously a vital part to ensuring the success of the overall design, and so it is important to be as detailed as possible.

5.1 LIDAR

This section will discuss the implementation of the Lidar system onto our buggy. This will incorporate a laser diode and photodiode, as well as driving and receiving circuits that will operate the diodes. The output of this system will be fed into an Arduino and an analysis will be carried out on that data in order to determine distances and create a rudimentary point cloud which will allow us to identify objects in the path of the buggy.

The Lidar module will consist, broadly, of two components: A laser emitter which will periodically emit a pulse of light out into the environment we wish to scan, and a receiver module, which will focus a portion of the laser light reflected directly

back from the environment. By using a periodic step function as a method of time-keeping, the difference in time between the output of a pulse of the laser diode and the registered input of the photodiode can be recorded and converted into a distance traveled, which can in turn be fed into an Arduino for computational purposes. This method of laser rangefinding is referred to as time-of-flight tracking, and for our purposes, we have chosen it as the most efficient and easily developed method of laser rangefinding. A visual depiction of the basic operation of the Lidar system is shown in figure 25.

As stated above, the Lidar module will consist of two major components: The laser emitter and the photodiode receiver. The laser emitter is a fairly simple set up, being driven by a laser driving circuit that produces a regularly pulsed signal, which is kept in sync by the sequential equivalent time sampling circuit. This pulse is sent to the laser diode and produces an output which is then passed through a collimating lens before being sent out into the environment. The photodiode receiver is also a relatively simple circuit design, as it only needs to incorporate a detector that is sensitive enough to receive the minute amounts of light reflected off of the surfaces in the environment, and an amplifier to turn the received signals into a useful pulse waveform. A rough block diagram for our system is displayed in figure 26.

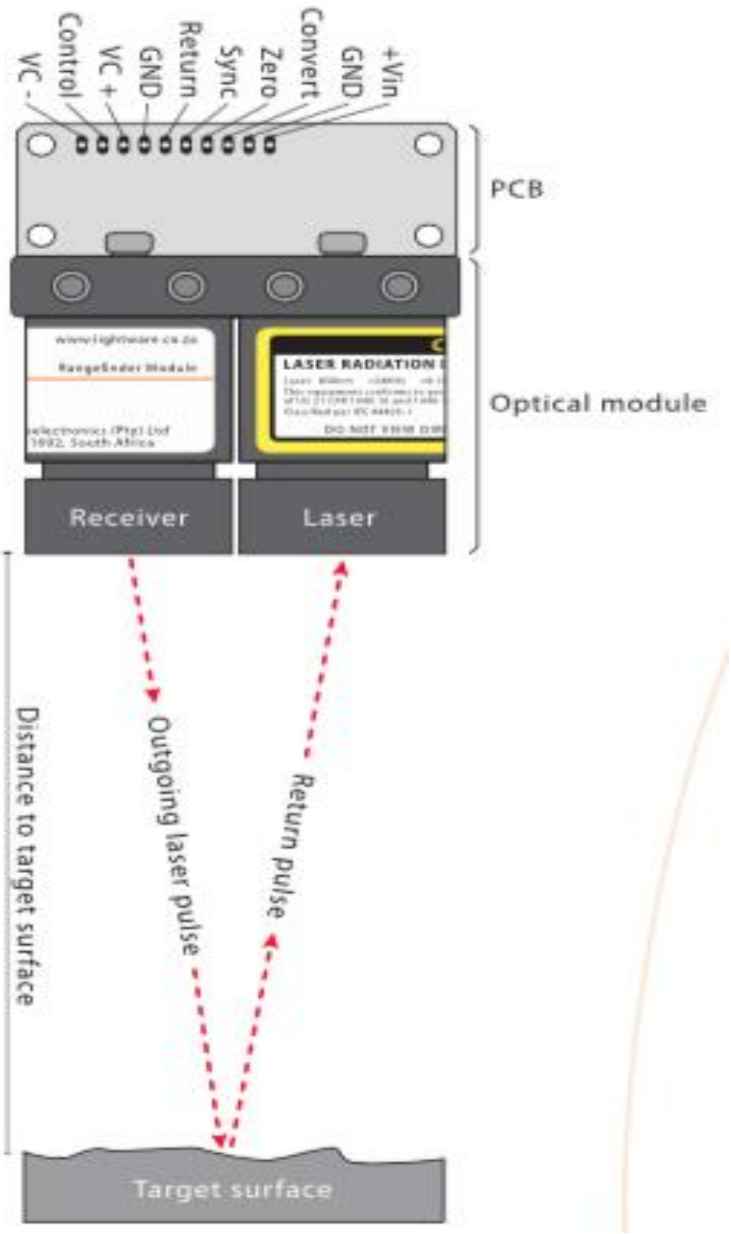


Figure 25- A basic diagram of the operation of the Lidar system
Courtesy of Lightware Optoelectronics.

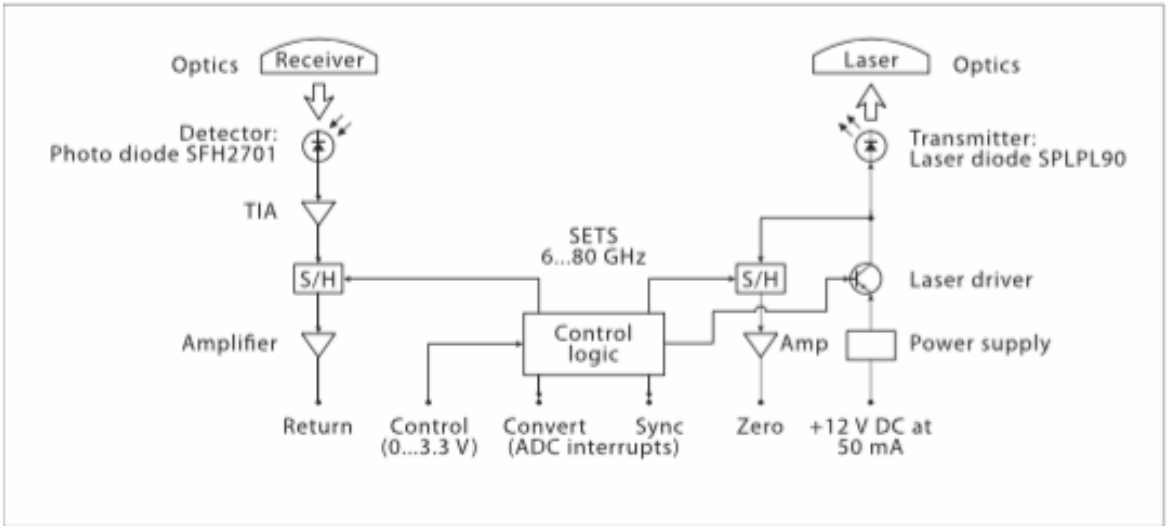


Figure 26 - A basic block diagram of the Lidar system
Courtesy of Lightware Optoelectronics.

One of the most important aspects of our Lidar set up is the timing circuits, referred to as the sequential equivalent time sampling circuit. Given the high frequency at which light pulses need to be generated, received, and calculated, an efficient and dependable timing circuit is vital to the successful operation of our device. As we are dealing with a system that would require clocking speeds on the order of 15 GHz in order to maintain a resolution of 1cm, it would be impractical to incorporate a direct timer to track our pulses. In order to design around the expensive equipment that a direct sampling method would require, the sequential equivalent time sampling circuit operates by establishing a discreet timebase of our own choosing, which can then be used to sample the outgoing and incoming pulse waveforms and convert high speed signals onto the slower timebase that we have established, thereby allowing the computations to be done with less expensive, power intensive hardware. The slowed down signals operate on a timebase that is around 100,000 times slower than the real-time signals, and the amount of timebase expansion can be modified to change the resolution of the measurements and the update rate.

The real-time span of the timer in our timing circuit is approximately 122 ns but can be stretched to more than 20ms using the timebase adjustment we had previously mentioned. This gives us an operational distance of approximately 18.33 m, which is more than sufficient for our purposes. A square sync signal is provided, the falling edge of which marks the end of one measurement and the start of another on our expanded timebase. The period of the sync signal is always equivalent to 18.33 m, meaning that distances between signals on our expanded timebase can be calculated as proportions of the period of the sync signal. The relationship between the sync signal and our output/input signals is illustrated in figure 27. This period can be altered by a change in the control voltage input, which alters the timing and results in a faster or slower expanded timebase. In addition, the two analog signals that represent the outgoing laser diode pulse and the return signal

are on the same timebase as the sync signal, due to the fact that the timing circuit performs a timebase expansion of each of the signals, with the sync being an expanded image of the real-time timer. In addition, there is a convert reference that can be used to trigger successive ADC conversions on a host controller. This convert signal is synchronous with our timing circuit and will help reduce noise in the digitized signals.

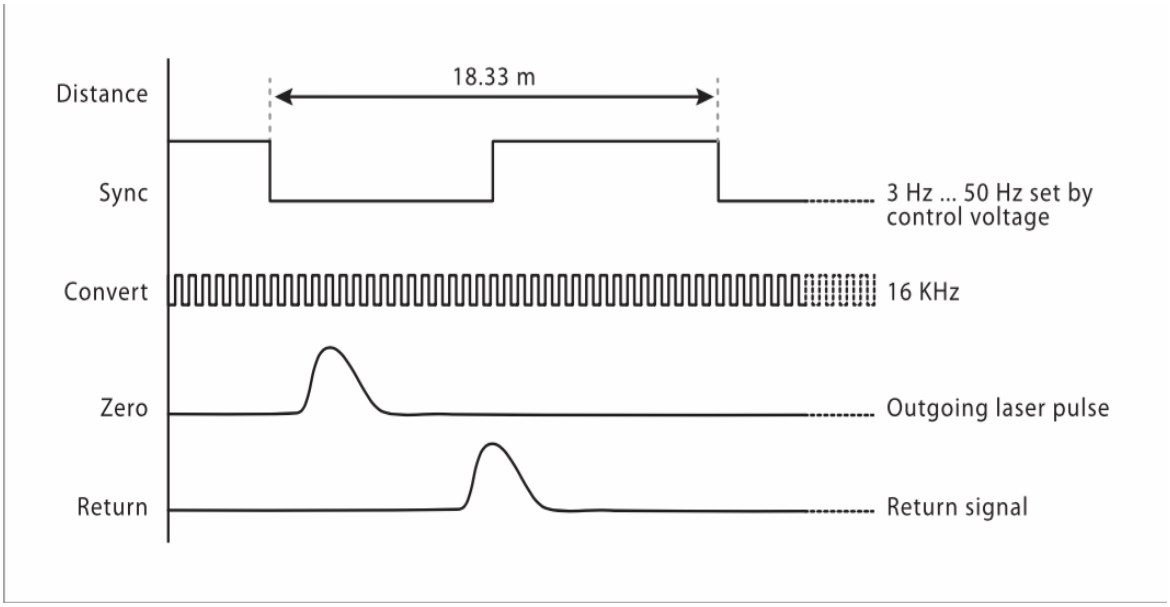


Figure 27 - An illustration of the timing relationship between signals
Courtesy of Lightware Optoelectronics.

On an expanded timebase, the signal to fire the laser diode is sent at the exact same moment as the falling edge of the sync signal. However, because there is an inherent delay between the signal being received and the diode producing light, there is a noticeable delay on the expanded timebase. There is also the fact that the pulses have a width associated with their real-time generation and travel. This means that we cannot utilize the full range of the sync signal for rangefinding, and our effective location distance will be slightly below 18.33 meters. Despite this minor limitation, it should not prove to be a significant detriment to our design, as the distances we are concerned with fall well within our operating potential.

SETS (sequential equivalent time sampling):

The time between the outgoing laser pulse and the return signal needs to be measured with very high precision in order to calculate the distance. The SETS circuit slows down these signals so that they can be viewed on a much lower frequency.

The basic principle of SETS sampling is illustrated in figure 28. In SETS, the test signal should be periodic or repetitively generated. SETS system captures one

sample at each acquisition. When the trigger signal is detected, the first samples is taken after a fixed time delay. At the next acquisition, after trigger signal, the second sample is taken after a slightly longer time delay. After enough samples are collected, the signal is reconstructed by time aligning the samples

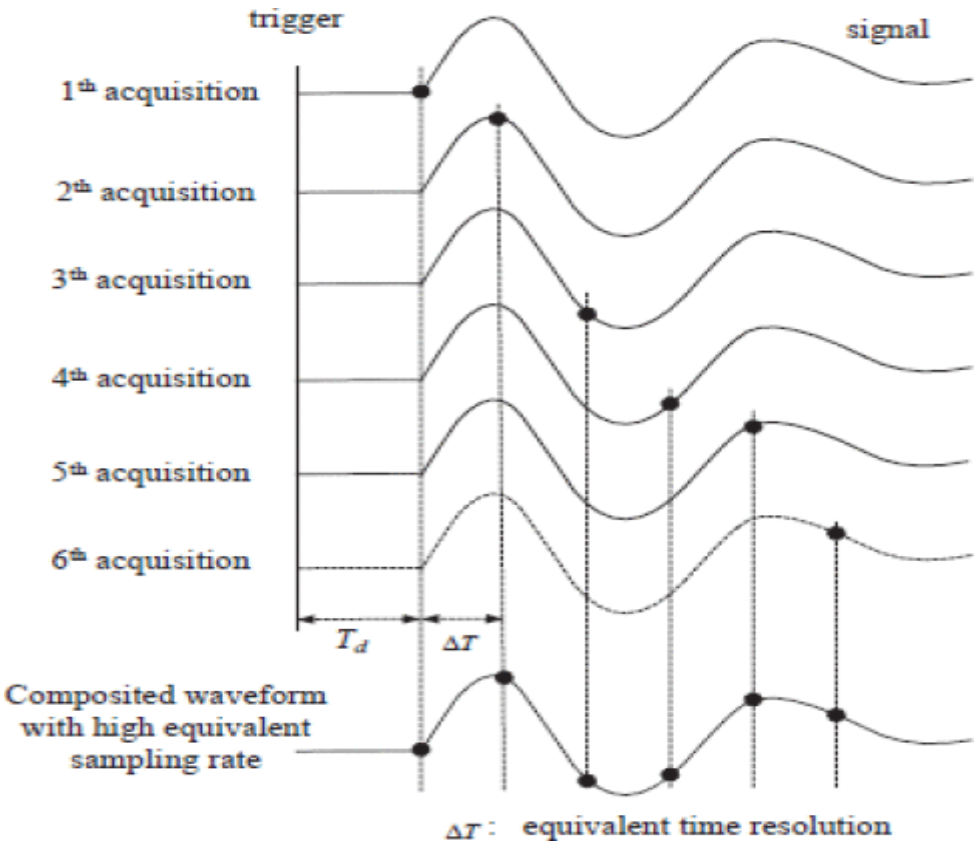


Figure 28. Basic principle of SETS sampling.

Software Timing Strategies:

After analogue to digital conversion (ADC), the digital signal can be analyzed using different software algorithms. The simplest strategy is to define a threshold voltage on and then to count the number of ADC samples from the falling edge of the sync reference to the rising edges of the zero and return that reach this threshold. Each ADC sample is counted, and these counts give the relative time to the edges measured on the expanded timebase. The number of ticks between falling edges of the sync reference allow the distance can be calculated as follows:

distance_to_target = ((ticks_to_return - ticks_to_zero) / ticks_between_Sync_edges) * 18.33m

One limitation to this strategy is that the return signal will change size and shape depending on the color and distance of the measured object. These changes will

cause differences in both the height and the width of the digitized signal and therefore the point at which the leading edge crosses the threshold. This effect can be minimized by setting a dynamic threshold that is set at a fixed proportion of the height of the return signal.

Another strategy is to use constant fraction discrimination (CFD). This method uses both the rising edge and the falling edge of the return signal, as they are timed as they cross a fixed threshold. The true position of the return is then defined as midway between these points.

The current pulse first goes through a transimpedance amplifier, labeled as IC13 in Figure below, which turns the input current signal into an input voltage. This voltage then goes through the sequential-equivalent-timing circuits to slow down the voltage before it is amplified and then outputted. The receiver circuit is illustrated in figure 29 below.

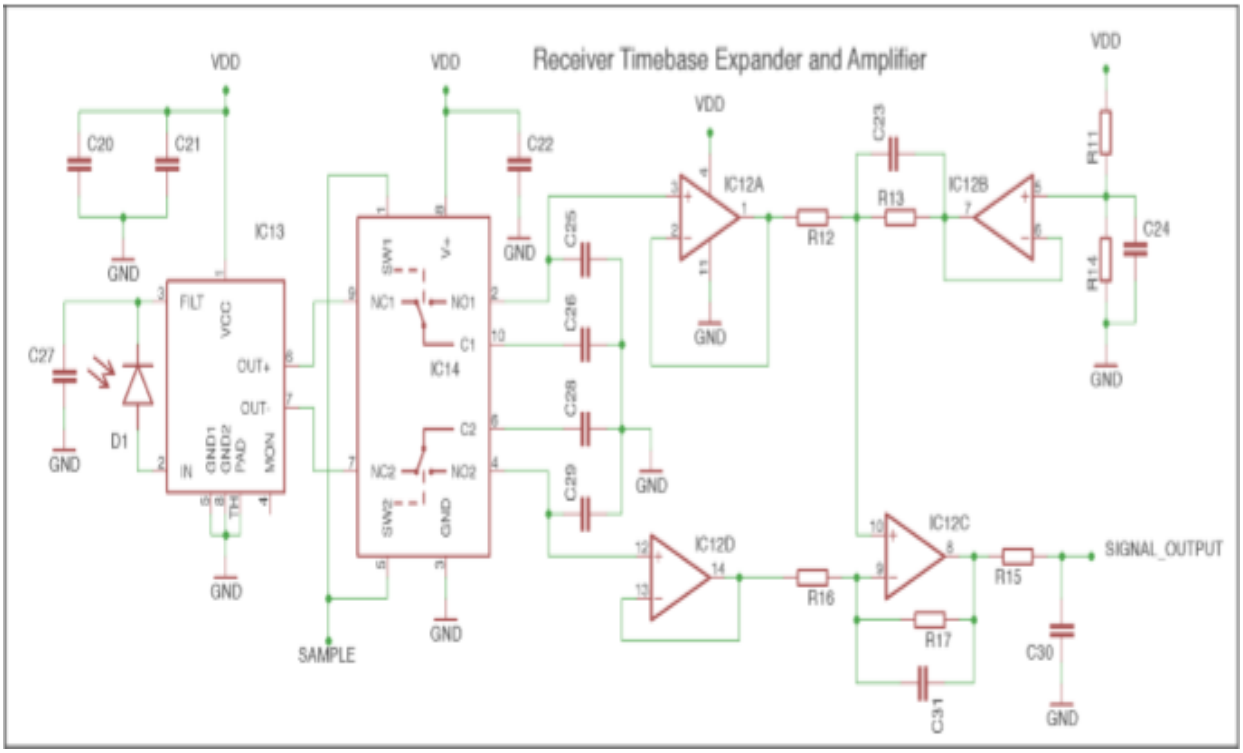


Figure 29. Receiver Circuit.

Once the signal travels through a laser driver, it must be also be sent to a sequential-equivalent-timing circuit and through an amplifier so that it can be outputted for timing purposes. Figure 30, shown below, shows the circuit schematic of the sequential-equivalent-timing circuit and the buffer used to slow down the signal.

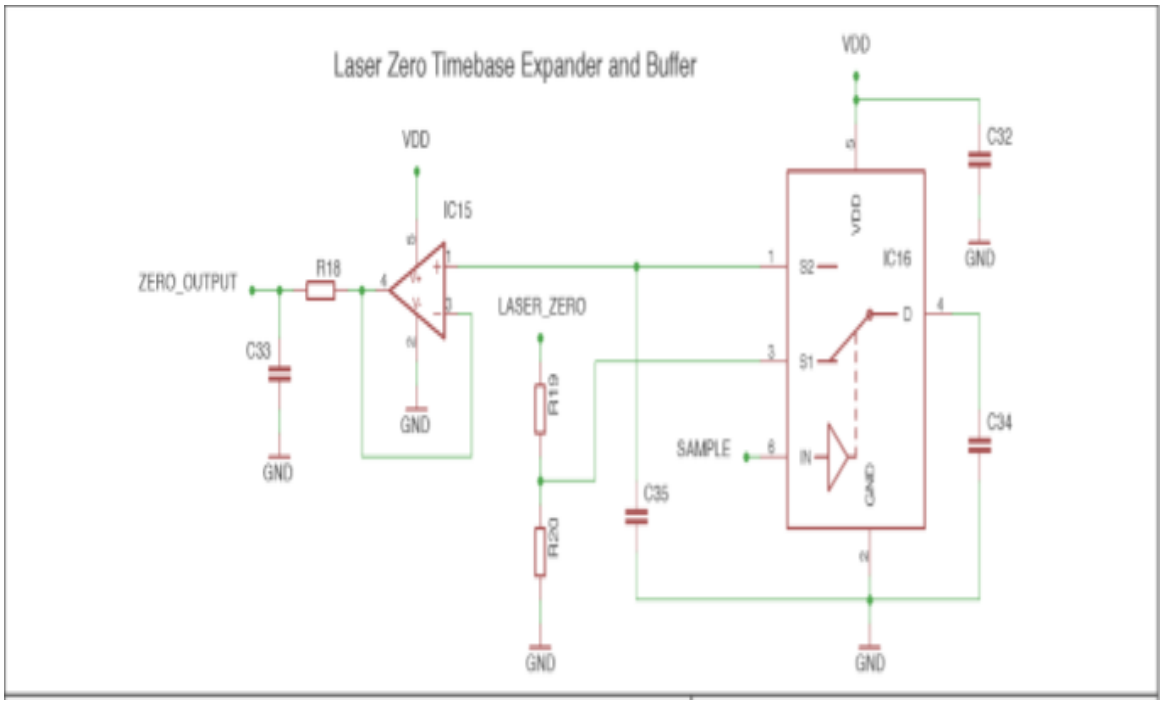


Figure 30. Circuit Schematic of the Sequential-Equivalent-Timing Circuit.

In order for the transmitting and receiving sides of the optics to produce slowed-down signals that can be outputted for timing purposes, it is controlled by circuitry designed to control the sequential-equivalent-timing-circuits on both the transmitting and receiving sides. It is also responsible for controlling the laser driver on the transmitting end. Figure 32 shows the schematic of this control logic, while figure 31 shows the laser driver.

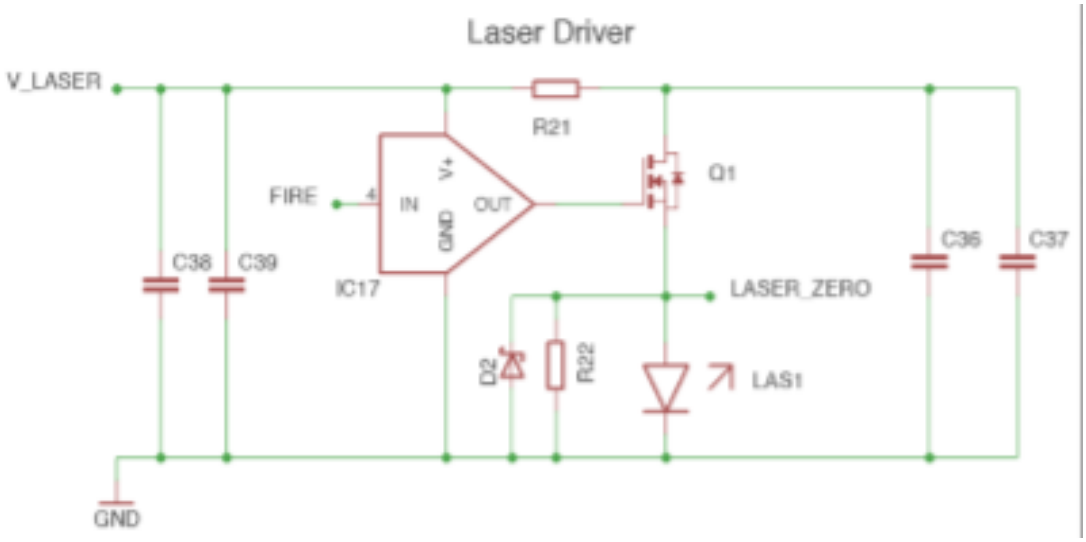


Figure 31. Laser driver.

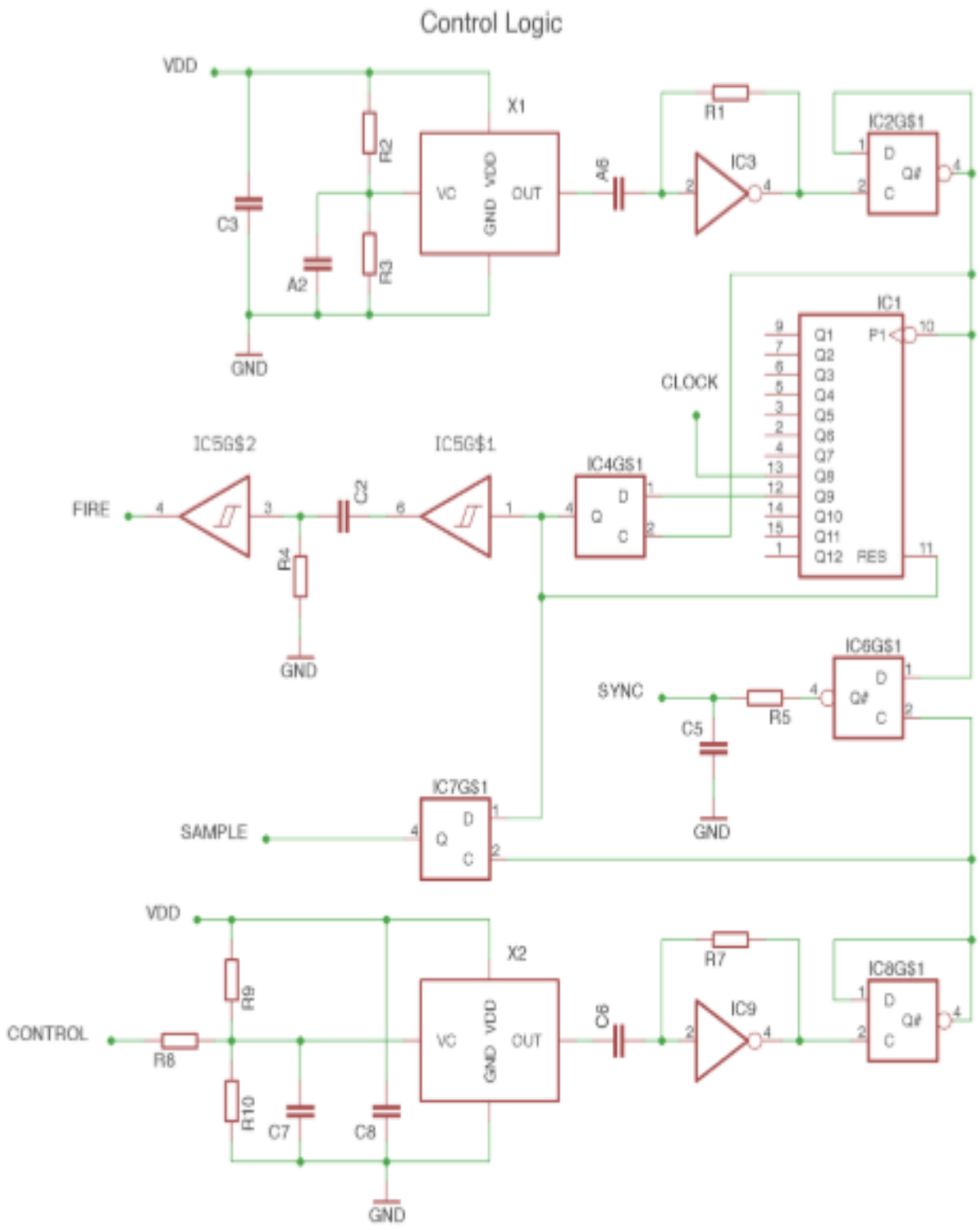


Figure 32. Control Logic Schematic.

Parts list: Table 19 shows all the components that will be on the PCB

		Reference	Description	Package	Supplier	Part code
2	Capacitors	C1	1nF, 50V, X7R	402	RS Components	723-5266
3		C2	120pF, 50V, X7R	402	RS Components	723-5376
4		C3	100nF, 16V, X7R	402	RS Components	723-5228
5		C4	100nF, 16V, X7R	402	RS Components	723-5228
6		C5	1nF, 50V, X7R	402	RS Components	723-5266
7		C6	1nF, 50V, X7R	402	RS Components	723-5266
8		C7	1uF, 10V, X5R	402	RS Components	723-5199
9		C8	100nF, 16V, X7R	402	RS Components	723-5228
10		C9	100nF, 16V, X7R	402	RS Components	723-5228
11		C10	100nF, 16V, X7R	402	RS Components	723-5228
12		C11	100nF, 16V, X7R	402	RS Components	723-5228
13		C12	100nF, 16V, X7R	402	RS Components	723-5228
14		C13	100nF, 16V, X7R	402	RS Components	723-5228
15		C14	100nF, 16V, X7R	402	RS Components	723-5228
16		C15	100nF, 16V, X7R	402	RS Components	723-5228
17		C16	100nF, 16V, X7R	402	RS Components	723-5228
18		C17	100nF, 16V, X7R	402	RS Components	723-5228
19		C18	10uF, 16V, X7R	1206	RS Components	723-6565
20		C19	10uF, 16V, X7R	1206	RS Components	723-6565
21		C20	1uF, 10V, X5R	603	RS Components	391-040
22		C21	100nF, 16V, X7R	402	RS Components	723-5228
23		C22	100nF, 16V, X7R	402	RS Components	723-5228
24		C23	100pF, 50V, X7R	402	RS Components	624-2929
25		C24	1uF, 10V, X5R	402	RS Components	723-5199
26		C25	1nF, 50V, X7R	402	RS Components	723-5266
27		C26	22pF, 50V, COG	402	RS Components	723-5408
28		C27	1nF, 50V, X7R	402	RS Components	723-5266
29		C28	22pF, 50V, COG	402	RS Components	723-5408
30		C29	1nF, 50V, X7R	402	RS Components	723-5266
31		C30	100pF, 50V, X7R	402	RS Components	624-2929
32		C31	100pF, 50V, X7R	402	RS Components	624-2929
33		C32	100nF, 16V, X7R	402	RS Components	723-5228
34		C33	1nF, 50V, X7R	402	RS Components	723-5266
35		C34	22pF, 50V, COG	402	RS Components	723-5408
36		C35	1nF, 50V, X7R	402	RS Components	723-5266
37		C36	2 x 22nF, 50V, COG	603	RS Components	391-195
38		C37	2 x 22nF, 50V, COG	603	RS Components	391-195
39		C38	10uF, 16V, X7R	1206	RS Components	723-6565
40		C39	0.1uF, 25V	603	RS Components	147-538
42	Resistors	R1	100k, 1%, 100ppm	402	RS Components	667-8977
43		R2	10k, 1%, 100ppm	402	RS Components	678-4697
44		R3	10k, 1%, 100ppm	402	RS Components	678-4697
45		R4	200R, 1%, 100ppm	402	RS Components	667-8628
46		R5	200R, 1%, 100ppm	402	RS Components	667-8628
47		R6	200R, 1%, 100ppm	402	RS Components	667-8628
48		R7	100k, 1%, 100ppm	402	RS Components	667-8977
49		R8	10k, 1%, 100ppm	402	RS Components	678-4697
50		R9	10k, 1%, 100ppm	402	RS Components	678-4697
51		R10	4k7, 1%, 100ppm	402	RS Components	667-8794
52		R11	10k, 1%, 100ppm	402	RS Components	678-4697
53		R12	1k, 1%, 100ppm	402	RS Components	667-8680
54		R13	10k, 1%, 100ppm	402	RS Components	678-4697
55		R14	1k, 1%, 100ppm	402	RS Components	667-8680
56		R15	10k, 1%, 100ppm	402	RS Components	678-4697
57		R16	1k, 1%, 100ppm	402	RS Components	667-8680
58		R17	10k, 1%, 100ppm	402	RS Components	678-4697
59		R18	10k, 1%, 100ppm	402	RS Components	678-4697
60		R19	470R, 1%, 100ppm	402	RS Components	678-9355
61		R20	200R, 1%, 100ppm	402	RS Components	667-8628
62		R21	100R, 1%	1210	RS Components	679-2373
63		R22	27R, 1%	603	RS Components	679-0099

65	Integrated Circuits	IG1	SN74HC4040PWG4, 12 stage counter	TSSOP-16	RS Components	663-2105
66		IG2	SN74LVC1G80DCK, D-flip-flop, Q#	SC-70	RS Components	662-8803
67		IG3	74LVC1GU04DCK, inverter	SC-70	RS Components	662-6670
68		IG4	SN74LVC1G79DCK, D-flip-flop, Q	SC-70	RS Components	662-8806
69		IG5	SN74LVC2G17IDCK, dual, ST, buffer	SC-70-6	RS Components	662-8948
70		IG6	SN74LVC1G80DCK, D-flip-flop, Q#	SC-70	RS Components	662-8803
71		IG7	SN74LVC1G79DCK, D-flip-flop, Q	SC-70	RS Components	662-8806
72		IG8	SN74LVC1G80DCK, D-flip-flop, Q#	SC-70	RS Components	662-8803
73		IG9	74LVC1GU04DCK, inverter	SC-70	RS Components	662-6670
74		IG10	L78L08, 8V, linear regulator	SC-89	RS Components	686-9476
75		IG11	L78L33, 3.3V, linear regulator	SC-89	RS Components	686-9426
76		IG12	MCP6L04, quad, op-amp	TSSOP-14	RS Components	768-1404
77		IG13	MAX3658, transimedance amplifier	TDFN8	Digikey	MAXIM
78		IG14	TS5A23157DGSRG4	MSOP-10	RS Components	662-2788
79		IG15	LMV321IDCKRG4	SC-70	RS Components	660-9603
80		IG16	TS5A3157DCKRG4	SC-70-6	RS Components	662-2798
81		IG17	MIC44F18, MOSFET driver	MSOP EP-8	RS Components	453-205
83	misc	CON1	10x1 male header	0.1”		
84		D1	SFH2701, photodiode	3216	RS Components	665-5338
85		D2	1N5819, diode	SOD123	RS Components. In parallel with R22	708-2197
86		LAS1	SPL_PL90, 25W laser	N/A	Jameco	OSRAM 2192763
87		LED1	LED, red	1206	RS Components	700-7893
88		Q1	BSP318S, avalanche SIPMOSFET	SOT223	RS Components	753-2816
89		X1	16.369MHz, VCTCXO	2.5x2 PQFN	DigiKey	535-11784-1-ND
90		X2	16.369MHz, VCTCXO	2.5x2 PQFN	DigiKey	535-11784-1-ND

Table 19. Parts List.

Table 20 shows the specifications of the open source LIDAR system OSLRF-01, which heavily inspired our system:

	OSLRF-01
Range	0.5 ... 9 m
Resolution	Adjustable
Update rate	Adjustable: 3 ... 50 readings per second
Accuracy	Adjustable
Power supply voltage	12 V (10 ... 16 V)
Power supply current	50 mA
Outputs & interfaces	Timing & laser signal outputs
Dimensions	27 x 56 x 65 mm
Weight	57 g
Mounting	4 x M3 (3.2 mm diameter) mounting holes
Connection	0.1 in. pitch header
Optical aperture	53 mm
Beam divergence	50 mm at 9 m (approx.)
Laser power	14 W (peak), 6 mW (average), Class 1M
Operating temperature	- 20°C ... + 60°C

Table 20. Open Source LIDAR Specifications.

However, due to design challenges from making a LIDAR system based on the OSLRF, our group ended up needing to purchase a distance sensor. Table 21 shows the specification for the TFmini LIDAR system, which is the end product used in the development of our buggy. The TFmini represents the best balance between price and effective range, and is accurate enough for our purposes. The data in table 21a) shows the max range for the module is 12 meters. It has a maximum distance of around 3 meters in significantly bright ambient light. The buggy will likely be operating in bright ambient light, given that it's operating on a beach. Three meters is an adequate distance to see ahead because the max speed the buggy will be 3 mph. The data in table 21a) shows the side detection length as a function of the distance between the sensor and object. From 3 meters away, the TFmini has a side-detection length of .12 meters, which is acceptable. The operational wavelength is 850nm, but the distance sensor is eye safe because it uses an LED instead of a laser. Other modules considered were either too expensive with unnecessarily long range, such as the LIDAR lite, or have adequate range under ideal conditions but are made useless by ambient light, such as the Sparkfun VL53L1X.Table 21 shows the specification for the TFmini LIDAR system, which is the end product used in the development of our buggy.

a)

Product Name	TFmini
Operating range	0.3m-12m
Maximum operating range at 10% reflectivity	5m
Average power consumption	0.12W
Applicable voltage range	4.5V-6V
Acceptance angle	2.3°
Minimum resolution ratio	5mm
Test frequency	100Hz
Test accuracy	1% (less than 6m), 2% (6m-12m)
Distance detection unit	mm
Operating center wavelength	850nm
Size	42mm×15mm×16mm
Operating temperature	-20℃-60℃
Anti-ambient light	70,000lux
Weight	6.1g
Communication interface	UART

b)

Distance/m	1	2	3	4	5	6
Detection range side length/mm	40	80	120	160	200	240

Table 21. TFmini Lidar Specifications. a) Tfmini characteristics b) Side detection length vs. object distance.

5.2 Solar Cells

In this section we will describe the functional solar panel circuit designs which we will be implementing in our final buggy platform. After an extensive study of similar projects which involved significant dependence on solar power, we were able to incorporate a rough design which would be able to adequately distribute the power generated in the solar cells to the rest of our system. As the solar cell is the lifeblood of our buggy, it is vital that the implementation of the solar panel circuit be well-defined and secure.

Our basic system will consist of two monocrystalline solar panels, placed in prominent positions on top of either the buggy itself or a platform designed for the sole purpose of raising the solar panels to a safe altitude, where they can be made to pan and tilt in order to maximize their efficiency. From there, the current generated in the photovoltaic cells by the impact of the incident sunlight will pass through to a charge controller, which will limit the rate that the current is passed through to the batteries, ensuring safe operation and charge speeds. This current will pass through to one of two batteries, which will charge over time as the other battery discharges in order to power the necessary operational elements of the buggy. These batteries will switch roles as they get discharged, ensuring that the buggy is always operational, provided a bountiful and consistent amount of sunlight.

From the battery that is being used to power the buggy, a current will pass through an inverter, which will convert the DC current provided by the battery to an AC current which can be utilized by the various electrical components of the buggy to function. As we had discussed in the prior sections, we have various promising candidates for the parts we will need to develop and implement this circuit.

5.3 Code Design

This section will detail the design aspects of the current model for the software that will be used to operate the buggy. Just as the physical hardware and materials of the chassis and wheels are vital to ensuring the durability of the buggy, the coding aspects of the computer systems onboard are vital to ensuring that the buggy is able to operate effectively and efficiently once it is assembled, and so it is necessary for us to develop a thorough model of how it is to operate once it is compiled into a single system.

5.3.1 Robot Operating System

ROS is fundamentally a full-featured message passing interface intended for use with robotics. Message passing interfaces are, as implied by the name, a system where one component can send some kind of information to one or more other components or make it available for other components to read. There are numerous predefined messages that can be used which can store data as simple as a single integer or as complex as a fully colored image. In addition, ROS offers the ability to create messages from a user made template if a suitable message type cannot be found.

5.3.1.1 Nodes

Every component in ROS is a “Node”; specifically, they are defined as a unit that does computation. They may additionally be publishers, subscribers, or both. Publishers are able to post messages of a specific type to specific topics, usually at a regular, fixed speed (although they can post as quickly as possible, or only in response to something else). Subscribers listen for messages posted to a specific topic or topics, and generally are event-driven.

5.3.1.2 Advantage of ROS

ROS is the largest, most common framework for programming robots today. Not only that, but it’s also been around since 2007, meaning it is very stable and well supported both by the Open Source Robotics Foundation and by community developers. From a development perspective, the biggest benefit is that nodes can be written in any programming language (although primarily C++ and Python). ROS also has the advantage of having many libraries (called packages) written for it and thus, some of the labor in creating code is reduced, allowing easier implementation of different hardware.

5.3.1.3 Limitations

ROS is a very powerful framework for robotics; however, users are limited to the Linux operating system. At times Linux is not as intuitive as other operating

systems. A basic understanding commands from the Linux terminal is required. As shown above, installation is not as simple as downloading a file from a web browser. Each node, including ROScore, must be run in order for the nodes and its resulting subscribers and publishers to be initialized. However, multiple nodes can be run easily by inserting their run directives in to a bash script.

5.3.1.4 Data Flow

This section details the method in which data input to the system will be processed and describes which nodes will perform what action in order to perform to the expectations we had established.

Figure 33 illustrates the data flow of the buggy with the nodes in circles and hardware in boxes. The buggy has 6 nodes that are important to its operation:

Serial Node—The serial node is how ROS sees the ATMega328p microcontroller; it is part of the *rosserial_python* library and pipes the ATMega328p microcontroller serial input and output into a ROS node. Since two microcontrollers are used in the buggy, two serial nodes will be created named ServoLidar and LidarServo.

Lidar Node—The Lidar node publishes data from the ATMega328p microcontroller controlling the Lidar through the serial node, sending range data from the Lidar with a rate of up to 67 Hz. Even though the Lidar is capable of sending up to 100 Hz, the overhead by the microcontroller, the lidar module itself, and the ROS library limits the ability to send at its maximum rate.

Servo Node—The servo node publishes and receives data to and from the ATMega328p microcontroller controlling the servo through the serial node. It publishes its current rotation angle, as well as receiving directives from the buggy node so that it turns only when directed, preventing it from turning too fast or too slow. This ensures that data from the Lidar are synchronized and connected with the rotation angle of the servo.

GPS Node—The GPS node parses the raw data received from the GPS module and publishes it in a meaningful matter. The published data includes latitude, longitude, track (heading) speed, and the error rates for these data, among others. The GPS node also tracks points that are in front and adjacent to the buggy for area avoidance purposes. If these forward points are in areas of avoidance, the GPS node will also publish to the buggy node an “obstacle” in relation to the buggy position that the point and restricted are collide so that the buggy initiates its obstacle avoidance procedures.

Motor Control Node—The motor control node receives data from the buggy node and converts it in to PWM signals for the motors. It controls the voltage output relative to the power percentage the buggy wants the motor to run, as well as control which motors run in which direction to control the differential drive aspect of the buggy.

Buggy Node—The buggy node is the main control node of the buggy. It contains the avoidance algorithm of the buggy, as well as algorithms that processes Lidar data, servo control, and motor control.

Arrows are labeled with topic names and indicate the data flow coming to or from the node whether it receives data as a subscriber, sends data as a publisher, or both.

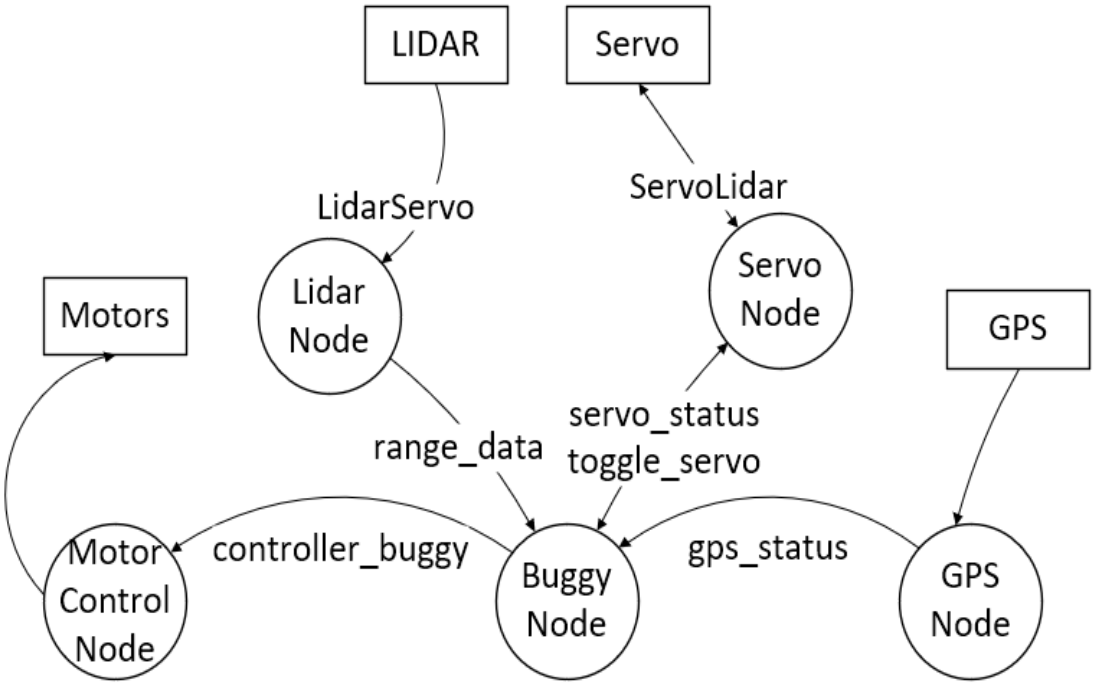


Figure 33: ROS Core Data Flow

5.3.1.5 ROS Messages

Via Topics, Nodes publish data as messages. These messages have different types and are essentially containers for more elaborate data, depending on the message type. The following are the ROS messages utilized in the buggy and are detailed as follows.

1. geometry_msgs/Twist

This message type is used to send motion control from the buggy to the motor control node. It expresses velocity in free space broken into its linear and angular parts. For the purpose of this project, instead of velocity, motor power expressed in percentage is used instead. The message has two components: *linear* and *angular*. Each component has a respective *x,y,z* component. For the purpose of

this project, only the *x* component of the *linear* component is used (to direct forward or backward motion), and only the *z* component of the *angular* component is used (to direct left or right motion).

2. gps_common/GPSFix

This message type contains a wide variety of GPS information. However, not all of them is used in this project and only the following are used and detailed: *latitude, longitude, track,* and *speed.* The *latitude* and *longitude* component are the decimal degree coordinates of the buggy. The *track* is a calculated heading based on previous coordinates and current coordinates of the buggy and is expressed in degrees clockwise from North. The *speed* is the calculated velocity of the buggy based on previous and current coordinates and is expressed in meters per second (m/s).

3. sensor_msgs/LaserScan

This message type contains the range data supplied by the Lidar. Its two most important components are the arrays *ranges* and *intensities* which contains the scans along a 180-degree arc in front of the buggy. The first index of each array represents the range and the intensity of the scan on the rightmost part of the buggy. The last index represents the range and the intensity of the scan on the leftmost part of the buggy.

4. std_msgs/Int16

This is a simple message type which contains only one container, *data,* which is a 16-byte integer data type. For the purpose of this project, this data type is used to send integers that represent a certain status for the respective node that utilizes it.

5. std_msgs/Bool

This is another simple message type which contains only one container, *data*, which is a boolean data type that is either True or False. For the purpose of this project, this data type is used to send data that represent a certain status for the respective node that utilizes it.

5.3.2 Python

Python is an interpreted high-level programming language for general-purpose programming and, except for the microcontrollers which are programmed in C/C++, is the programming language of choice for this project. This section details the resources used with Python in the creation of this project.

5.3.2.1 Python Libraries

Python, being a popular programming language, has a wide variety of libraries written for it. To offload some of the work required for this project, the following, non-default libraries are used.

1. Geopy

Geopy is a Python 2 and 3 clients for several popular geocoding web services. For this project, it is mainly used to calculate the point ahead of the buggy, the point to the left of the buggy, and the point to the right of the buggy from the buggy's current GPS coordinates and its calculated heading.

2. Shapely

Shapely is a BSD-licensed Python package for manipulation and analysis of planar geometric objects. It is based on the widely deployed GEOS (the engine of PostGIS) and JTS (from which GEOS is ported) libraries. Shapely is not concerned with data formats or coordinate systems but can be readily integrated with packages that are. For this project, Shapely is used to determine whether the forward points of the buggy calculated with Geopy are within the off-limit GPS areas specified by the team.

3. RPi.GPIO

RPi.GPIO provides a class to control the GPIO on a Raspberry Pi. For this project, RPi.GPIO is used to set the GPIO pins as output for the Motor Controller and also output the necessary PWM signals to emulate the RMS voltage required for the motor controller to translate it as power to the motors to give them speed.

4. Rospy

Rospy is a pure Python client library for ROS and provides the essential APIs to interface with ROS topics, services, and parameters. This library is an absolute necessity and it controls all the nodes, subscribes, and publishers of the system.

5.3.3 Additional Libraries

This section details other libraries used that are not Python-specific. These libraries are also essential to the system and is mainly used on either our microcontroller or as a parser for the raw GPS output provided by the GPS module.

1. TFMini

TFmini is a library that allows our ATMega328p microcontroller to interface with the Lidar. It provides API directives that allows us to grab input from the lidar, as well as other error-checking algorithms that occurs in the background mainly: checksum verification, garbage data collection, and try-again directives.

2. gpsd

Gpsd is a service daemon that monitors one or more GPSes or AIS receivers attached to a host computer through serial or USB ports, making all data on the location/course/velocity of the sensors available to be queried on TCP port 2947 of the host computer. Gpsd is used as a sort of driver for the GPS module so that it can communicate with the Raspberry Pi and thus, allow the data to be parsed so that it is accessible at the end-user level.

3. gpsd client

Gpsd client is a package for ROS that interfaces with gpsd and then publishes the data provided by the GPS module through gpsd and is run as a node. It publishes the ROS messages GPSFix and NavSatFix messages through publishers it creates, and the data contained within those messages are translated directly from the raw data provided by the GPS module.

5.3.4 Other Software

This section details other software used in the development of the buggy. This software does not run on the buggy itself but nonetheless is vital in the creation of the design and codebase of the working buggy system.

1. Pip

Pip is a package management system used to install and manage software packages written in Python. Similar to *apt-get* but for Python instead of Linux, many of the packages and their dependencies can be found in the official third-party software repository for Python, PyPi.

2. Arduino IDE

The Arduino Integrated Development Environment (IDE) provides comprehensive facilities to write code and program our megaAVR-family chip created by Atmel: the Atmega328p. The Arduino IDE provides resources to program the bootload unto our microcontroller, allowing advanced customization of it such as adjusting the source of its clock, as well as uploading to the microcontroller the compiled machine code written from C/C++ in the Arduino IDE.

3. Gedit

Gedit is the default text editor of the GNOME desktop environment and part of the GNOME Core Applications. Designed as a general-purpose text editor, gedit emphasizes simplicity and ease of use, with a clean and simple GUI, according to the philosophy of the GNOME project. Gedit was part of Harvard's CS50 Appliance Virtual Machine and as such, is the preferred source code editor for this project.

4. VMware Workstation Player

VMware Workstation Player is the free version of VMware Workstation and is a hosted hypervisor that runs on x64 versions of Windows and Linux operating systems. It enables users to set up virtual machines on a single physical machine and use them simultaneously along with the actual machine. Each virtual machine can execute its own operating system. This software is essential as it provides the necessary Unix-like environment to code and debug the software used to run the buggy.

5. Linux Mint Cinnamon

Linux Mint is a community-driven Linux distribution based on Debian and Ubuntu that strives to be a "modern, elegant and comfortable operating system which is both powerful and easy to use." Due to its user-friendliness, this operating system is our operating system of choice.

5.4 Sensor Design

This section will discuss the composition of the sensor technologies, used to interpret data collected by the Lidar system in order to detect objects and maneuver around them. This is the heart of our design, and so requires an extensive design process.

5.4.1 Components

1. Single Board Computer:

In selecting the proper hardware system in providing a back-end for the software required to operate the robot, the following must be considered: its weight, its power draw, its processing power, and its architectural capabilities. The hardware should be powerful enough to be able to control and process the different sensors on the buggy without having too much overhead, allowing a real-time autonomous operation of the vehicle but at the same time, its power draw should be conservative in such a way as to not overdraw from the solar panels and the batteries as power might be needed elsewhere.

The Raspberry Pi 3 was the optimal choice due to its price and its ability to decently run the required software implementations for the buggy, as well as its extensive GPIO pins, as illustrated in Figure 34, allowing flexibility in connecting a wide variety of hardware. The Nvidia Jetson TX1, while impressively powerful, was turned down since it has an MSRP of $300.

2. Servo:

The Lidar alone does not have a built-in servo. As such, it can only see straight and is essentially a laser range finder. Having a servo in lieu with the Lidar will allow us to sweep the entire 180-degree front cone of the buggy for optimal object avoidance. Figure 35 illustrates the pins of the servo, the duty cycle, and its PWM

period. The servo accepts a 5V signal voltage and a 1 ms pulse is all the way to the left, while a 2 ms pulse is all the way to the right. The orange pin will be connected to a microcontroller where its rotation angle is going to be controlled.

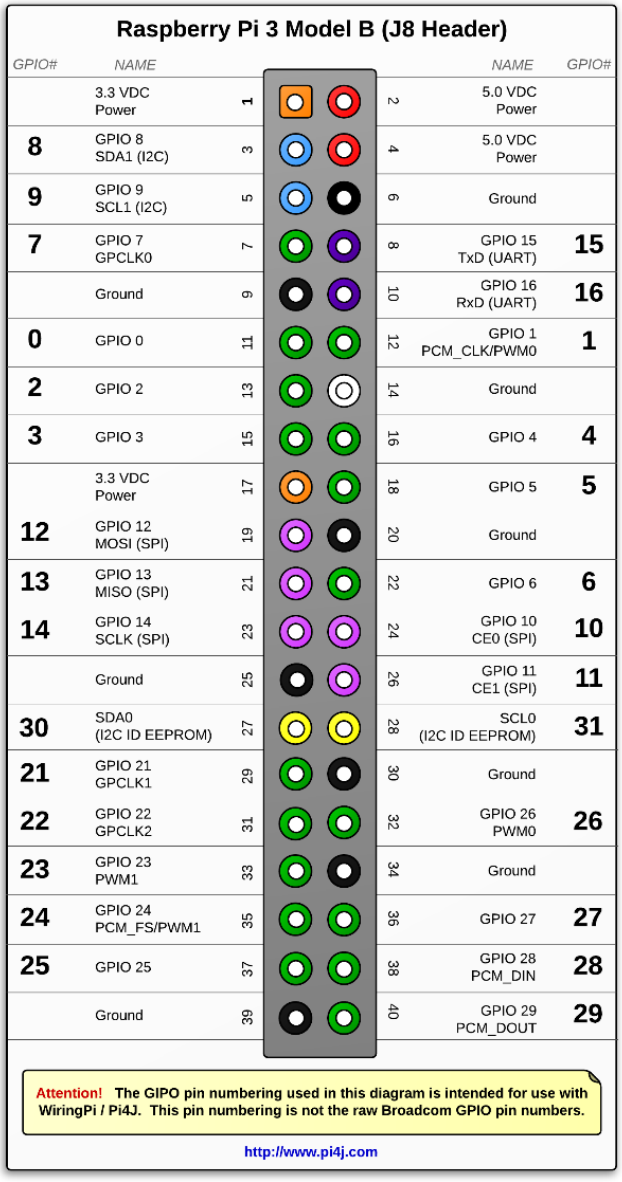


Figure 34. GPIO Pins on the Raspberry Pi 3 B+. *Courtesy of pi4j.com*.

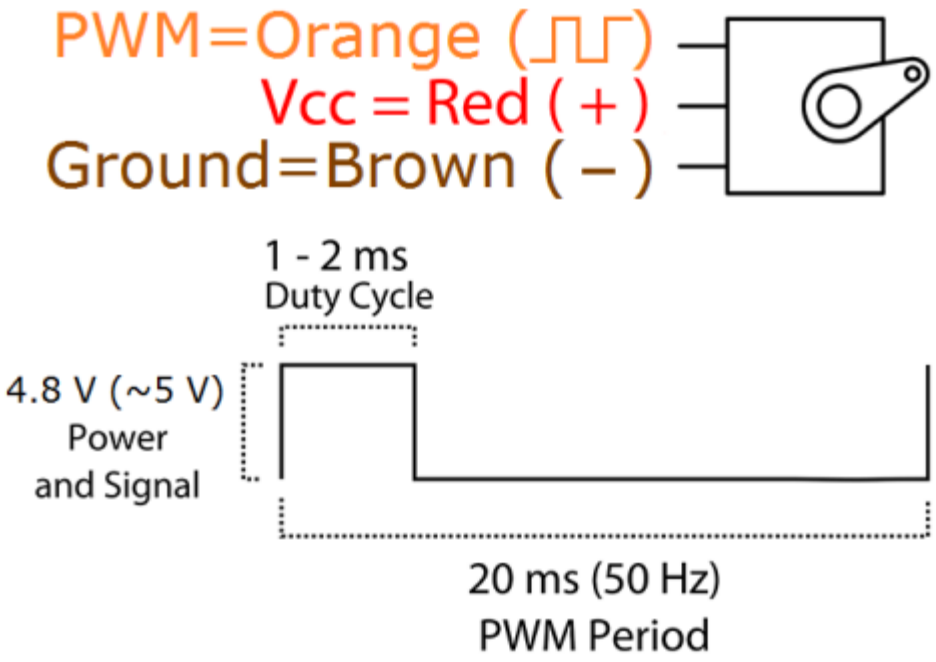


Figure 35. Servo Pins with PWM Period and Duty Cycle.

3. Lidar:

The Lidar is the main component of our sensor system. It has a range maximum of 12 meters with a range minimum of 0.3 meters. It is versatile even on outside sunlight and proves reliable in detecting objects for avoidance. Paired with a servo, the Lidar can sweep 0-180 degrees in front of the buggy at a rate of around 2 seconds per sweep. Figure 36 illustrates the backside of the Lidar along with its pins. As we cannot communicate to the Lidar itself, Pin 2 (RX) is not needed and is not used in this project.

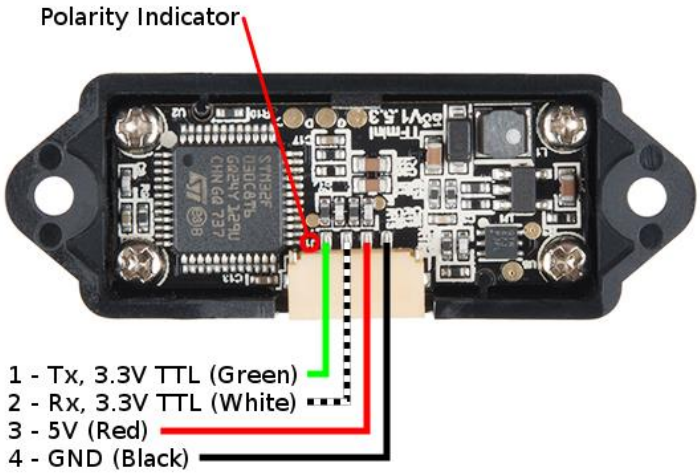


Figure 36. Lidar with I/O pins labeled.

4. ATMega328p microcontrollers

Two ATMega328p-based microcontrollers designed and constructed will each be used to control the servo and the Lidar. The ATMega328p is a megaAVR chip that is popularly used by the Arduino Uno. As such, so long as the circuitry is correct, many libraries used on Arduinos can be used in our ATMega328p microcontroller.

5. TTL to USB converters

Our ATMega328p microcontrollers do not have a built-in TTL to USB connection and thus, a separate TTL to USB converter is utilized. This converter will connect to the TX/RX, 5V, GND, and Reset pins of our microcontroller as input and will have a USB 2.0 Type A output that will connect to the Raspberry Pi.

5.5 Navigation Design

This section will concern the design of the navigation system of the buggy, which will be used to ensure that the buggy does not leave a safe area of the beach and enters, say, open traffic. Such a malfunction would be disastrous to the safety aspects of this project, and so it is necessary to ensure that they are functioning properly.

5.5.1 Components

The navigation system is based on one chip: an Adafruit Ultimate GPS Breakout v3 chip.

Adafruit Ultimate GPS Breakout v3 Chip

The Adafruit Ultimate GPS Breakout v3 Chip is a module developed by Adafruit that contains an MTK3339 GPS chipset. It is capable of connecting to 22 tracking satellites, 66 searching satellites, with a position accuracy of less than 3 meters. It is small, low-power, and versatile for our navigational needs.

5.5.2 Concept

The team identifies the ending point for where the buggy will travel to. The GPS coordinate for the ending point will be hardcoded in to the buggy. The buggy will try to maintain the heading towards this ending point so it will not drift away from its path.

Next, the team identifies off limit areas for the buggy. A series of GPS coordinate points are taken and then combined to form a polygon that coincides with the area

the buggy will try to avoid. Figure 37 illustrates the concept for this. Note that with this design, the buggy can start anywhere as it will be able to correct itself to move towards the proper heading.

Detecting whether the buggy is or is about to reach the designated off-limit areas is a point-in-polygon problem. As such, the proper libraries and methods are used in determining whether the buggy is drifting towards the off-limit areas. When it is, the software will mark the point ahead as an obstacle so that the buggy will initiate its object avoidance directive, which will put the buggy right back in its path.

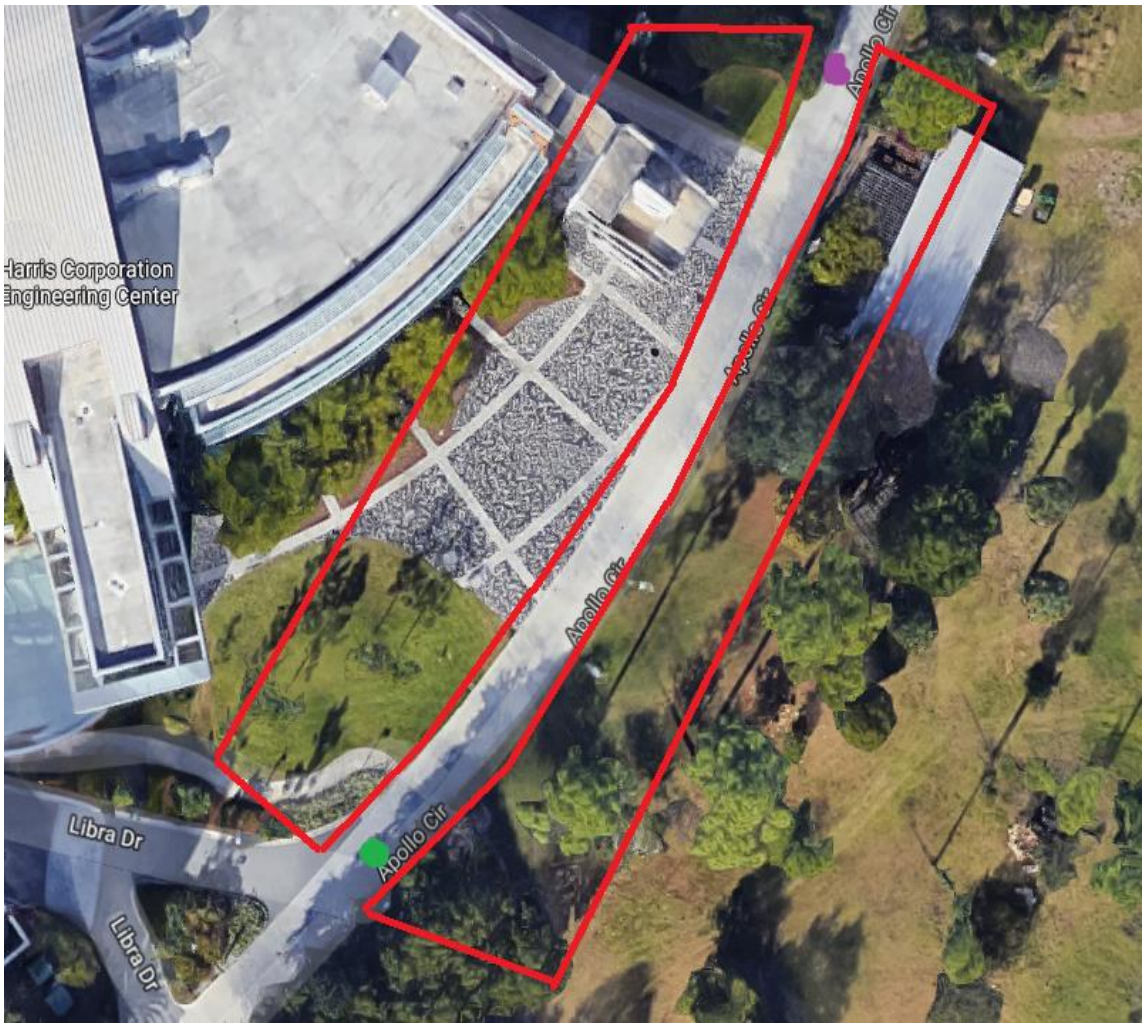


Figure 37: Navigation Map. Red lines indicate the polygons that cover the off-limit areas. The green dot is a sample starting point. The purple dot is the end point.

5.6 Obstacle Avoidance Design

The buggy will use a simple obstacle avoidance algorithm to avoid collision with objects in its path. The buggy will divide its 180-degree frontal cone into three smaller cones: two 75-degree cones for its sides, and a frontal 30-degree cone.

The buggy's default directive is to move forward. If it detects an obstacle in front, it will check its right cone for any obstacles. If there is none, it will turn toward that direction. If there is, it will check its left and do the same. If there are obstacles on all of its cones it will stop but will continue scanning in case one of the detection cones clears up of obstacles.

5.7 Software Design

The buggy will incorporate all the design in the code, sensor, navigation, and obstacle avoidance design in order to form a unified system that will drive the buggy autonomously. Figure 38 illustrates the software flow of the final design and demonstrates the logic process of the buggy.

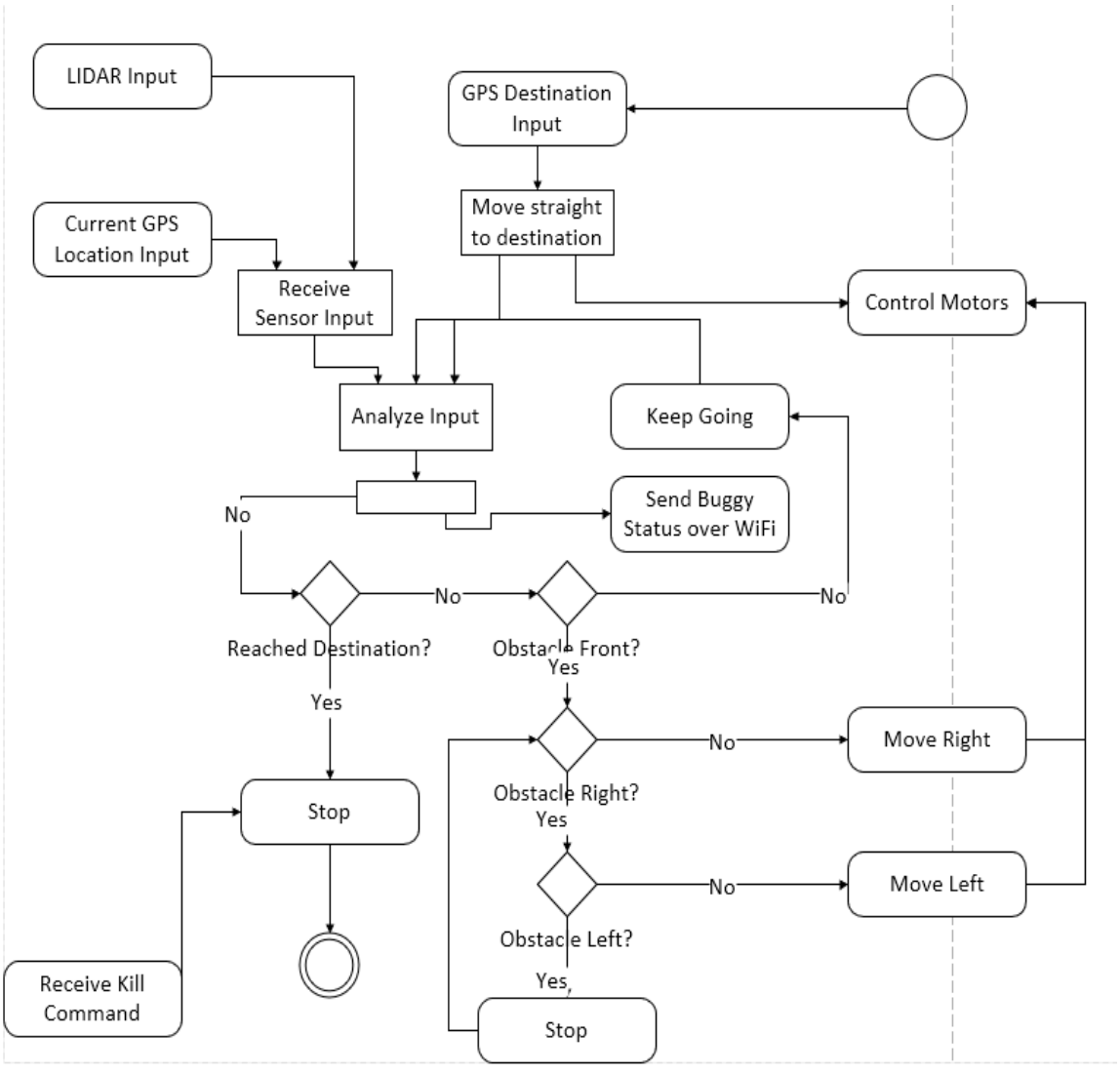


Figure 38. UML Activity Diagram.

Due to the PWM nature of the servo, the Lidar cannot be operated with the same microcontroller as this disables the Lidar when done so. As such, the servo and Lidar are separated in to different microcontrollers, but both are connected via the Buggy Node on the Raspberry Pi.

Also, due to the size of the *range* and *intensities* arrays, there is a huge overhead when such an array is sent from the microcontroller to the Buggy Node. Thus, the Lidar only sends its current reading and the Raspberry Pi will handle the processing of placing the data in to arrays and interpolating from that the range reading positions relative to the buggy.

To summarize, the Lidar constantly scans and publishes range data. However, the servo does not move until the buggy node tells it to. The buggy node puts the gathered range data in to its respective array, tells the servo to move to the next point of its scan arc, grabs the range data and puts that in to the next index of the array, and so on. It does this from left to right after which it will consider it as a complete sweep then does the same process again from right to left albeit this time, the array is reversed at the end of the right-to-left sweep to conform with the convention that the first index of the array is the leftmost point.

The servo-lidar system does a total of 60 scans with each scan 3 degrees apart.

In a separate thread, the GPS node constantly monitors the position of the buggy. Should the buggy achieve 0.3 m/s, the GPS node assumes that it is fast enough to get a more accurate heading. From this heading, it calculates GPS points that are in front and to the sides of the buggy. It then determines via the point-in-polygon problem if these GPS points are within the bounds of the off-limit areas. If they are, the GPS node sends an “obstacle” to the buggy node so that it will perform an evasive maneuver to avoid the area.

Meanwhile, as a sweep is completed, the array is analyzed and divided in to cones. The default directive of the buggy is to move forward. Depending on whether a cone has an obstacle in it or not, it will move in to that direction. If there is an obstacle in front, it will move right. If there’s an obstacle right, it will move left. If it still detects an obstacle on the left, it will stop and will continue once a cone is clear of obstacles. It should also be noted that it is in this logic that GPS “obstacles” are processed. So, if the GPS says there’s an “obstacle” to the right, it will treat it as if there is a real physical obstacle to the buggy’s right.

The directive to move is sent as a message to the motor control node which sends the PWM outputs for the motor controller. The buggy node sends the percent power it wants the motors to run and the motor control node converts this in to a PWM voltage respective of that percentage. The motor controller is configured to run 100% power at an input of 3.3V with 0% power at 0V. The direction of each motor is controlled by separate GPIO pins wherein an output of 3.3V will direct the motor to move forward while 0V direct it to move backwards. Since the buggy is differential drive, the motor control node also controls the direction of each wheel so that the buggy can turn accordingly.

As defaults, the buggy is set to move forward at 30% power and the buggy turns with 1% on each wheel such that it does not turn too fast that the Lidar does not complete its sweep.

5.8 Hardware Design

In this section, we will discuss the hardware design process in detail. We discussed, in previous section, the part selection process for our project, and here we will show the implementation.

Below in Figures 39, 40, 41, and 42 are the schematic designs of the circuit that the team will use to control the sensors and microcontrollers of the beach buggy.

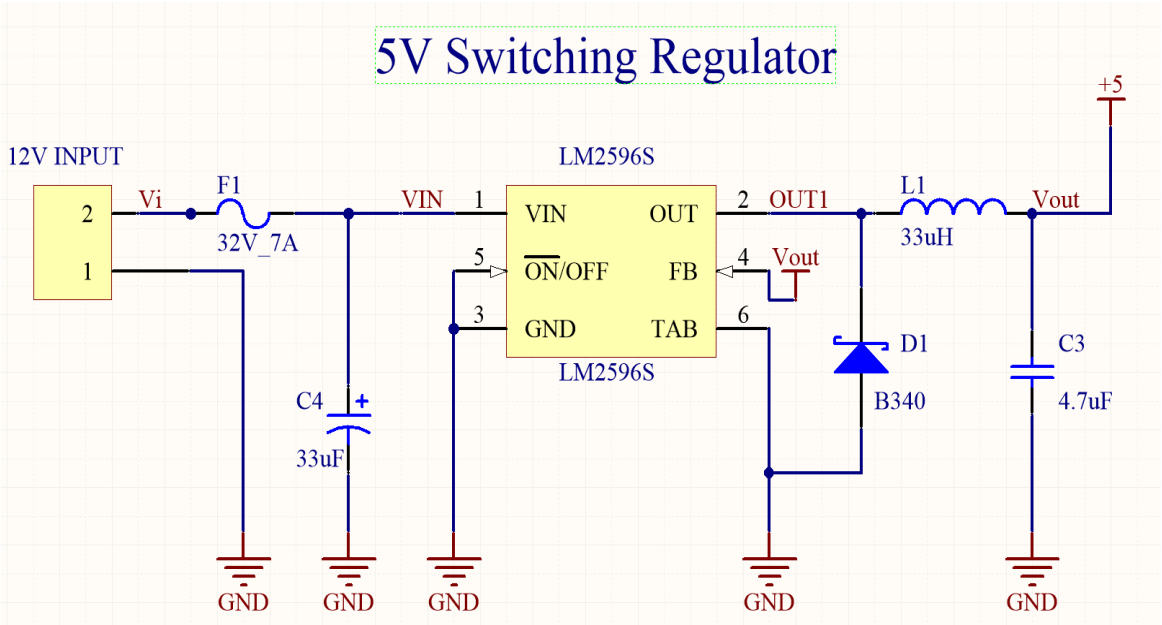


Figure 39: 5V Switching Regulator

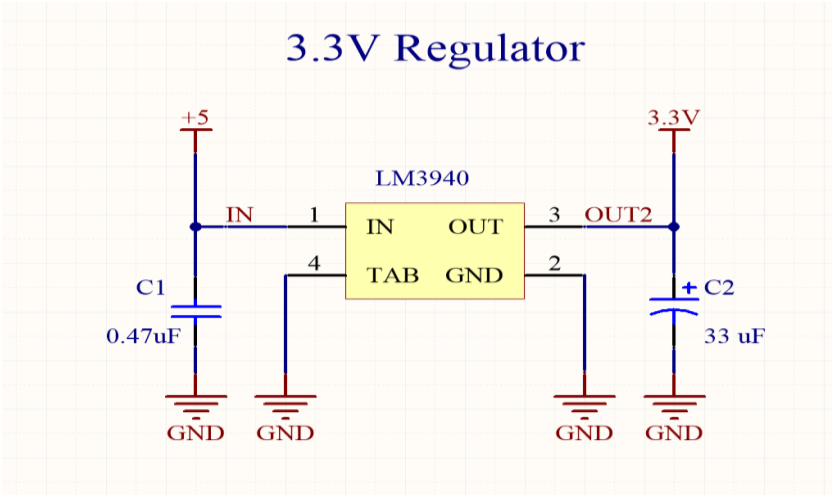


Figure 40: 3.3V Voltage Regulator

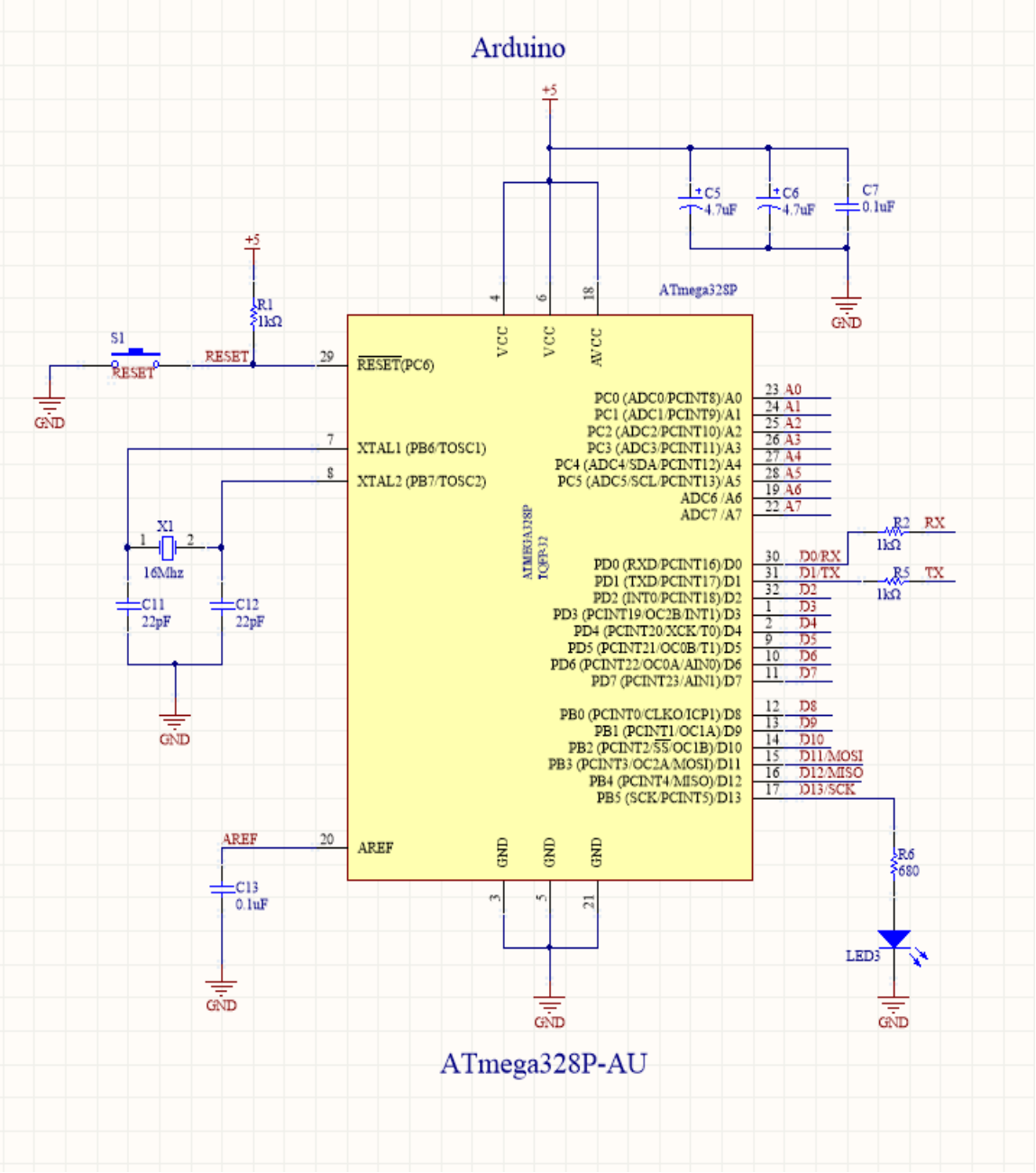


Figure 41: Microcontroller Circuit

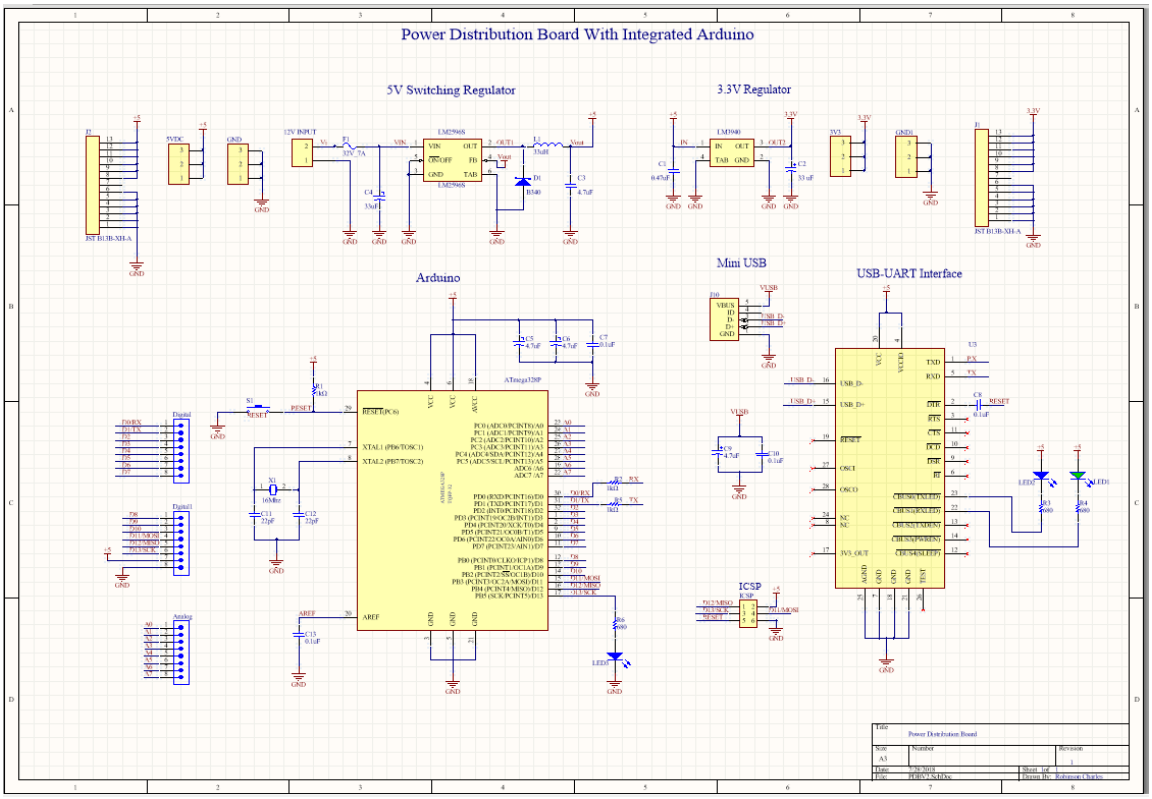


Figure 42: Overall Circuit. Power Distribution Board

5.8.1 Breadboard Design

The circuit schematic in figure 46 is then built and tested in a breadboard. We connected the input to a 14V power supply. The output of the 5V regulator was tested by placing the two leads of the multimeter to the positive and ground and record the reading. We found that our output voltage was 4.97V. This was well within our accuracy requirement. Similarly, the output of the 3.3V regulator was probed and a reading of 3.26V was recorded. The microcontroller was flashed and tested using the ICSP (In Circuit Serial Programming) pin. Pin 1, 3, and 5 were connected to D12, D13, and RESET respectively, while pin 2, 4 and 6 were connected to 5v, D11 and Ground. A simple sketch was sent to the chip- a blinking led program- and the led was blinking every second as expected. We then proceeded to program the ATmega328p. Figure 43 shows the circuit on a breadboard.

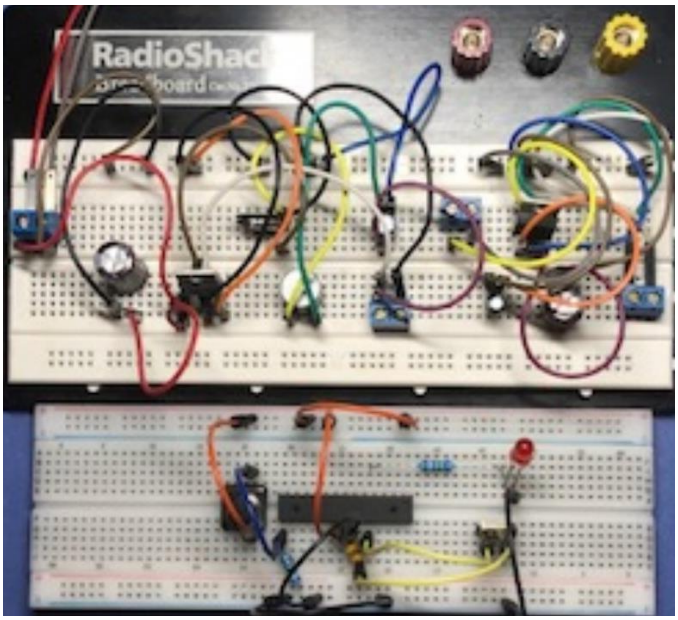


Figure 43: Schematic Diagram on a breadboard

5.8.2 PCB Design

For this project, we considered multiple EDA tools: EasyEDA, CircuitMaker, KiCAD, OrCAD, and CadSoft EAGLE. We ended up choosing Altium CircuitMaker, a free community-driven design tool design by Altium Designer that has same feel and look of the Altium Designer Software, one of the industry standard EDA tools.

5.8.3 PCB Vendor and Assembly

When it comes to choosing a PCB vendor, we are looking for a reliable company that produces great PCBs at an affordable price. This section briefly discusses the three companies we have considered and the services they offered.

One of the company that offers this service is ExpressPCB. They offered different types of PCBs from the standard MiniBoard Service to the ProductionPlus Service. However, we looked at the MiniBoard option, their least expensive option; these board have a fixed size of 96.52 x 63.5 mm. They are a two-layer PCB boards with no solder mask or silkscreen. For $51, one can expect 3 PCB boards with one day lead time.

Another company that offers PCB assembly is JLCPCB. JLCPCB is a high-tech manufacturer specializing in quick PCB prototype and small batch production. It offers 10 boards 100 x 100 mm or less for just $2 with a lead time of just 2-3 days.

OSHPARK is a community printed circuit board company that makes good quality boards which are manufactured in the United States and shipped for free to

anywhere in the world. They charge $5 per square inch for 3 boards with a 12 days lead time. For a 4 x 4 inches board, it will cost $80.

After we considered all available options, we chose JLCPCB services. They offered 10 boards for less money than the other PCB manufacturers we considered. The advantage of using their services over the other is that you can order all the components needed for the assembly process along with the PCB order. All we had to do was to upload the BOM (Bill of Materials) file to their website and the components ordering process was taking care of.

We then ordered components for the schematics design. Once we received the boards and the components, we started assembling the board.

5.9 Buggy and Electronics Platform

Due to scheduling conflicts between the two disciplinary teams, the non-electrical portion of this project was not able to be constructed by the mechanical engineering team. In order to produce a functional prototype, we decided to produce a temporary, makeshift buggy which utilized an existing frame from a commercially available product. As our designs are solely responsible for the navigational functionality of the buggy and are therefore not meant to fulfill the weight requirement specified for the project, we opted to utilize a small, child-sized toy which resembled an all-terrain vehicle. This toy came with two motors included, which allowed us to very easily modify the system and use the differential steering techniques that we intended for use in the finalized buggy. By placing a piece of plywood on top of the buggy, and securing it to the main body, we were able to adapt the toy into a functional and moderately robust platform on which we could mount our electronic system and connect it to two differential motors, in order to quickly and easily test the functionality of our electronics and programming.

6. Prototype Testing and Construction

This section covers all the prototype testing for both hardware and software.

To make sure our design will be working when it comes to the presentation time, we tested every components and part before we set them up. If either a hardware or software components fail our test, a new part will be ordered.

6.1 Hardware Testing

This section focuses on hardware testing to ensure that all components are working and connected properly. This is an important step because the software testing depends on a good working hardware design.

6.1.1 Testing Environment

The beach buggy is designed to travel at the beach therefore all testing will be outdoors. The components for the PCB will be tested indoors. This will ensure that we have access to all necessary equipment to conduct the tests.

6.1.2 Solar Panel

Table 22 shows the testing procedure of the solar panel.

Test Objective	To test whether the solar panel will provide 12V output
Equipment	1. Solar panel 2. Multimeter 3. Wires
Preparation	1. Place the solar panel under the sunshine 2. Connect the two wires to the solar panel
Procedure	1. Connect the two leads (black and red) of the multimeter to the – and + wire of the solar panel
Expected Result	The multimeter should show a reading of at least 12V

Table 22: Solar Panel testing module

6.1.3 Solar Charge Controller

Table 23 shows the testing procedure of the solar charge controller.

Test Objective	To test whether the solar charge controller will charge the battery.
Equipment	1. Solar panel 2. Solar charge controller 3. Battery 4. Wires 5. Multimeter
Preparation	1. Record the charge on the battery with a multimeter 2. Place the solar panel under the sunshine 3. Connect the battery to the charge controller 3. Connect the two wires of the solar panel to the charge controller 4. After a few hours, take a new reading of the battery
Procedure	1. Connect the two leads (black and red) of the multimeter to the – and + wire of the battery
Expected Result	The multimeter should show an increased reading.

Table 23: Solar Charge Controller testing module

6.1.4 Battery

Table 24 shows the testing procedure of the battery.

Test Objective	To test whether the battery is fully charged
Equipment	1. Multimeter
Preparation	1. Identify + and – of the battery terminal
Procedure	2. Connect the two leads (black and red) of the multimeter to the – and + of the battery terminal
Expected Result	The multimeter should show a reading of at least 12V

Table 24: Battery testing module

6.1.5 Motor Controller

Table 25 shows the testing procedure of the motor controller.

Test Objective	To test whether the motor controller works
Equipment	1. Motor Controller 2. Wires 3. Motors 4. Laptop 5. Power supply 6. Atmega328p
Preparation	1. Connect the motor controller to a power supply. 2. Connect Atmega328p to the motor controller.
Procedure	1. Check all connections. 2. Load program to the motor controller. 3. Execute the test program.
Expected Result	The motor controller controls the motor as expected.

Table 25: Motor Controller testing module

6.1.6 Motor

Table 26 shows the testing procedure of the motor.

Test Objective	To ensure the motor is operational
Equipment	1. Motors 2. Wires 3. Multimeter
Preparation	1. Connect the wires to the motor.
Procedure	1. Set the multimeter to Ohm 2. Connect the two leads (black and red) of the multimeter to the motor wire
Expected Result	The multimeter should display an ohm reading.

Table 26: Motor testing module

6.2 Software Testing

This section focuses on software testing to ensure that all components are working and runs properly. Appendix 9.2 contains detailed instructions in running command line directives.

6.2.1 Testing Environment

Software testing is done with all the components, except the motors, connected. The Raspberry Pi is connected wirelessly via the 802.11 protocol to a laptop running VMware Workstation Player with Linux Mint as its guest operating system.

6.2.2 Lidar Node

Table 27 shows the testing procedure for the Lidar Node.

Test Objective	To ensure that the Lidar Node runs properly
Procedure	1. Initiate the Lidar node at port */dev/ttyUSB0* with a baud rate of 57,600. 2. Echo the *range_data* topic to ensure that updated data is being broadcasted.

Expected Results	1. *LidarServo* node is initialized. 2. *range_data* topic is initialized. 3. *range_data* topic publishes updated messages from the Lidar.

Table 27: Lidar Node testing module

6.2.3 Servo Node

Table 28 shows the testing procedure for the Servo Node.

Test Objective	To ensure that the Servo Node runs properly
Procedure	1. Initiate the Lidar node at port */dev/ttyUSB1* with a baud rate of 57,600. 2. Echo the *servo_status* topic to ensure that updated data is being broadcasted.
Expected Results	1. *ServoLidar* node is initialized. 2. *servo_status* topic is initialized. 3. *servo_status* topic publishes updated messages from the Servo.

Table 28: Servo Node testing module

6.2.4 GPS Node

Table 29 shows the testing procedure for the GPS Node.

Test Objective	To ensure that the GPS Node runs properly
Procedure	1. Disable conflicting gpsd sockets 2. Hook the GPS input */dev/ttyAMA0* to a gpsd socket. 3. Initialize the gpsd client. 4. Initialize the GPS Node.
Expected Results	1. *gps_processor* node is initialized. 2. *extended_fix* topic is initialized. 3. *extended_fix* topic publishes new messages from the GPS. 4. *gps_status* topic is initialized. 5. *gps_status* topic publishes new messages from the GPS.

Table 29: GPS Node testing module

6.2.5 Motor Control Node

Table 30 shows the testing procedure for the Motor Control Node.

Test Objective	To ensure that the Motor Control Node runs properly
Procedure	1. Initialize the Motor Control Node. 2. Publish Twist messages as *controller_buggy* topic to the motor control node. 3. Hook up a multimeter to the output GPIO pins of the node.
Expected Results	1. *listener_control* node is initialized. 2. GPIO pins controlled by the node shows changing voltages (when measured by a multimeter) and controls respective voltages when twist messages are sent to it.

Table 30: Motor Control Node testing module

6.2.6 Buggy Node

Table 31 shows the testing procedure for the Buggy Node.

Test Objective	To ensure that the Buggy Node runs properly
Procedure	1. Perform the preceding procedures to initialize every other node. 2. Initialize the buggy node. 3. Test the Lidar.
Expected Results	1. *buggy_control* node is initialized. 2. *toggle_lidar* topic is initialized. 3. *controller_buggy* topic is initialized. 4. *toggle_lidar* topic publishes updated messages from the buggy node. 5. *controller_buggy* topic publishes updated messages from the buggy node. 6. The servo sweeps back and forth. 7. The Lidar publishes update range data in the output terminal.

	8. The GPIO pins controlled by the motor control node changes voltage depending on where an object is detected by the Lidar.

Table 31: Motor Control Node testing module

6.3 Construction

This section will be used to detail the process by which various components of the project were assembled. Just as it is vital to know how to test a system and ensure its functionality, it is also of great importance to know how a system is assembled in the first place. For this reason, this section will go into significant detail on this process.

6.3.1 Buggy Construction

In this section we will go into detail on how the temporary buggy we used was created. This buggy was a makeshift solution agreed upon by the ECE team due to scheduling conflicts with the mechanical engineering team which prevented them from assembling the final buggy. As such, this buggy was constructed with the idea that it was a temporary platform, meant to do nothing more than provide a base with which to attach the necessary electrical components and demonstrate the functionality of the system which we had developed.

We began by procuring a small, child-sized car from a local flea market. This toy was roughly a foot high and two feet long, and consisted of four wheels, in two sets of two on the front and back of the buggy. One of these sets was equipped with two motors, one for each wheel, which provided the operating capability of the toy. This was suitable for our needs, as we had intended to utilize differential steering in our finished product, and this configuration meant that we could devise code to operate these motors which would be immediately transferable to a final product.

We had determined that it would allow the buggy more ease of motion if the non-motorized wheels where removed and replaced with free-rotating casters. This would allow the buggy to make stationary turns, which is more time and energy efficient than it would be to force the buggy to reverse and make wider turns every time it encountered an obstacle. In order to achieve this, we removed the axle that was holding the wheels in place and attached a piece of wood to the frame instead. The steps of this process are depicted visually below in figures 44, 45, 46, and 47.

Figure 44: Forward profile of buggy sans rear axle.

Figure 45: Side profile of buggy sans rear axle.

Figure 46: Front view of buggy sans rear axle.

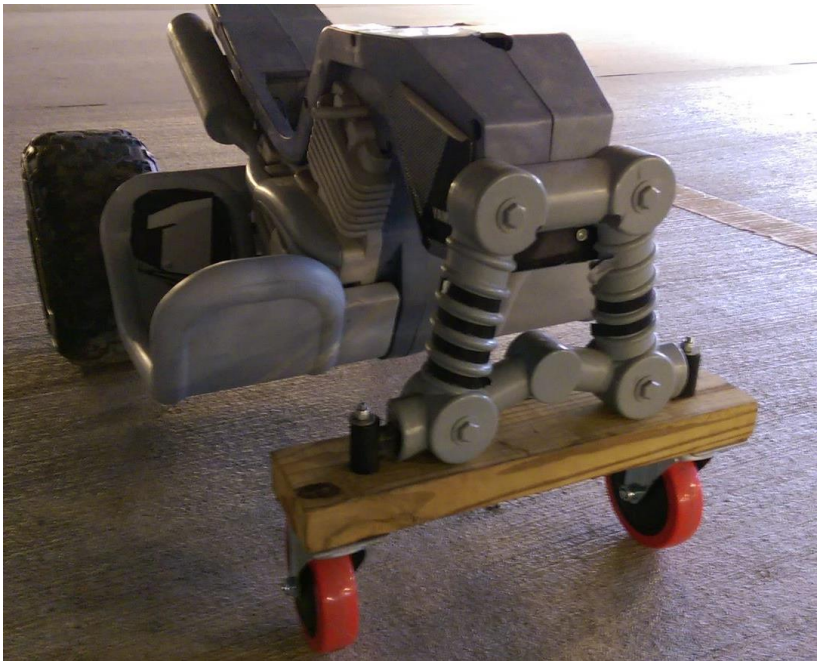


Figure 47: Rear view of buggy, with wheels replaced by freely rotating casters.

The design of the toy we purchased ended up being quite convenient in unforeseen ways. Namely, that the foot rests on either side of the seat were of a size that allowed us to use them as a holding area for the batteries that power our system. We completed the enclosure by adding one more piece of wood on either side, and for aesthetic purposes we painted the majority of the buggy with black paint. We also procured a large piece of plywood which we could use as the platform and secured it to the top of the buggy. These facets are shown below in figures 48, 49, and 50.

Figure 48: Front view of the painted buggy, with one battery and the platform on top.

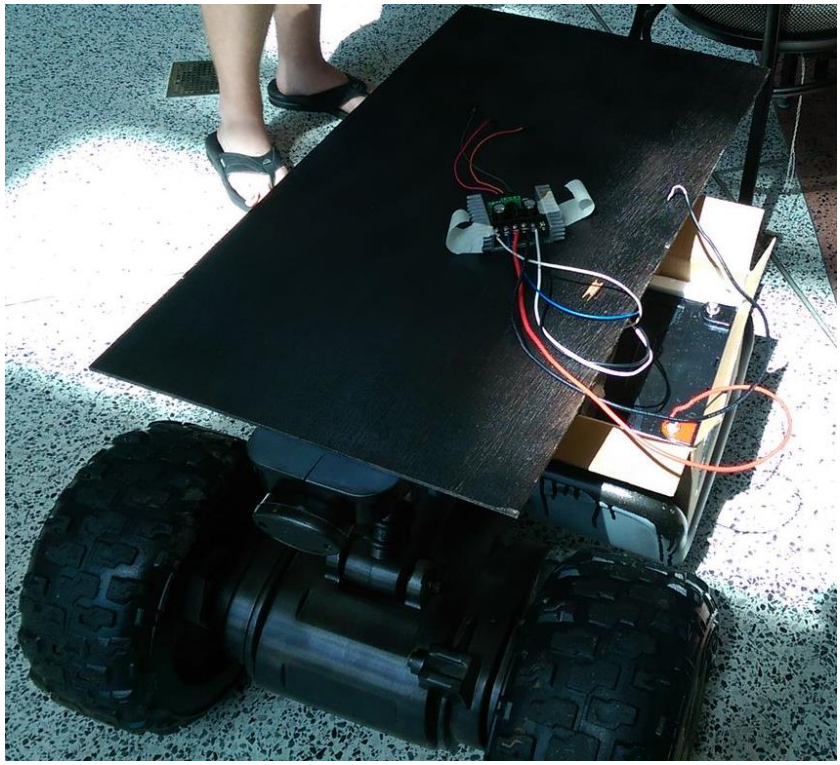


Figure 49: Top view of buggy with platform and battery.

Figure 50: Alternate angle of painted buggy.

Once we had the battery enclosures and the platform, it was a simple matter to place the completed circuit on top of it, mount it to the platform, and then secure the platform the buggy itself. It was important to make absolutely certain that no damage was likely to be incurred on either the batteries or the electrical components. The figures 51 to 57 below show several angles of this process and the components used, such as the battery, the solar charge controller, and the microcontroller.

Figure 51: The battery used to power our circuit.

Figure 52: Size comparison of battery terminal with a standard American dime.

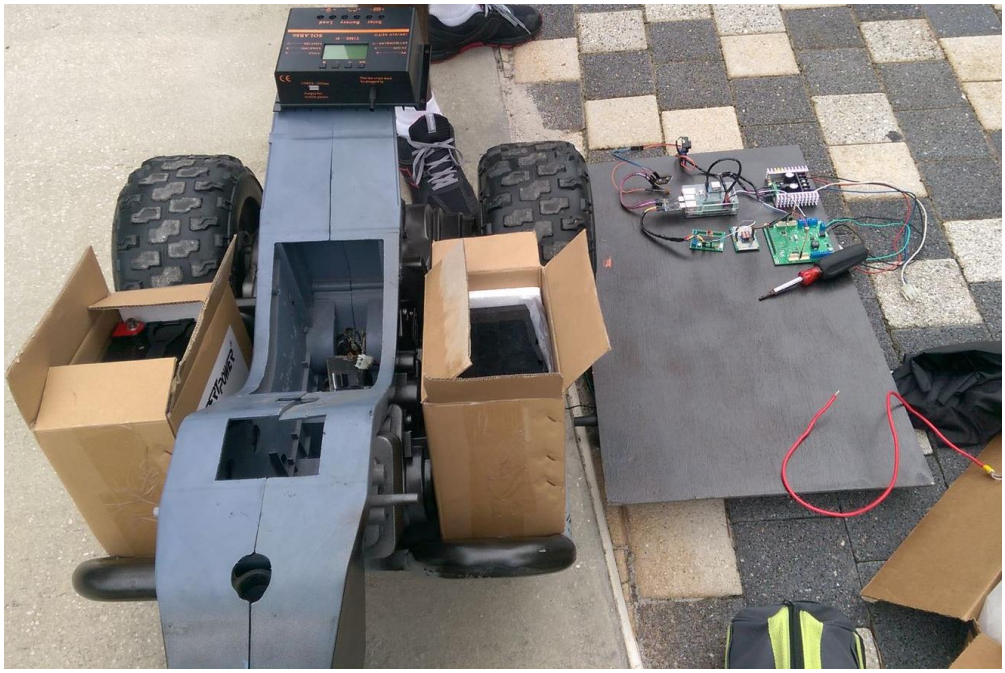


Figure 53: View of buggy with both batteries in their enclosures, the electronic circuit, and the unsecured platform.

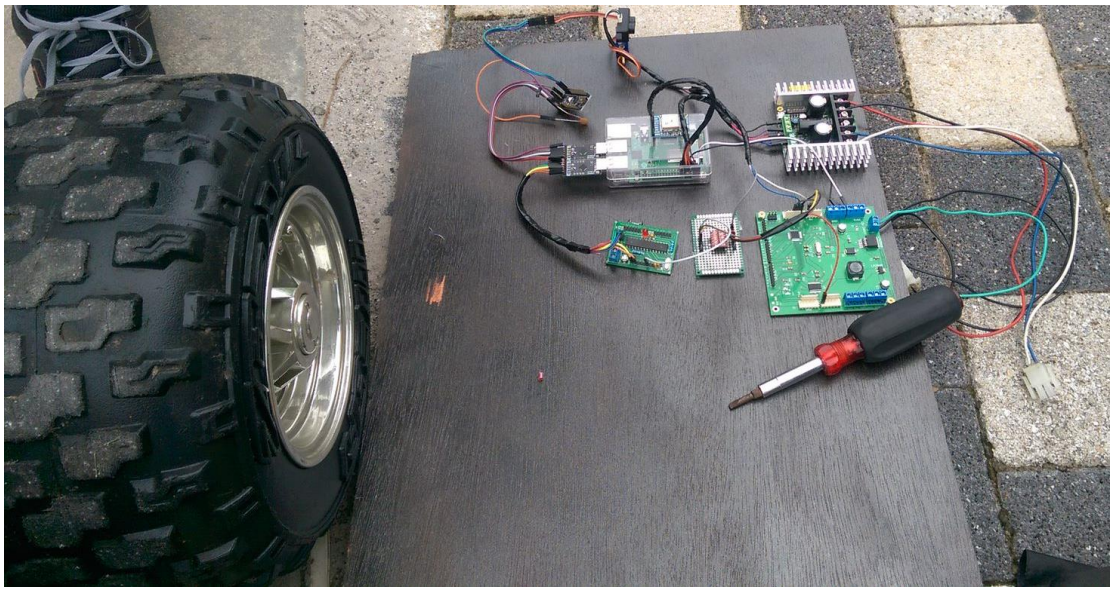


Figure 54: Closer view of the circuit secured to the platform.

Figure 55: View of buggy with platform secured.

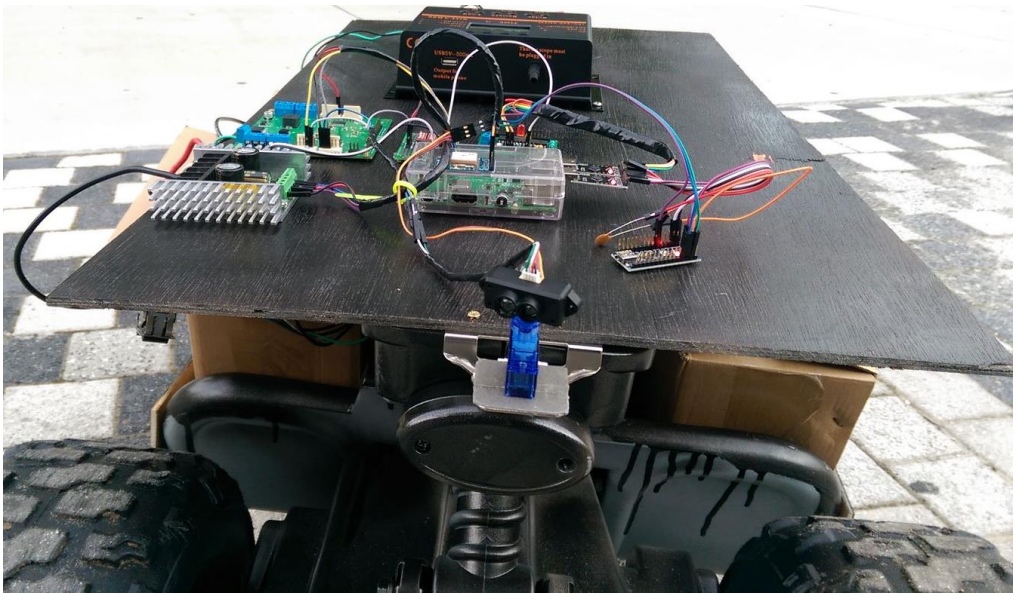


Figure 56: Front view of fully assembled buggy, with Lidar module mounted to a door hinge for adjustable scanning angle.

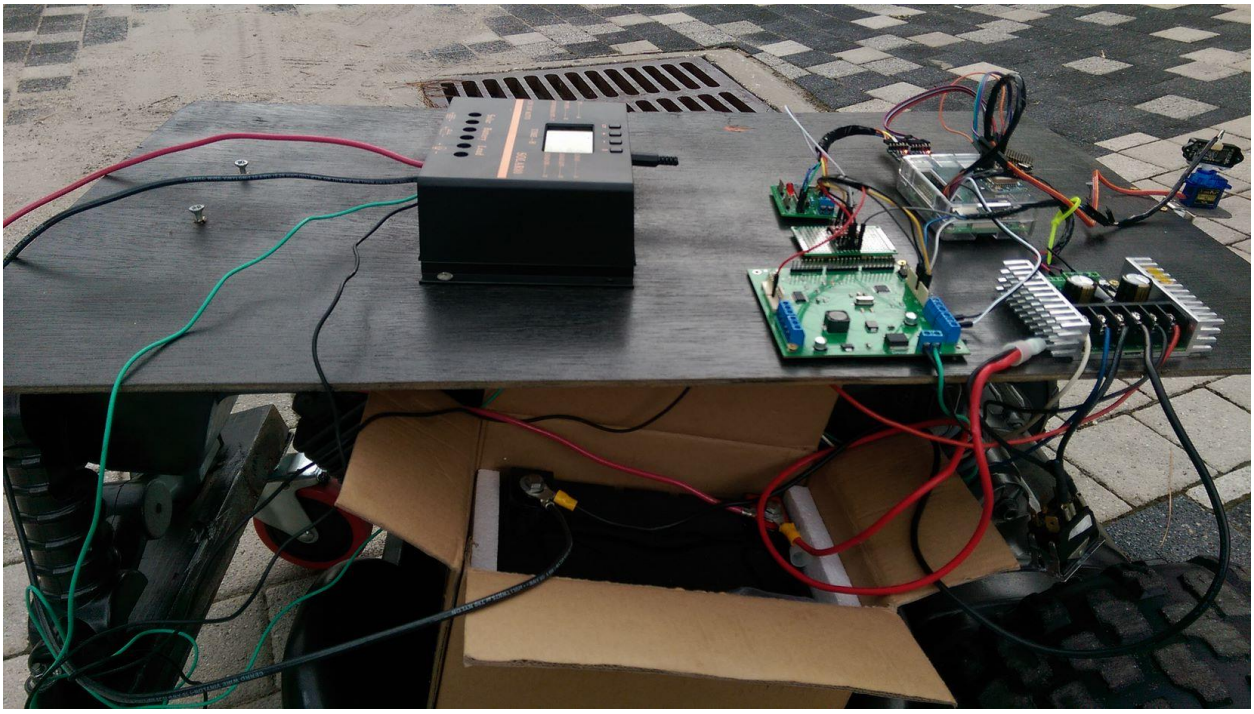


Figure 57: Side view of fully assembled buggy, with batteries connected to the solar charge controller.

7.0 Administrative Content

This chapter describes the agreed time schedule and budget of the team. These were agreed upon through both discussions within the team, and discussions among the other teams in the buggy project.

7.1 Milestone Discussion

This section breaks down the time allotment planned for the completion of the project. Table 32 lists major project milestones and their approximate times of completion.

Senior Design 1		
Number	Milestone	Planned Completion Week
1	Brainstorming	1-2
2	Project selection	2
3.	Divide and conquer	3
4.	Research and documentation	4-11
5.	Table of contents	12
6.	Writing	13
7.	Design	14
8.	Final Senior Design 1 Paper	15
Senior Design 2		
Number	Milestone	Planned Completion Week
9.	Order PCB and parts	1-2
10.	Build prototype	3-4
11.	Testing and revisions	5-9
12.	Final Testing	10-11

13.	Final Report	11-12
14.	Presentation	12

Table 32. Major Project Milestones

7.2 Budget and Finance Discussion

This section will discuss the distribution of the budget for this project. $2000, provided by Duke Energy, is split between the Mechanical Engineering team and the Electrical and Computer Engineering team. The size of the split is dependent on the basis of importance of parts that falls under a team, to the overall function of the system. It has been agreed upon that since the mechanical system is a vital and also the most expensive part of the project, a good portion of the budget will fall under the Mechanical Engineering team, with the motors and tires being the most expensive parts of the buggy. As a result, our electrical systems were produced with the limitation that they do not exceed $1000, as this is the minimum amount of money required by the mechanical engineering and computer engineering teams in order to create their portions of the project. The table below shows the breakdown of our costs for this project.

Item	Supplier	Quantity	Total Price
Misc. PCB Components	Digikey	~100	$95.49
LDO Voltage Regulators	Mouser	3	$4.77
Raspberry Pi	Adafruit	1	$35.00
GPS HAT	Adafruit	1	$44.96
Active GPS Antenna	Adafruit	1	$12.95
Adapter Cable	Adafruit	1	$3.95
Passive GPS Antenna	Adafruit	1	$3.95
16GB Micro SD Card	Amazon	1	$8.55
32A Motor Driver	Amazon	1	$119.95
12v Deep Cycle Battery	Amazon	2	$129.98
24V Motor	Amazon	2	$159.98
80A Charge Controller	Amazon	1	$72.88
GPS Breakout	Adafruit	1	$39.95
Servo Motors	RobotMarketplace	1	$11.49
TF Mini Lidar Module	RobotMarketplace	1	$39.00

Buggy Components	Misc.	~5	$20.00
Total Cost			$802.85

Table 33. Project budgeting

8.0 Conclusion

After an intensive, multi-semester long study of this design project and all of its requisite components, we believe we have developed a very firm understanding of the challenges we face as engineers, and the methodology that goes into selecting techniques and parts to solve them. Our buggy was mandated to be completely self-sufficient, relying only on solar panels, and to that end we have identified some of the most efficient, cost effective circuit components we could find, and have assembled them into a system which can create a large enough current to power two motors and a host of necessary computer systems. Our buggy was mandated to be able to identify and avoid collision with objects, and after we assembled and install our Lidar system and programmed the Arduino to recognize objects from the cloud of distance points it generates, it was able to successfully maneuver around obstacles in it's environment, and it is be aided in this by a robust navigational system and an advanced microcontroller. Our buggy was mandated to be able to operate for a distance of 20 miles, and it can as a careful analysis of the power costs vs. the power generated by our solar cells allows us to design around the power limitations of our solar cells.

Our ECE group undertook Senior Design II in the summer term of 2018, while the portions of our larger group, consisting of mechanical engineers and computer scientists, will be taking the course in the fall of that same year. As a result of this discrepancy, were unable to produce the buggy chassis designed by the mechanical members of our team. Therefore, we sought out a substitute vehicle of comparable complexity and design and attempt to tailor our power and detection systems to the substitute vehicle’s design. In doing this, we had managed to create a working model car that demonstrates the validity and effectiveness of our Lidar and solar power systems, both with regards to providing the ability for a vehicle to autonomously detect obstacles and drive around them, and in general operational terms. By demonstrating that our designs are functional on a smaller scale vehicle, it would stand to reason that they would be applicable on the full-scale buggy that we had originally intended.

This project provided us with valuable insight into the workings of circuit design, Lidar design, Lidar programming, system wiring, battery circuits, solar cells, and more. After two semesters of diligent study, we feel that we were able to finish this course with a greater understanding and appreciation for the challenges that professional engineers struggle with on a daily basis. With this, we look forward to broadening the scope of our expertise and pursuing a long a fruitful career in the fascinating world of engineering.

8.1 References

1) BAJA SAE Forums. http://forums.bajasae.net/forum/monoshock_topic723.html

2) AutoNEWS. “Das ist der Leichtbau-Offroader Ariel Nomad.” *Irrer Offroad-Racer Ariel Nomad: Infos, technische Daten und Preis (Bild 2)*, www.auto-news.de/auto/news/bildergalerie_Irrer-Offroad-Racer-Ariel-Nomad-Infos,-technische-Daten-und-Preis_id_36251&picindex=1.

3) Bureau, The Masterbuilder. “Analysis of Space Frame Structure.” *The Masterbuilder*, 4 Nov. 2017, www.masterbuilder.co.in/analysis-space-frame-structure/.

4) Life of Riley. “Space frame.” *Wikipedia*, Wikimedia Foundation, 5 Nov. 2017, en.wikipedia.org/wiki/Space_frame#/media/File:Articulacion_malla.svg.

5) *Young's Modulus - Tensile and Yield Strength for common Materials*, www.engineeringtoolbox.com/young-modulus-d_417.html.

6) “How To Prep Your ATV For The Sand Dunes.” *MotoSport*, 31 Dec. 2013, www.motosport.com/blog/how-to-prep-your-atv-for-the-sand-dunes.

7) Ken962. “Brushless DC Electric Motor Torque-Speed Characteristics.” *Wikimedia Commons*, 6 Nov. 2010, commons.wikimedia.org/wiki/File:Brushless_DC_Electric_Motor_Torque-Speed_Characteristics.png.

8) kennybobby. “Ev Conversion Torque Calculation and Motor Specifications.” *DIY Electric Car Forums RSS*, 14 Oct. 2013, www.diyelectriccar.com/forums/showthread.php/ev-conversion-torque-calculation-and-motor-89694.html

9) Hymel, Shawn. “Alternating Current (AC) vs. Direct Current (DC).”*Alternating Current (AC) vs. Direct Current (DC)*, SparkFun Electronics, learn.sparkfun.com/tutorials/alternating-current-ac-vs-direct-current-dc.

10) “4x4 Buggy Building & Tubing - 4Wheel & Off-Road Magazine.” *Four Wheeler*, 1 Feb. 2006, www.fourwheeler.com/how-to/body-chassis/131-0602-4x4-buggy-building-tubing/.

11) “Switching Regulator Fundamentals,” TI , Texas Instruments , Sept. 2016

http://www.ti.com/lit/an/snva559a/snva559a.pdf

11) “AmpFlow Wheels & Gearmotors.” Robot MarketPlace. The Robot MarketPlace, n.d. Web. 11 Apr. 2018. http://www.robotmarketplace.com/products/ampflow-motors.html

13) ”BatteryUniversity,”Battery Cofiguration. N.p., n.d. Web.

http://batteryuniversity.com/learn/article/serial_and_parallel_battery_configurations

14) DimensionEngineering," User's Guide. N.p., n.d. Web.

https://www.dimensionengineering.com/datasheets/Sabertooth2x60.pdf

15) "A sparse signal reconstruction approach for sequential equivalent time sampling" published in Instrumentation and Measurement Technology Conference Proceedings (I2MTC), 2016 IEEE International

16) Weitkamp C. *Lidar. [Electronic Resource] : Range-Resolved Optical Remote Sensing Of The Atmosphere* [e-book]. New York : Springer, c2005.; 2005. Available from: UCF Libraries Catalog, Ipswich, MA.

17) Lightware OSLRF datasheet http://www.lightware.co.za/shop2017/download/Documents/OSLRF-01%20-%20Laser%20Rangefinder%20Manual%20-%20Rev%202.pdf

18) "*ROS: Wiki Documentation*". N.p., n.d. Web. http://wiki.ros.org/.

19) *"OpenCV 2.4.8.0 documentation - Welcome to opencv documentation!"* N.p., n.d. Web. https://docs.opencv.org/2.4.8/

20) *"ISO/IEC/IEEE 29119 Software Testing -- The international standard for software testing"* N.p., n.d. Web. http://www.softwaretestingstandard.org/index.php

21) IPC. *"IPC-221B Generic Standard Generic Standard on Printed Board Design"* Web. http://www.ipc.org/TOC/IPC-2221B.pdf

9.0 Appendix

9.1 Review of Relevant Papers

9.1.1 Navigator Design Report 2011 Intelligent Ground Vehicle Competition

This was a paper discovered early on in the design process. Initial designs of the autonomous system involved stereo vision to detect obstacles in the surrounding area. The robot discussed in this report exhibited many features that seemed relevant to this initial design. The robot utilized LiDAR to detect and measure the distance to nearby obstacles. It used a three-camera setup in addition to the LiDAR system by detecting narrow objects that the LiDAR misses and collecting even more data for more detailed mapping of the surrounding area. This design utilizes the ROS navigation package. One of the main drawbacks of this design is the cost. The budget for this robot was listed at $4708. In addition, key components of the system such as LiDAR were listed as costing $0 implying that they were either sponsored or they already possessed them. The LiDAR system used by itself is over the allocated budget for the entire buggy.

9.1.2 End to End Learning for Self Driving Cars

This paper was written by Nvidia which is a leader in computer vision and machine learning. Here they attempt to create a vehicle capable of driving on roads safely. What is very interesting is the only kind of sensor used was a camera. This is done using a trained Convolutional Neural Network. It works by taking input from three separate cameras and feeding the stream into the neural network. First the images, which are initially only 66x200 pixels, are normalized and passed through three layers of convolution. These layers reduce the images to 64 separate 3x20 images that together are trained to mark the features of the road. These images are then fed into another three layers which convert them into steering controls.

The results of this experiment are very interesting. The neural network was able to learn to drive entirely from video training data. The programmers did not need to define road markers at all. It learned to distinguish the boundaries of the road entirely on its own. The conditions we will be driving under will be very different. We will not be able to rely on a defined road. Instead, we will be driving through unmarked land while avoiding any obstacles that may be present. The solution should be relatively similar though. Instead of training the buggy to follow a path, we will need it to avoid obstacles. In addition, we should be able to add a node to the input layer that feeds the neural network GPS data. This should allow the buggy to avoid obstacles with a tendency to turn towards our destination.

9.1.3 End-to-end Driving via Conditional Imitation Learning

The abstract of this paper explains the high-level concept; using deep networks trained to drive a vehicle in the sense of staying on the road and avoiding obstacles, combined with imitation learning for driving policies, ultimately creating a vehicle that can follow directions like “turn right at the next intersection”.

Practically speaking, the problem the paper is solving has to do with imitation learning. Imitation learning works under the assumption that by watching an expert perform a task, a network can be trained to imitate the expert and will then perform as well as the expert. This relationship is described in sets of observations and associated actions; that is, when the expert observes X it does Y, which is what the network seeks to replicate. However, an “implicit assumption ... is that the expert’s actions are fully explained by the observations” [2]. The problem this creates is that an imitation trained network can successfully drive a car and keep it on the road, and follow lanes- but, for example if it come to a fork in the road, it’s steering commands oscillate between the two options because it has no concept of intention. The solution in this paper was to provide an additional “command” argument that provided that intention, in addition to a clever method of training the model with a limited dataset by applying a variety of transformations to good data in order to create noisy training data, which allows the model to generalize better.

9.1.4 Autonomous Off-Road Vehicle Control Using End-to-End Learning

Though dated, this paper is an early and fairly effective example of end-to-end learning being used to drive a vehicle in unknown terrain. This paper is particularly relevant because it focused on off-road driving, with obstacles like rocks, ditches, and ponds. Furthermore, it specifically used a stereo camera design, which is one of our potential detection options. Lastly, it includes a concept of live self-adjustment, in order to adapt to new and unknown obstacles.

Overall, the utility of the paper comes from its similarity to our project – cameras for input, ultrasonic rangefinders for close range detection, a magnetometer compass – and from their method of getting useful training data, which is to record a human driving the vehicle around based on the camera feed only. Fortunately, the age of the paper may act in our favor - even though they had a $30,000 budget, in 2004 that got them a 2-core CPU with 512 MB of RAM. This means that ideally their performance should be worse than or similar to our single-board computer, giving us an idea of what we can expect. Lastly, because this paper was so early in the development of end-to-end learning as a practical solution, everything is explained at an extremely accessible level, very thoroughly.

9.2 Development Environment Setup Instructions

This section details the steps and commands used in performing various tasks in the development process.

9.2.1 Raspberry Pi 3 Operating System Install

Download the preconfigured ROS VM from Ubiquity Robotics, linked in Appendix 9.4. Save the downloaded file in a folder inside the Linux Guest operating system. Gnome Disks tool is suggested for installing the image which can be obtained via apt-get:

```
sudo apt-get install gnome-disk-utility
```

It will take a while to boot for the first time, but once done, the Raspberry Pi will broadcast a WiFi accesspoint with SSID ubiquityrobotXXXX where XXXX is part of its MAC address. The WiFi password is *robotseverywhere* and you can then login through the system via *ssh* with a password of *ubuntu:*

```
ssh ubuntu@10.42.0.1
```

Ensure that unneeded startup scripts are disabled with:

```
sudo systemctl disable magni-base
```

Another alternative to logging in to the system is hooking up an HDMI monitor, USB keyboard, and USB mouse to the Raspberry Pi and treat it as a regular computer.

Make sure the system is updated with

```
sudo apt-get upgrade

sudo apt-get update
```

9.2.2 ROS

It should be noted that ROS is already included in the Raspberry Pi 3 Operating System Image and thus, not further installation is required.

9.2.3 Arduino IDE Install

This section covers the setup of the Arduino IDE and installation of rosserial so that Arduino output appears as a node in ROS. The Arduino IDE should be installed on either the guest or home operating system (not on the Pi) for maximum performance gains.

First, install the Arduino IDE from their website, found in Appendix 9.4. Extract the files into a folder, and then open the created folder and run the *'install.sh*' file in the terminal.

9.2.4 Rosserial Arduino

The Raspberry Pi 3 image has ROS Kinetic installed by default. To install Rosserial, run the following in the Pi 3 terminal:

```
sudo apt-get install ros-indigo-rosserial-arduino

sudo apt-get install ros-indigo-rosserial
```

Next, the Arduino libraries need to be installed with the Arduino IDE. To do this, use the built-in library manager of Arduino IDE by going to Sketch->Include Library->Library Manager and search for *rosserial*. Install it using the install button, and the library can now be included in to source code via

```
#include <ros.h>
```

9.2.5 Publishing a Message to ROS via Terminal

To publish messages to a subscriber, the following is used:

```
rostopic pub <topic-name> <topic-type> [data...]
```

For example, to publish to a topic named *lidar_node* an Int16 message, type

```
rostopic pub lidar_node std_msgs/Int16 "2"
```

9.2.6 Listening to a ROS Topic via Terminal

To listen to a publishing ROS topic, the following is used:

```
rostopic echo <topic-name>
```

For example, to listen to what the *range_data* topic is publishing, type

```
rostopic echo range_data
```

9.2.7 Initializing the GPS Node

Gpsd initializes sockets that conflict with manually-run gpsd instances. As such, they must first be disabled by:

```
sudo systemctl stop gpsd.socket

sudo systemctl disable gpsd.socket
```

The port the GPS is connected to is then assigned to a gpsd socket:

```
sudo gpsd /dev/ttyAMA0 -F /var/run/gpsd.sock
```

Once done, the GPS node can then be initialize to begin publishing GPS data to ROS:

```
rosrun gpsd_client gpsd_client _host:=localhost
```

9.2.8 Initializing the rosserial Nodes

By default, rosserial creates a node with the name *serial_node*. This poses a problem as no new nodes can be made with the same name. Two rosserial nodes need to be created, one for each microcontroller connected to the buggy. A simple workaround to this is creating an XML launch file. An example follows:

```
1  <launch>
2      <node pkg="rosserial_python" type="serial_node.py"
3      name="LidarServo" args="/dev/ttyUSB0">
4          <param name="baud" value="57600"/>
5      </node>
6      <node pkg="rosserial_python" type="serial_node.py"
7      name="ServoLidar" args="/dev/ttyUSB1">
8          <param name="baud" value="57600"/>
9      </node>
10 </launch>
11
```

Once created, the rosserial nodes can be initialized by simply typing in the terminal:

```
roslaunch <XML file>
```

9.3 Development Environment Resources

1. VMware Workstation Player. https://www.vmware.com/
2. Linux Mint. https://www.linuxmint.com/
3. Ubiquity Robotics Operating System Image.

   https://downloads.ubiquityrobotics.com/
4. Arduino IDE. https://www.arduino.cc/en/Main/Software

9.4 Miscellaneous Specifications

This section details the specifications of the chips used in the creation of the circuitry of the buggy. Table 33 shows the specification for the microcontroller used and Figure 58 and 59 shows the pinout of the two microcontrollers used in the buggy

Features	ATmega328/P
Pin Count	28/32
Flash (Bytes)	32K
SRAM (Bytes)	2K
EEPROM (Bytes)	1K
General Purpose I/O Lines	23
SPI	2
TWI (I^2C)	1
USART	1
ADC	10-bit 15kSPS
ADC Channels	8
8-bit Timer/Counters	2
16-bit Timer/Counters	1

Table 33. ATMega328P specifications.

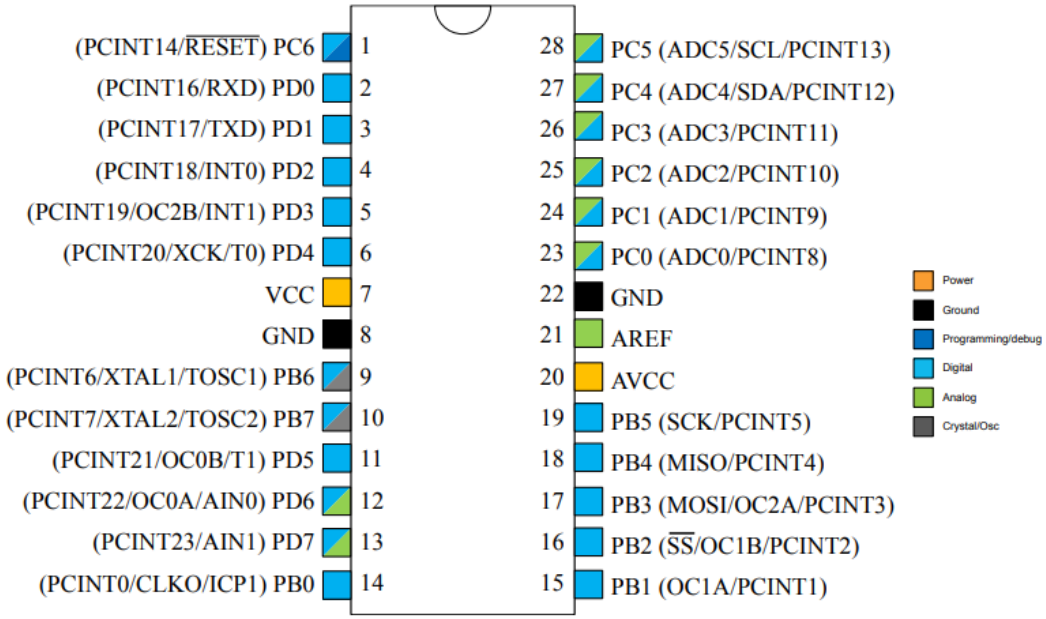


Figure 58. 28-pin PDIP ATMega328P Pinout.

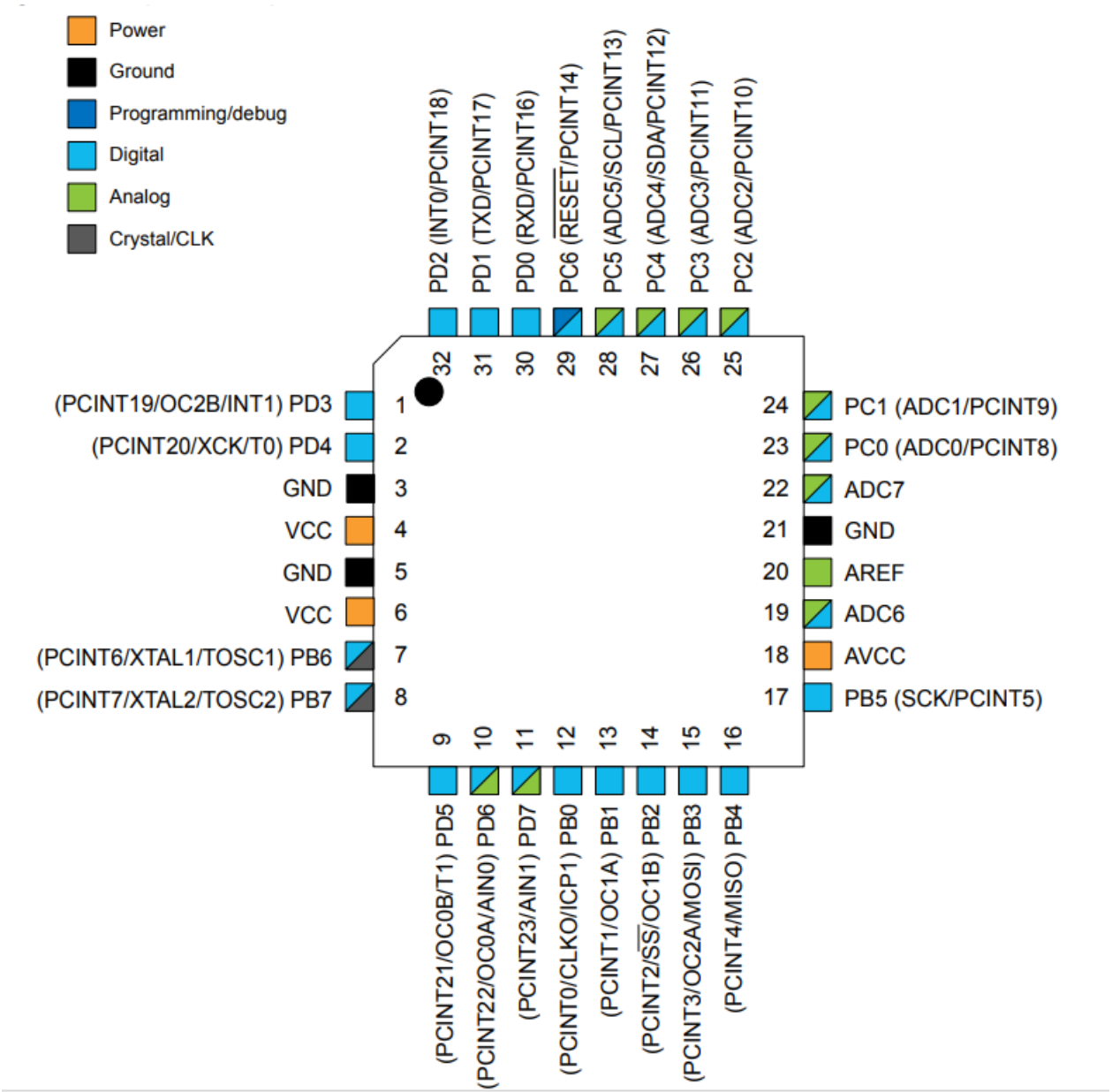
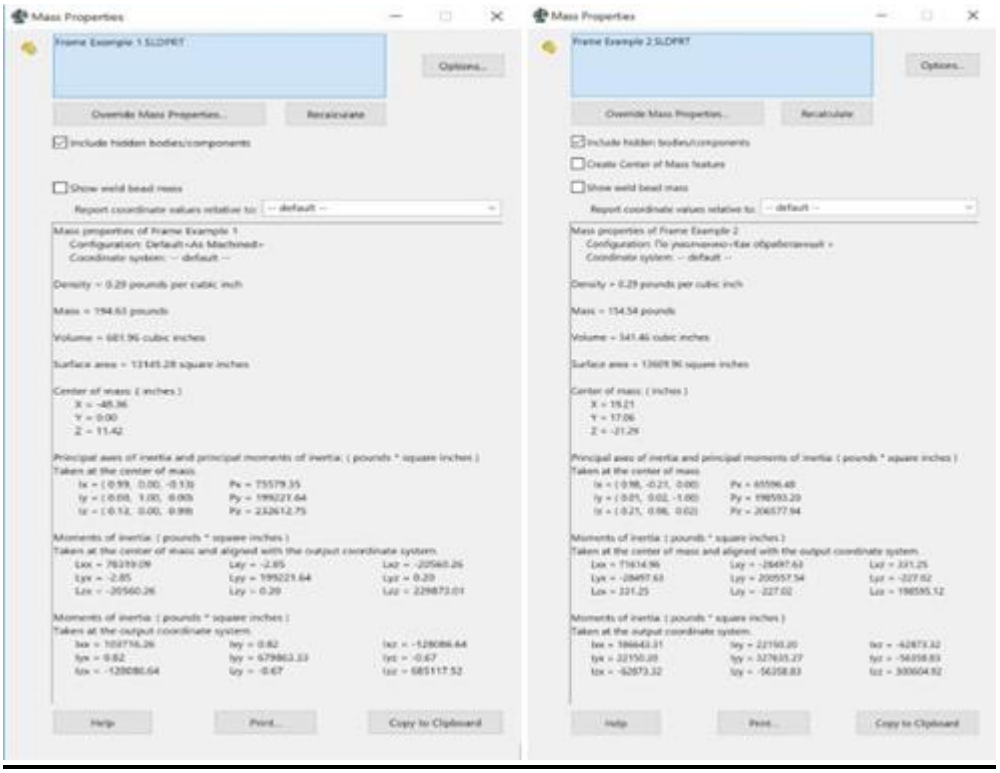


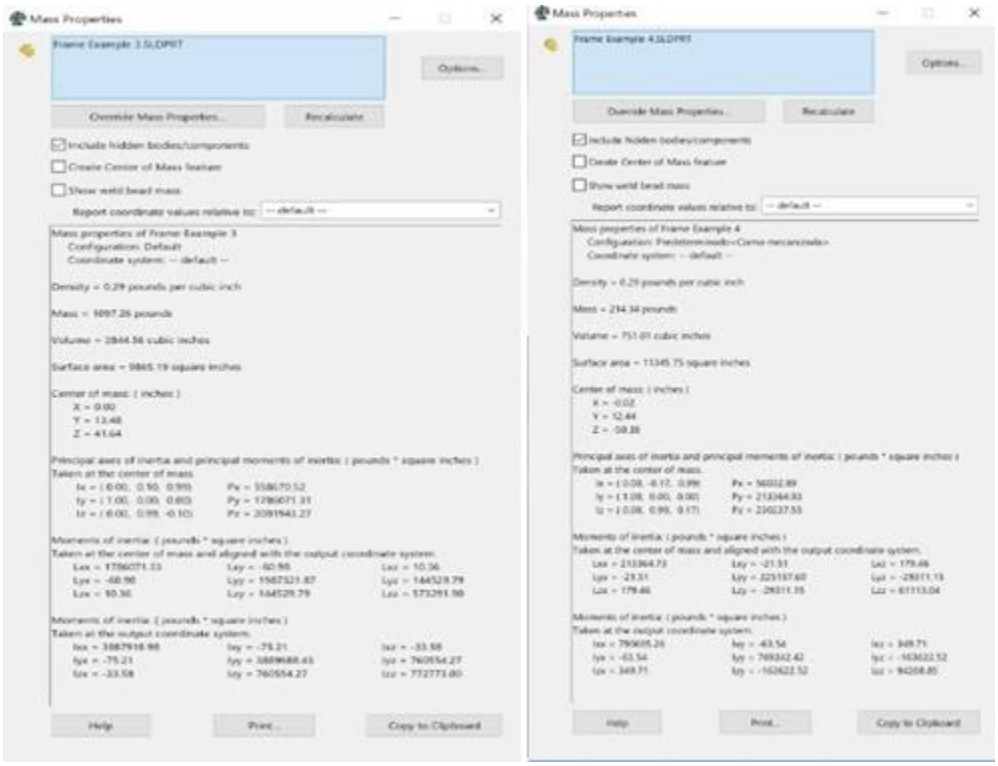
Figure 59. 32-pin TQFP ATMega328P Pinout.

9.5 SolidWorks CAD Iteration Information

This section details the iteration results from SolidWorks CAD.



(a) Iteration 1 Mass Properties (b) Iteration 3 Mass Properties



(a) Iteration 3 Mass Properties

(b) Iteration 4 Mass Properties

(a) Iteration 6 Mass Properties

(b) Iteration 7 Mass Properties

9.6 Ansys Analysis Data

This section provides Ansys test results (table 33 to 46) in simulating the different factors and materials for the construction of the buggy frame.

1. Factor of Safety

Time [s]	Minimum	Maximum
1	1.7666	15

Table 34 - Static Structural Safety Factor

Object Name	Stress Tool
State	Solved
Definition	
Theory	Max Equivalent Stress
Stress Limit Type	Tensile Yield Per Material

Table 35 - Static Structural Stress Safety Tools

Object Name	Safety Factor
State	Solved
Scope	
Scoping Method	Geometry Selection
Geometry	All Bodies
Definition	
Type	Safety Factor
By	Time
Display Time	Last
Calculate Time History	Yes
Identifier	
Suppressed	No
Integration Point Results	
Display Option	Averaged
Average Across Bodies	No
Results	
Minimum	1.7666
Minimum Occurs On	chassis Default<As Machined>
Information	
Time	1. s
Load Step	1
Substep	1
Iteration Number	1

Table 36 - Static Structural Stress Tool Results

Object Name	Stress Tool 2
State	Solved
Definition	
Theory	Max Shear Stress
Factor	0.5
Stress Limit Type	Tensile Yield Per Material

Table 37 - Static Structural Stress Safety Tools

Object Name	Safety Factor
State	Solved
Scope	
Scoping Method	Geometry Selection
Geometry	All Bodies
Definition	
Type	Safety Factor
By	Time
Display Time	Last
Calculate Time History	Yes
Identifier	
Suppressed	No
Integration Point Results	
Display Option	Averaged
Average Across Bodies	No
Results	
Minimum	1.5343
Minimum Occurs On	chassis Default<As Machined>
Information	
Time	1. s
Load Step	1
Substep	1
Iteration Number	1

Table 38 - Static Structural Stress Tool 2 Results

Time [s]	Minimum	Maximum
1	1.5343	15

Table 39 - Static Structural Stress Tool 2 Safety Factor

2. Material Data

Density	0.2836 lbm inˆ-3
Coefficient of Thermal Expansion	6.6667e-006 Fˆ-1
Specific Heat	0.10366 BTU lbmˆ-1 Fˆ-1
Thermal Conductivity	8.0917e-004 BTU sˆ-1 inˆ-1 Fˆ-1
Resistivity	8.5235 ohm cmil inˆ-1

Table 40 - Structural Steel Constants

Compressive Ultimate Strength psi
0

Table 41 - Structural Steel Compressive Ultimate Strength

Compressive Yield Strength psi
36259

Table 42 - Structural Steel Compressive Yield Strength

Tensile Yield Strength psi
36259

Table 43 - Structural Steel Tensile Yield Strength

Tensile Ultimate Strength psi
66717

Table 44 - Structural Steel Tensile Ultimate Strength

Zero-Thermal-Strain Reference Temperature F
71.6

Table 45 - Structural Steel Isotropic Secant Coefficient of Thermal Expansion

Alternating Stress psi	Cycles	Mean Stress psi
5.80E+05	10	0
4.10E+05	20	0
2.75E+05	50	0
2.05E+05	100	0
1.55E+05	200	0
63962	2000	0
38000	10000	0
31038	20000	0
20015	1.00E+05	0
16534	2.00E+05	0
12502	1.00E+06	0

Table 46 - Structural Steel Alternating Stress Mean Stress

Relative Permeability
10000

Table 47 - Structural Steel Isotropic Relative Permeability

9.7 Requests for Image Use

This section shows the emails sent to various companies to request the use of images found on their website.

Status: Request Sent
Subject: Request for permissions to use graphs, images, figures, and data

To Whom this may concern:

I am currently an electrical engineering student at the University of Central Florida. I am working on a senior design project, and I was wondering if I could use images, graphs, tables and information presented in your website http://ti.com and http://www.ti.com/tools-software/design-center/webench-power-designer.html for my senior design documentation.
I will cite the source/owner of the material.

Thank you for your time.
Robinson Charles
imobile@knights.ucf.edu

Status: Request Sent
Subject: Request for permissions to use graphs, images, figures, and data

To Whom this may concern:

I am currently an electrical engineering student at the University of Central Florida. I am working on a senior design project, and I was wondering if I could use images, graphs, tables and information presented in your website www.microchip.com for my senior design documentation.

I will cite the source/owner of the material.

Thank you for your time.
Robinson Charles
imobile@knights.ucf.edu

Figure 58a. Emails Sent Requesting Image Use

Status: Request Sent
Subject: Request for permissions to use graphs, images, figures, and data

To Whom this may concern:

I am currently an electrical engineering student at the University of Central Florida. I am working on a senior design project, and I was wondering if I could use images, graphs, tables and information presented in your websitehttps://www.renogy.com for my senior design documentation.

I will cite the source/owner of the material.

Thank you for your time.
Robinson Charles
imobile@knights.ucf.edu

Status: Request Sent
Subject: Request for permissions to use graphs, images, figures, and data

To Whom this may concern:

I am currently an electrical engineering student at the University of Central Florida. I am working on a senior design project, and I was wondering if I could use images, graphs, tables and information presented in your websitehttps://www.dimensionengineering.com for my senior design documentation.

I will cite the source/owner of the material.

Thank you for your time.
Robinson Charles
imobile@knights.ucf.edu

Figure 58b. Emails Sent Requesting Image Use

From: Renishaw Web Site [mailto:email@renishaw.net]
Sent: Saturday, July 21, 2018 3:13 PM
To: USA
Cc: CRM.leads
Subject: Renishaw - Pricing query (General enquiries) (routed to default country contact) [US]

Renishaw Logo

Contact us - Pricing query - General enquiries

Any further questions or requests for assistance please contact usa@renishaw.com or call your local Renishaw office - visit www.renishaw.com/worldwidecontacts to find the number.

Your details

Name	Jose Rosales
Email address	j053r64@knights.ucf.edu
Telephone	
Company name	University of Central Florida
Address line 1	4000 Central Florida Blvd.
Address line 2	
City / town	Orlando
County / state	FL
Country	United States
Post / zip code	32816

Enquiry details

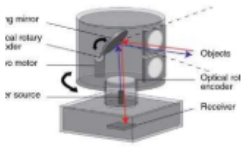
Product	Optical encoders and LiDAR scanning
Product line	General enquiries
Contact method	Email
DPA	No
Enquiry	I'm a university student writing a paper on a project which utilizes Lidar scanning, and I would like permission to use Figure 2 provided on the page describing the basics of Optical Encoders and Lidar scanning as it is an ideal diagram describing the structural concepts behind a Lidar module.

From: Charlie Falco <charlie.falco@renishaw.com>
Sent: Monday, July 23, 2018 1:16 PM
To: Jose Rosales
Cc: Robert Koch
Subject: RE: Renishaw - Pricing query (General enquiries) (routed to default country contact) [US]

Hello Jose,
What document are you referring to? Can you forward me a copy, or a link to the page?

Best Regards,
Charlie Falco
Technical Support Engineer
Renishaw Inc

Phone +1 (631) 321-0418 Cell +1 (847) 340-3120
Email charlie.falco@renishaw.com Website www.renishaw.com



Show all 1 attachments (407 KB) Download Save to OneDrive - Knights - University of Central Florida

Please see the attached jpg.

Best Regards,
Charlie

From: Charlie Falco
Sent: Monday, July 23, 2018 1:47 PM
To: Jose Rosales <J053R64@Knights.ucf.edu>
Subject: RE: Renishaw - Pricing query (General enquiries) (routed to default country contact) [US]

OK, thank you for the link. I will check on this and get back to you.

Best Regards,
Charlie

From: Jose Rosales <J053R64@Knights.ucf.edu>
Sent: Monday, July 23, 2018 1:23 PM
To: Charlie Falco <charlie.falco@renishaw.com>
Subject: Re: Renishaw - Pricing query (General enquiries) (routed to default country contact) [US]

Hello Mr. Falco,

This is a link to the page I was referring to: http://www.renishaw.com/en/optical-encoders-and-lidar-scanning--39244
The figure I was asking to utilize was #2, about halfway down the page.

Sincerely,
Jose Rosales

Figure 58c. Emails Sent requesting Image Use

You received a new message from your online store's contact form.

Name:
Jose Rosales

Email:
j053r64@knights.ucf.edu

Phone:

Body:
I'm a university student writing a paper on a project which utilizes Lidar scanning, and I would like permission to use the figures provided in the OSLRF-01 Product Manual, as they are phenomenal aides in describing the basics of how a Lidar scanner functions. If you could please give me a response at your earliest convenience, it would be greatly appreciated.

lightware optoelectronics

NN Nadia Nilsen <NadiaNilsen@lightware.co.za>
Mon 7/23, 4:58 AM
Jose Rosales

Good day Jose,

Please feel free to do so. They were meant as educational products and we are glad to hear they are helping.

***PLEASS NOTE: LIGHTWARE HAS A NEW ADDRESS ***

Kind regards,

Nadia Nilsen
Financial Director
LightWare Optoelectronics (Pty) Ltd
E-mail: nadianilsen@lightware.co.za
Tel: +27 (0)12 942-0408
Mobile: +27 (0)73-373-8828
www.lightware.co.za

On 23 Jul 2018, at 7:28 AM, Info :: LightWare <info@lightware.co.za> wrote:

Begin forwarded message:

From: "LightWare Optoelectronics (Shopify)" <mailer@shopify.com>
Date: 21 July 2018 at 22:30:28 SAST
To: info@lightware.co.za
Subject: New customer message on July 21, 2018 at 10:30 PM
Reply-To: j053r64@knights.ucf.edu

Figure 58d. Emails sent Requesting Image Use