6-6-2014

Proteus: Mini underwater remotely operated vehicle

Jorge Guerra
Santa Clara University

Robert Heinevetter
Santa Clara University

Tristan Morris
Santa Clara University

Killian Poore
Santa Clara University

Alexandra Waschura
Santa Clara University

SANTA CLARA UNIVERSITY

Department of Mechanical Engineering

Date: June 6, 2014

I HEREBY RECOMMEND THAT THE THESIS PREPARED
UNDER MY SUPERVISION BY

Jorge Guerra, Robert Heinevetter, Tristan Morris, Killian Poore and Alexandra Waschura

ENTITLED

PROTEUS: MINI UNDERWATER REMOTELY OPERATED VEHICLE

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

BACHELOR OF SCIENCE
IN
MECHANICAL ENGINEERING

Thesis Advisor

Department Chairman

Proteus: Mini Underwater Remotely Operated Vehicle

By

Jorge Guerra, Robert Heinevetter, Tristan Morris, Killian Poore and Alexandra Waschura

THESIS

Submitted in Partial Fulfillment of the Requirements for the
Bachelor of Science Degree in
Mechanical Engineering in the School of Engineering
Santa Clara University, 2014

Santa Clara, California

PROTEUS: MINI UNDERWATER REMOTELY OPERATED VEHICLE

Jorge Guerra, Robert Heinevetter, Tristan Morris, Killian Poore and Alexandra Waschura

Department of Mechanical Engineering
Santa Clara University
Santa Clara, California
2014

ABSTRACT

Marine ecosystems contain life, minerals, information, etc, that can help the planet, however, only 5% of them are explored. This is mainly because existing Underwater Remotely Operated Vehicles (ROVs) are expensive and require a lot of work and time to use. Team Proteus designed a low cost, easy to use, portable, safe, and reliable ROV capable of being used for scientific research, while being operated and maintained by students. In this paper we explain the necessity behind this project, how it compares to similar projects and the design decisions made in developing the ROV, to include the options and trade-offs considered. We also present project budgets, the final design, and results of our field tests.

ACKNOWLEDGMENTS

Thanks to Dr. Christopher Kitts, Thomas Adamek, Mike Vlahos, Matt Chin, Dr. Rich Schweikert, Dr. Jim Moore and Dr. Winnie Kortemeyer for their help on the project as well as the Robotics Systems Lab and the School of Engineering for financial support. We would also like to thank our friends and families for their support during this project

Table of Contents

List of Figures

List of Tables

Chapter 1 - Introduction

This section discusses the background and motivation for this project as well as a review of field literature and current systems.

1.1 - Background/Motivation

With around 44% of the world's population living within 150 km of a coastline, it is evident that the marine environment plays a big role in human lives (Humans Settlements on the Coast). The ocean provides many resources to humans including oil, minerals such as salt, sand, gravel, and even nickel, iron, and cobalt can be found. About 200 billion pounds of fish and shellfish are caught every year for human consumption (Ocean Resources). The ocean also provides a means of transportation, and a form of recreation. However our oceans have suffered from industrial run-offs, oil spills, over-fishing, and climate change. Give the importance of our oceans, the first motivation behind this project was the necessity to learn more about our oceans so we can learn to use these resources sustainably, efficiently, and intelligently because, if not, we will have to deal with the consequences.

Oceans cover 71% of the planet and only 5% is explored (Oceans). Scientists have researched marine environments for decades, and marine technology has given them novel ways to explore this environment. Robotic systems have augmented scientist's tools for research. Scientists used to manually collect samples for later testing; they also had to explore the marine environment by diving and recording what they found. They were usually constrained mostly by human capacity, restricted by the inability to research and collect multiple data sets at once, the amount of time one can spend underwater, the depth that could be reached and/or the tiring nature of these missions.

Conventional exploration methods are being replaced by robotic approaches, as they provide a more efficient and powerful solution to ocean exploration. These robotic systems have already given insight into previously unexplored areas. Marine robotic systems can range from tethered Remotely Operated Vehicles (ROVs), usually used in short missions (hours, days), to Autonomous Underwater Vehicles (AUVs), usually used for longer duration missions (weeks). Robotic systems have a wide range of sensing capabilities useful for scientific research, including temperature, depth, conductivity, pH, chemical makeup, light, and location. Figure 1 shows Monterey Bay Aquarium Research Institute's (MBARI) ROV and AUV systems.

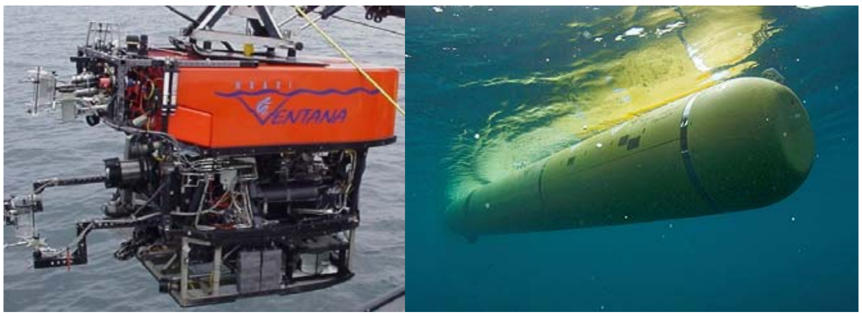


Figure 1: ROV Ventana (left) and AUV Dorado (right) (*Photos: MBARI*)

MBARI's ROV Ventana weighs 10,500 lbs with dimensions of 6 feet by 12 feet by 7 feet, requiring a large crew as well as specialized deployment systems (Vessels and Vehicles). These systems, while extremely capable, are very expensive and difficult to use.

Underwater ROVs, which can work at depths beyond the reach of scuba divers, give us the opportunity to explore and fill the "information gap" between near shore and offshore habitats. This is critical for developing comprehensive management strategies for the ocean's resources. Small ROVs are the future for exploring oceans and lakes. Being more cost effective and requiring less people, these ROVs will allow for more opportunities to research our oceans.

The motivation behind the development of Mini ROV "Proteus", as detailed in this thesis, was to develop a low cost, safe, and portable ROV capable of collecting data from its environment and conducting scientific missions. The ROV is to be operated by students and maintained by the Santa Clara University (SCU) Robotic Systems Lab (RSL).

1.2 - Reviews of Field Literature

The ROV we designed was not the first of its kind so we reflected on past work involving underwater robots to help with our design. One of the requirements for our design was to be relatively inexpensive. The sources we looked at had to do with designing similarly inexpensive ROVs. The next three have to do with the diverse ways people have been able to use underwater ROVs, ranging from scientific research to the recovery of people.

1.2.1 - Design of an Inexpensive Waterproof Housing

The article, *Design of an Inexpensive Waterproof Housing* by four students at Lake Superior State University, contains a detailed description of one of the most difficult tasks with underwater robotics, which is water proofing the electronics so they can be used even at the greatest depths of the ocean. There are many ways electronics can be waterproofed. This article deals with two possibilities: epoxy resin dunking (permanently sealing electronics in epoxy) and bottling. This article discusses all the considerations that need to be taken into account when bottling electronics. Some of these considerations are chemical resistance, abuse when handling the robot and, of course, making the bottle able to withstand high pressures, their system was tested to a depth of 300 feet. We bottled our electronics since there was extensive testing that had been done on these systems (Harrington).

1.2.2 - Design and Manufacture of a Low Cost Underwater Remote Operated Vehicle (ROV)

In 2004, David Buecher made a low cost remotely operated vehicle and his thesis, *Design and Manufacture of a Low Cost Underwater Remote Operated Vehicle(ROV),* explains how he did it. This is relevant to this project because Buecher's goals were to make this robot out of commonly found items and for less than $1500. The goal for Proteus was to be smaller and less expensive than Triton, an existing ROV the RSL uses. This system is described in more detail later in Section 1.3.1. Buecher highlighted how he was able to find most of the pieces he needed for the robot at places like Lowes and

Home Depot. Anything he could not find inexpensively, he made himself. For example, the tether required to communicate with the ROV that he wanted to purchase was too expensive for his budget so he instead made a neutrally buoyant tether himself (Buecher).

The projects are different in that our budget was not as small as Buecher's. His ROV consisted of motor controllers, an AVR mini board, and a camera. Top-side, he had a computer and Logitech joystick to control the robot via tether, and a VCR to record images from the camera. This thesis helped show how to weigh cost versus quality and helped us maintain our budget.

1.2.3 - Marine Heterogeneous Multi-Robot Systems at the Great Eastern Japan Tsunami

The article, *Marine Heterogeneous Multi-Robot Systems at the Great Eastern Japan Tsunami Recovery* by Robin R. Murphy, describes the response and recovery efforts by a team of heterogeneous unmanned vehicles at the 2011 Great Eastern Japan Earthquake. Three different remotely operated vehicles (ROVs) were used in the effort to recover victims and clear ports. ROVs were chosen over autonomous underwater vehicles (AUVs) for the following reason: ROVs are tethered, so if communication is lost or an ROV is grounded, it can be retrieved using the umbilical. AUVs also usually use side scan sonars, which have a lower resolution than the imaging found on ROVs (Murphy).

The ROVs required specific pieces of technology to complete these missions effectively. The ROVs all had video capabilities, as well as sonar imaging for when the water was too turbulent to see. Three different systems were used for resilience; one system could succeed where the other failed. Each ROV's position could be found using an external sonar, or simply by tether length. All the systems chosen were small, portable, and could fit in a personal truck. This article also gives good insight into the uncertainty of field deployments and the need for a flexible system. Some launch locations were large and capable of deploying several ROVs at once, while others had physical limitations and only one ROV could be deployed. Some systems also could not run in close proximity because their sensors would interfere with each other, as well as there being a danger of tether entanglement.

1.2.4 - Assisting Micro-ROV Operators During Surveys in Fragile Environments

The article, *"Assisting Micro-ROV Operators During Surveys in Fragile Environments"* by David Scaradozzi, Giuseppe Conte, and Laura Sorbi, is about how a team of ROV engineers came up with a way for inexperienced ROV operators to pilot an ROV in a highly sensitive area without any expert training. What they did was essentially nest a Micro-ROV inside of a larger ROV, which was brought down to a certain depth. The larger ROV would be controlled automatically to navigate a certain path at a specific depth, while the Micro-ROV was allowed to roam free at the depths below. This is a good improvement because not only does it allow for a smaller ROV to be less intrusive, but it also takes some human element out of the process.

Another important part of the system is what they call the Assisted Guidance System. This system makes the operator's job even easier. The Assisted Guidance System is implemented on the Micro-ROV, and essentially creates boundaries within which the Micro-ROV has to stay. When the ROV starts to drift out of these boundaries the joystick resistance starts to increase, which encourages the operator to return it to the center. In sensitive areas where ecosystems need to be maintained, this level of precision operation is crucial (Scaradozzi).

This article showed potential uses for our ROV. We have developed a small ROV, which could possibly be the "pet" for another big ROV down the road, however our system is currently limited to fairly shallow depths (less than 500 feet). The article does show that our Mini-ROV could be good for more precision work and tight spaces, such as caves.

1.2.5 - ROVs Continue to Develop Capacity for Deepwater Operations

Martin Wareham wrote the article *ROVs Continue to Develop Capacity for Deepwater Operations*, where he discusses the many uses of an underwater ROV as well as how they will continue to improve over the next few decades. There is a large variety of ROVs due to the wide range of different underwater tasks that they can perform. Smaller ROVs are now capable of doing things that only large ROVs could do before, while larger ROVs are pushing the boundaries as to what was thought possible. These huge improvements are due to the ongoing developments in robotic technology. Some of these developments include more capable sensing products, lighter/stronger materials, and more advanced control systems. These are just a few of the current improvements, and these advancements will continue to grow along with the rise of offshore exploration and subsea field development (Wareham).

1.3 - Review of Existing Systems

Our ROV was designed to be smaller than, lighter than, less expensive than, and easier to use than the underwater remotely operated vehicles below. Some are Santa Clara University projects while others are not related to the school. It is important that we study these ROVs to find requirements we need to consider as well as to learn from any mistakes made during production of these ROVs.

1.3.1 - Triton ROV (Santa Clara University and Deep Ocean Engineering)

Santa Clara University's Robotic Systems Lab has developed and worked with several ROVs; Triton is one of them. Triton is a heavy duty professional class ROV developed by SCU students with assistance from engineers at Deep Ocean Engineering. It has been used for several research missions per year over the last 15 years , mostly with geologists from the University of Nevada – Reno and the US Geological Survey.

Comment [anw1]: Citation

Figure 2: Triton ROV during deployment

Triton is about 36 in x 28 in x 28 in and weighs approximately 250 lbs. The system runs on a 120 AC volt supply stepped to 240 AC when sent down the tether. It has a camera, lights, pressure sensor, and magnetometer. Due to its size, it takes about 5 people to deploy Triton, as well as a lot of equipment. Triton is constrained by its tether length to a depth of 500 ft. The system cost approximately $75,000 to develop (Weast).

1.3.2 - PVC ROV (Santa Clara University)

PVC ROV began as a senior capstone project at SCU and has since been worked on by other students. This system is meant to be a inexpensive, portable, reliable multi-robot test bed used by students to test cluster control techniques.

Figure 3: PVC ROV on test bench *(Photo: Killian Poore)*

The ROV is 12 in x 12 in x 12 in and weighs about 12 lbs. This system is powered by batteries and can run for about 2 hours. The ROV is made from PVC. The system has no camera or pressure sensor, but has a magnetometer. The system has a 50 ft negatively buoyant tether and has been tested in Stevens Creek Reservoir, Del Valle, and Lake Tahoe. The cost per ROV is about $1,200 (Vlahos).

1.3.3 - Seabotix vLBV300

This is a rugged ROV with a vectored thruster configuration making it very agile. It is powered by 120 – 240 volts AC. The frame is made from high density polyethylene and uses foam for flotation.

Figure 4: Seabotix vLBV300 (*Photo: Seabotix*)

The ROV is 24.6 in x 15.4 in x 15.4 in and weighs 40 lbs. It has a mounted camera, lights, magnetometer, and a pressure sensor. It comes with a 820 foot neutrally buoyant tether, and is rated for 1,000 ft. The entire system costs $88,000 (Seabotix).

1.3.4 - VideoRay Explorer X3

This ROV is designed as a system for users on a budget. It costs $14,500 and lacks some of the capabilities of larger ROVs, but it is good for inspection and recreational use. Its dimensions are 12 in x 9 in x 8.5 in, and it weighs 8 lbs, making it very portable. The ROV can be seen in Figure 5 below.

Figure 5: VideoRay Explorer X3 (*Photo: VideoRay*)

It has an integrated camera, halogen lights, and heading, and depth sensors. The included neutrally buoyant tether has a length of 130 but the ROV is rated to 250 ft (VideoRay).

1.4 - Statement of Project Goals and Objectives

The goal of this project was to design and build a low cost, easy to use, portable, safe, and reliable ROV capable of being used for scientific research, while being run by students. We deployed ROV systems from the RSL in order to get a feel for how ROVs work. An in depth survey was conducted with potential users, experienced users, and industry experts in order to understand what was required in an ROV and what to keep in mind when developing one. We developed several sketches of possible designs for our ROV, and built several prototypes; getting feedback from our customer on each design. We tested resulting components of our system when appropriate before integrating the full system, ensuring a successful build.

The system was used for real research missions in Lake Tahoe at the end of the year, validating the success of the ROV. Proteus reached a depth of 75 feet while sending depth, temperature and heading readings as well as the live feed from the camera to the topside console. The maximum speed of Proteus was found to be 1 foot per second. The ROV will continue be used by the RSL to educate students, further research in control techniques, and aid scientists in understanding marine environments.

Chapter 2 - System Level

This chapter gives a system level description of our project including how it is used, how our requirements were decided and how the team works. This section also includes discussions about the challenges we encountered, our budget, our timeline and the design process.

2.1 - Systems Level Overview

Communication is constant from Proteus to the operators through a tether that connects the topside console to the robot. There are three options for the topside console during scientific missions: joystick, computer, tablet. With this topside console, the user can drive the ROV and observe the live feed coming from the camera mounted on Proteus, as well as the heading, temperature of the water and depth of the ROV. Data recovered during a mission can be uploaded to “The Cloud”. This information, as well as Proteus, can be used in the future by students of the university as well as scientists and faculty members for scientific missions.

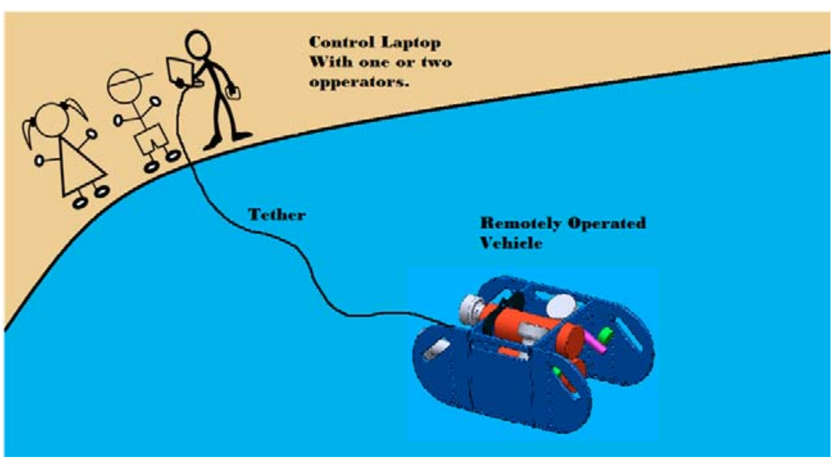


Figure 6: Shows the ecosystem for the robot

Not only can Proteus be used in the future, but it also offers an auxiliary port and mounting holes if anyone would like to attach supplementary equipment, like a manipulator, to expand on the capabilities of the ROV. Proteus can be deployed from a boat or from on land, allowing for ease of use.

2.1.1 Component Block Diagram

The component block diagram for the ROV can be seen in Figure 7. A main electronics bottle holds a microcontroller, motor drivers, communication protocol converter and video feed amplifier. The sensors, lights, and camera are controlled by the microcontroller. The motor drivers control the thrusters. The battery pack is mounted in a separate bottle and powers the whole system. The microcontroller receives commands from the topside station to control components and drive thrusters, and it collects sensor data and sends it up the line. This communication line and camera feed make up the tether connecting the ROV to the topside console.

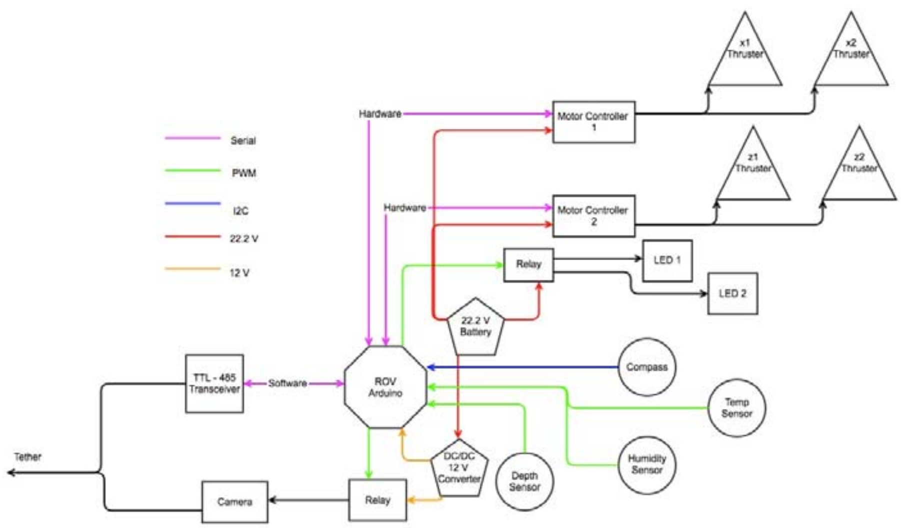


Figure 7: Component block diagram of ROV.

The ROV has three intended options for topside control. The first mode is a pilot console where an external display is used for the video feed, data is displayed on a LCD screen, and the user inputs drive commands using a joystick.

The second option uses a laptop interface to control the ROV. There is still the option for manual drive with a joystick, or it can be autonomously driven with a controller designed by the user.

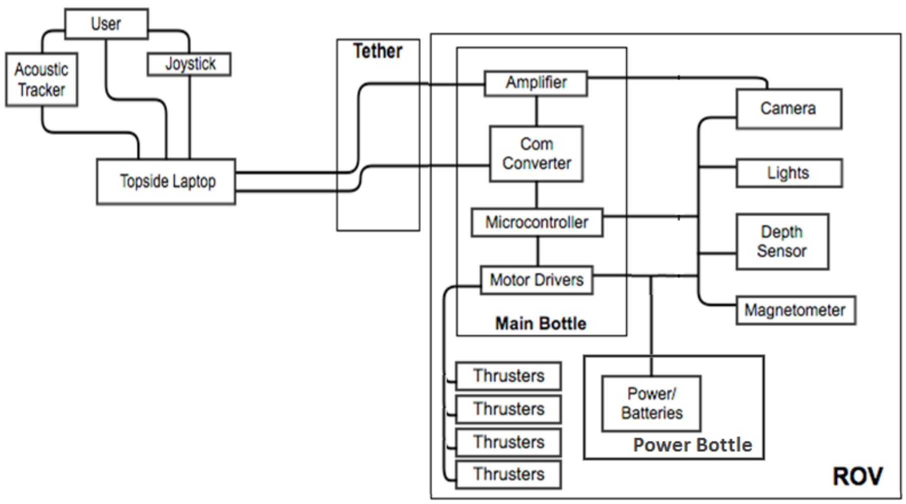


Figure 8: Component block diagram with second mode of topside control, all plugged into a laptop to display data.

We also teamed up with two computer engineering capstone groups, who tried developed a third interface to control the ROV with a tablet. This system would have video, data overlay, and control of the full functionality of the ROV. This interface was not completed due to development problems within that team.

2.2 - Customer Needs

Our initial and primary customer was Professor Kitts and the Robotics System Lab. The project was funded by Professor Kitts and the Robotics Systems Laboratory (RSL) and was to be used for education as well as a backup to the current Triton ROV. The RSL already had ROV's, however, not like the one we designed. The current systems were either too expensive and resource intensive, too risky to operate, or too cheap with very limited functionality. This was the gap that Proteus was intended to bridge.

We conducted a customer needs analysis and engaged with a variety of customers to find key features we needed to focus on. We talked to industry experts, potential users (scientists, graduate student), experienced users, and key customers. Table 1 shows our customers and their roles in a more detailed manner.

Table 1: Interviewees

Interviewee	Description	Customer type
Dr. Christopher Kitts	Head of the Robotic Systems Lab (RSL) at SCU	Key customer, stakeholder
Thomas Adamek	Head of marine operations, RSL	Key customer
Bill Kirkwood	Engineer at Monterey Bay Aquarium Research Institute (MBARI)	Industry expert
AJ Cecchettini	Engineer at Deep Ocean Engineering (DOE)	Industry expert
Rich Schweickert	Geologist at University of Nevada - Reno	Potential user
Geoff Wheat	Scientist at MBARI	Potential user
Mike Vlahos	Graduate student/RSL associate	Experienced user

The feedback gained from conducting these interviews was analyzed and grouped by themes. These themes were: attachments, performance, operation, user interface, portability, purpose, simplicity, safety, robustness, and cost. A spreadsheet with the categorized feedback can be seen in Appendix 1. Table 2 shows some of the more important feedback we received, separated by customer type.

The primary needs our customers had related to cost and portability. The ROV needed to be smaller than Triton and weigh less. The system was to be deployable by 1 - 3 people. The RSL wanted a system that could be deployed using a single boat and car, limiting the amount of equipment used for deployment. Our customer also wanted the ability to fly the ROV out to different universities, so we had to design it to be small and light so that it would be easy to ship.

The ROV also needed to be relatively inexpensive. This meant that it would be inexpensive compared to its counterparts offered in the current market, and in particular, to the current ROV at SCU, Triton.

Another big need the RSL has was modularity and versatility of the system. The ability to change the system for certain missions, or as technology improves, is invaluable. Therefore, little effort is required to enable the system to be capable of accepting auxiliary features and to easily swap out parts. This allows future capstone projects to build on our design.

Table 2: Analyzed feedback

Key customers	Experienced users	Potential users	Industry experts
Deploy from shore	Electrically safe system	Good camera	Thrust lines through center of gravity
Operate for 1-8 hrs / multiple deployments	Tether management system	Data overlay on video feed	Bound mission, establish what to solve
Safe, low voltage	Good camera	500-600 ft depth rating	Tether management for driving dynamics
1-3 people deploying	Winch hook	Manipulator	Minimal number of marine plugs
$10-15 k for parts (no labor)	Handles for ergonomics	Positioning data (x,y,z coordinates)	Extra line or two in tether for future use
Transport in back of car	Well documented	Laser scaling system	Simple, form follows function
500 ft depth rating	Easy to maintain	Perform well with required payload	Design for robustness, will save in the long run
Quick set up	Split video lines / automatic recording		
Small and light	Variable ballasting		
Serve as student development project	Extra lines in tether		

The system also had to be easy to use and work on. The goal was to have students (graduate and undergraduate) run the system, maintain it, and troubleshoot it as needed. This was needed in order to save the RSL time and money and to give students an opportunity to work on a real engineering project.

2.3 - System Requirements

This customer needs exercise provided us with a refined list of needs, which translate into refined system requirements.

These requirements answer most of the needs expressed by our customers. They definitely answer the needs we as a team deemed the most important after analyzing the feedback we got. The Product Design Specification table can be seen in Appendix 2. The system requirements, baseline and aims, are as follows:

Table 3: System requirements

Description	Baseline	Aim
Cost of parts	< $15,000	
Dimensions	~ 62 linear inches (L+W+H)	
Mass	< 75 lbs	< 50 lbs
Deployment personnel	3 people	2 people
Portability	Entire system fit in a personal vehicle	
Voltage of system	< 48 Volts	
Depth rating	~ 500 ft	~ 1000 ft
Battery life	> 1 hour	> 3 hours
Buoyancy of ROV	Slightly positive buoyancy	
Payload	~ 5 lbs	> 10 lbs
Auxiliary port	1 auxiliary port connected to microcontroller	Multiple lines, with access to power
Camera	Live feed	Zoom, focus control
Sensing	Attitude, depth, temperature	Conductivity, humidity (electronics),
Set up time	~ 15 min	
Ergonomic	Handles around structure	

2.4 - Functional Analysis

Our ROV is broken down into the following main subsystems:

- Flotation
- Frame
- Waterproof housing
- Processing
- Communication
- Propulsion
- Power
- Camera & lights
- Sensors

The frame is the structural skeleton of the ROV and can be seen in blue in Figure 9. The flotation is a material mounted to the frame, not in the figure, that will keep the ROV slightly positively buoyant to make driving the ROV easier. The waterproof housings, in red, are the two waterproof bottles that contain the electronics, communication and processing equipment and batteries. The tether connects the topside console to the communications and processing equipment to control Proteus. This equipment relays data from the sensors and camera (pink) to the operator. Lights, in green on the model, allow a better picture for the camera. The processing equipment also controls the propulsion system that consists of motor controllers and four thrusters (black) to propel the ROV through the water.

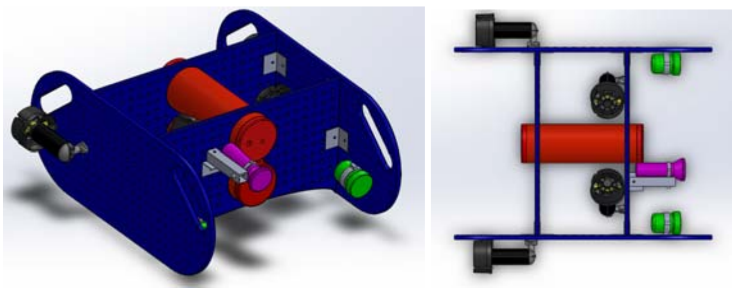


Figure 9: CAD model of Proteus the ROV from two angles.

2.5 - Team Management

2.5.1 - Project Challenges

Challenges faced while working on this project were maintaining the different budgets we had. Cost, power and weight do no always work together in favor of the design and customer requirements. Cost, power and weight effected all decisions when it came to picking parts for the ROV. Because of this, each piece went through the process of considering all options, weighing pros and cons and making trade-offs with the three categories.

We also faced challenges when it came to getting parts. Because of the short period of time we had to work on this project, it was essential to get parts on time, however, this did not always occur. This project allowed us to experience real work environment problems, including not being in control of everything. From this we learned that we should have ordered things as soon as possible rather than at the last minute.

2.5.2 – Budgets (Cost, Mass, Power)

Our project had a budget of $15,000 dollars that came from our customer, the RSL. In the end, we spent around $9,000 on the ROV including testing costs and donations. We were able to stay under budget by finding alternatives to expensive items. We were lucky in that usually the less expensive options would work for us because we were not going past 500 feet underwater. Refer to Appendix C for the Bill of Materials and Cost breakdown.

Our customers wanted an ROV less than 75 lbs and in the end, it weighed 49 lbs. A large factor in weight reduction was the frame material; we went with the lighter option because it was a large percentage of the total mass. We had a constraint of 75 lbs because the RSL wanted an ROV that could be safely lifted by two students and was easy to transport. A mass breakdown can be seen in Appendix D

In an attempt to make the ROV safe, we limited the power to less than 48 volts. We put a battery onboard the ROV rather than have a generator topside that would send power down the tether. When sending power down a tether, tether losses require a higher voltage top-side. For the tether length required by our project, this would have required a system with more than 48 volts. This limited the power for the ROV so a budget was made based on the components we needed on the ROV. This can be seen in Appendix E.

2.5.3 - Timeline

This project was started during the summer of 2013 by two of our team members who were testing previous ROVs, specifically the PVC ROV system, in an attempt to learn from flaws in the project. We were designing the ROV until the middle of Winter quarter when we started to test and integrate the electronics. The project was delayed a bit due to machining the frame material. We had chosen high density polyethylene because we thought we could machine it here on campus; however, due to how thick it was, we could not use the laser cutter and we did not want to use the mill because it would have talk a large amount of time to machine it with all of the holes that we wanted in the frame. Instead, the material was taken to the Monterey Bay Aquarium Research Institute to be water jetted (a more detailed description of the manufacturing process can be seen in Appendix L). We were able to assemble and perform preliminary tests on the ROV before the Senior Design Conference. After, we tested Proteus in Tahoe and these results can be found in Chapter 4. Please refer to Appendix 6 for the project timeline.

2.5.4 - Design Process

For every part of the project, the team went through a process of finding the most effective and functional solution. There are many things that were considered when making decisions including cost, functionality, weight, customer needs, etc. To efficiently design the robot, each part went through a design process. First, general questions were asked to make specifications and requirements. These answers were used to find possible options that were talked over with the team. Lists of pros and cons were created for each idea and comparisons were made. From here, we made decisions based on what the robot needed, making trade-offs in all requirements and making sure the team agreed; when the team did not, it was back to the drawing board to find more solutions/options and the process started again.

2.5.5 - Team Management

Our senior design team was composed of five mechanical engineering students sharing one common goal: build a fully functional, reliable underwater ROV (Remotely Operated Vehicle) that would be used as a benchmark for future underwater ROV's. Achieving this required a large amount of intelligence, hard work, and time. By the middle of spring quarter 2014, all of the fabrication, assembly, and testing of our underwater ROV was completed. This was a very difficult task to take on so we decided, as a group, to lay out two main ground rules to follow as we worked our way through our senior design project.

By far, the biggest constraint on our group was being on schedule. Starting from the beginning, we had about seven months to design, build and prepare our own underwater robot for testing in Lake Tahoe. We could not risk rushing through an assignment or task incorrectly because we simply did not have enough time to go back and repeat it. To prevent rushing and mistakes, we tried to put deadlines on projects that were sooner than required and built in time just in case things took longer than expected.

Rushing through anything almost always results in mistakes being made and with something as complicated as an ROV, we tried our best to limit these mistakes.

Our second and most important goal was to efficiently and amiably work together as a team. One of the big reasons many groups have a difficult time with their senior design project is the lack of communication and friendship within the group. Without a sense of comradery, any group would find it very hard to organize and operate as a single unit, which was definitely needed since we had such a small amount of time. All in all, we followed our ground rules and were able to successfully achieve our one common goal of designing a reliable, underwater ROV.

Chapter 3 - Subsystems

The robot was divided into ten sub-systems. This chapter discusses the requirements, options and testing methods of each system. It was important that we understood our customer requirements and material/system limitations to assess which product to use. There were at least two options for each system. The pros and cons of each were weighed to make the correct decision for the team and customers.

3.1 - Floatation

One subsystem of the mini ROV was the flotation. The ROV required floatation in order to remain positively buoyant while in the water, as well as not sinking straight to the bottom. The flotation mechanism was affixed directly to the ROV frame, usually on the top to prevent a rollover that could have happened with bottom mounted floats.

3.1.1 - Requirements

There are several ways to create the flotation for an ROV, but the best type for each one generally depends on the size and weight of the ROV as well as the desired depth. The depth dependency is due to the loss in buoyancy which many materials experience under water pressure. For example, PVC ROV is very small and light with a shallow-water depth rating such that pool noodles suffice as the source of flotation. For our ROV, we needed a foam option that was durable, easy to work with, and relatively inexpensive.

3.1.2 - Options

Four options for the flotation device were considered; syntactic foam, welded tubes, polyurethane, and a BCD scuba bladder. Below are the pros and cons of each product.

Table 4: Pros and cons for flotation.

	Syntactic Foam, Zolotone Sealed	Welded Tubes, (pontoons)	BCD Scuba Bladder	Polyurethane
Pros	o Incompressible o Can be used structurally	o Less expensive than foam o Accessibility of parts (metal)	o Variable buoyancy for different water o Compresses down o Off the shelf parts o Cheap and low weight o Efficient power usage	o Cheaper than other types of foam o Easy to machine and deal with o Rigid design wont rupture
Cons	o Primer/Sealant hard to come by and work with o Expensive ($500/ft^3) o Detailed sealing process o Small cracks will soak up water over time and screw up buoyancy	o Welding compared to light fab o Cannot shape to form fit ROV o Not structural o Pressure bomb potential o Heavy	o Fragile, can burst if punctured o Need a control system to account for bladder compression o Need air tank and valve o More maintenance and parts	o Compresses at high pressures o (more than 500ft) o Heavier than a scuba bladder

For the first option, syntactic foam, we used these calculations to find that about 17.28 cubic inches of the foam produced sufficient buoyancy. This wasn't a bad size; however, syntactic foam is incompressible due to the fact that it is made with small glass beads so cutting this material can be dangerous. Also, sealing the material is complicated, and the sealant itself is very hard to find.

The scuba bladder would have been the best way to increase the buoyancy because it would have allowed us to change the buoyancy at any given time. However, we would have needed a way to control it, and that just added more maintenance and complications on top of an already complicated system. Using welded cylinders like you would see on a pontoon boat would be another good option but they would have to be welded to the frame and are very heavy.

The fourth option was polyurethane foam, and it was what we ended up using. While not inexpensive, this foam was easier to machine, easier to acquire, and easy to seal through the use of a readily available wood sealant.

Figure 10: Sample of flotation material.*(Photo: Alex Waschura)*

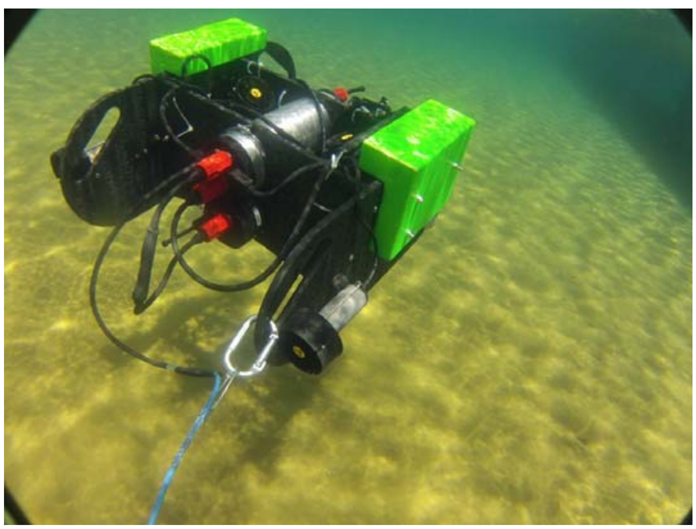


Figure 11: Proteus underwater with the flotation in green. *(Photo: Robert Heinevetter)*

3.1.3 - Testing

The testing done on the flotation material consisted of different methods of shaping it. For the foam, the buoyancy is very well documented by the company, and we based our calculations off the company specs. Tests showed that simple woodworking tools were sufficient in cutting and shaping the material.

Calculations were done to find the required amount of flotation to keep the ROV slightly positively buoyant and these can be seen in Appendix G. After it was cut and holes were drilled for mounting to the frame, the material was sealed with a simple deck sealant, spray painted and then sealed again. Next, they were put on the ROV and the whole system was put in the water to see if it was slightly positively buoyant. We had overestimated the weight of the ROV, believing it to be more negatively buoyant than it actually was. In the end, we decreased the size of the flotation by almost half and changed the shape so that it fit better and was positioned higher on the frame of Proteus. The flotation can be seen in Figure 11 in green. We included more flotation in case it is need when an auxiliary manipulator or sensor is added in the future. In the end, we added 24 oz of fishing weights so that, with combined flotation and ballast, the ROV was slightly positively buoyant.

3.2 - Frame

3.2.1 - Requirements

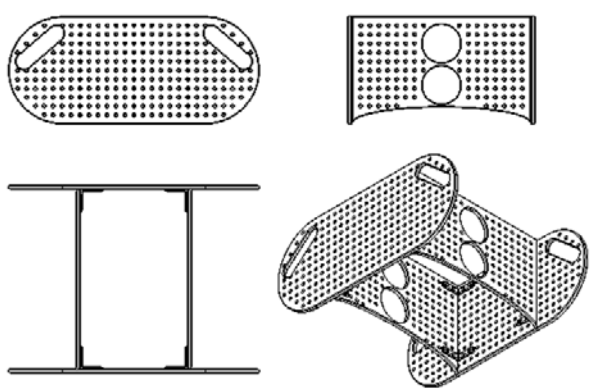


Figure 12: Computer Aided Design of frame.

The frame is essentially the skeleton of the ROV. It determines how the robot moves through the water as well as where all the components go and how easy it will be to handle. There are many different materials that are used for ROV frames, some more suitable in different applications. For our ROV we were looking for a material that was easy to manufacture or make into the correct shape. This requirement meant that we would either have to make it in the machine shop at school, or be able to out-source it at a low cost. This brought us to our next requirement, cost. We were trying to make this an inexpensive ROV, therefore our frame needed to fit the budget. As for strength, the frame needed to withstand the pressures at depths of up to 500ft and be able to hold all the components while not deforming due to the weight. We also had a weight budget and tried to keep the weight as low as possible so that two to three people could launch the robot.

3.2.2 - Options

Here we provide a comparison of the two materials we considered for the frame. They were HDPE, a high density plastic polymer, and aluminum.

Table 5: Pros and cons for frame material.

	Aluminum	**HDPE**
Pros	o High strength o Rigid design allows for strong mounting points o A very popular choice of material o Relatively cheap	o Very light o Very easy to machine on in house laser cutter when thickness is small o Infinite design options for a very streamline vessel o Can collapse down easily for transport.
Cons	o Al welding would need to be completed out of house. o Heavy compared to HDPE o Water proof welds are tricky o Corrosion possibilities	o Not as strong o A little more expensive.

Overall, the best choice was HDPE. HDPE was the perfect material because it is light, versatile, and easy to manufacture. HDPE allowed for more creativity in the design, and the attachments could be adapted with ease. Even though the material was not as strong as aluminum, we did not foresee a problem. The price difference greatly outweighed the cost of the aluminum, especially when including the cost of welding the frame.

3.2.3 - Testing

The preliminary testing of the frame material was done with SolidWorks simulations. A weight of 50 lbs, to simulate the load due to the bottles and electronic equipment (it is an overestimate of what it would be carrying), and gravity was applied to the modeled frame. In the simulation, the ROV was supported by all four hand-holds as if being held up by two people. The thinnest parts of the material were around the hand-holds so this was an area for concern. Below is a picture of the calculated stress on the frame with red being high stress areas.

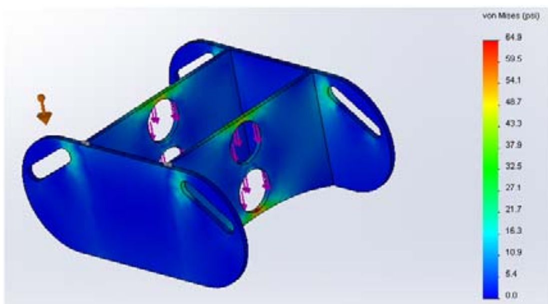


Figure 13: Frame held by two people showing stress.

In this simulation, the points around the hand-holds reach a maximum of 65 psi when held by two people at both ends. This simulation, although not perfect, projected that we had a factor of safety of 58.

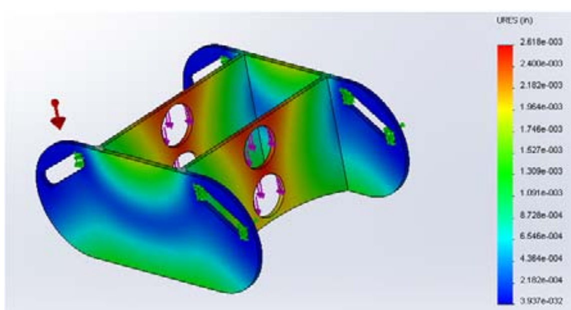


Figure 14: Frame held by two person showing deformation.

The deformation of the material was also simulated due to a 50 lbs load and gravity when held by all four hand-holds. The maximum deformation was found to be .003 inches.

The second part of testing was concentrated on how to manufacture the frame. The frame was initially going to be cut by a laser cutter; however, the thickness of the frame material was too large, and the laser cutter on campus was unable to cut it. We tried with different speeds and settings; however, it was not going to work. In the end we were able to use the water jet cutter down at MBARI to cut the material. A more detailed description of the process can be found in Appendix L.

Another test for our frame material was the buoyancy test. Since it was very important to make sure our ROV was positively buoyant, all positive and negative buoyant forces were calculated. For the frame, we put the pieces in the pool and weighed them with a fish scale and they were basically neutrally buoyant. The strength of the material, being HDPE, is very well known, and since the thickness of the material is more than sufficient, no tests were conducted in regards to strength other than simulations.

3.3 - Waterproof Housing

There were two waterproof housings on Proteus, one contained all electronics on the underwater ROV and the other contained the battery. This section describes the requirements and options for the housing including material and what it was filled with. It is concluded by the testing that was performed on the bottles after they were chosen.

3.3.1 - Requirements

When it comes to making a waterproof housing for robots it seems like every company has a different idea of what is right. The requirements were simple: it needed to be waterproof down to a maximum of 500 feet, light weight, and inexpensive.

3.3.2 - Options

Unfortunately, these three requirements lead to three very different options. A basic overview of waterproof housings is below; this includes the review of homemade options (ABS tubes), water proof boxes (otter boxes) and marine grade waterproof bottles.

Table 6: Pros and cons for waterproof housing.

	ABS tubes	Otter Boxes	Bottles
Pros	o Cheap o Easy to find supplies	o Inexpensive o Off the shelf part	o Water proof to any depth depending on build
Cons	o Quality of piping determines seal	o Water proof to 100 feet o Size	o Expensive o Have to be custom made or ordered

Due to the depth restraints we worked with, we decided to go with custom made bottles. This was unfortunate because the other alternatives would have been a lot easier to work with, but we did not have much of a choice. When it came to bottles, there are still a few options: mineral oil filled, permanently sealed, and air filled with removable end caps. These are reviewed below for pros and cons.

Table 7: Pros and cons of what to fill the waterproof housing with.

	Mineral Oil	Permanently sealed	Air filled
Pros	o Most reliable form of water proofing	o Second most reliable form of water proofing	o Maintenance
Cons	o Maintenance o Added cost	o Maintenance	o Less reliable

Since our robot was experimental and required that the electronics be periodically swapped out and worked on we decided to go with an air filled bottle with removable end caps. This was an expensive, more risky decision, but it was required for the type of robot we built.

Figure 15: Waterproof housing. *(Photo: Alex Waschura)*

3.3.3 - Testing

Testing for the waterproof housing was simple. The end caps were put in and sealed, and the housings were submerged for a day to check for obvious leaks. They stayed dry. This allowed us to verify that they were water tight before we put expensive electronics onboard. There was a greater chance of it leaking at greater depths; however, we had to wait to test that in Tahoe when the whole ROV was doing its final check because we did not have enough time to test the bottles in MBARI's 40 foot deep tank.

In Tahoe, we did notice a leak in the electronics bottle, but it was slow enough that we could complete other testing before we harmed any electronics. Because of this leak, we did not try to test the ROV at 500 feet. When we got the bottles, one was not within tolerance. We had hoped it would be okay, and we did not have the time to replace it. We were wrong, but students in the future can replace the bottle.

3.4 - Power

Typical power supplies for large scale commercial ROV's involve above water inverters that transform power into a high voltage supply to run down the tether. They do this because, over a long distance, the voltage drops significantly across the tether. For our system, safety was a primary concern. This prevented us from using a conventional high power system. We decided to use battery power as an alternative.

3.4.1 - Requirements

Batteries and power supplies always seem to limit the capabilities of remotely operated vehicles because of the available options on the market. For our project, we needed to consider a few requirements and chose the power option that was best suited. These requirements included being safe to use, low voltage, at least 280 Watt hours, light, small enough to fit in the waterproof bottles, low cost, and easy to charge.

3.4.2 - Options

Due to these limitations, we considered three main ways to power our robot. The first was a high voltage power system that uses inverters above the surface of the water to feed high voltage power down the line to the robot. The second way we considered was lead acid batteries and the third was lithium polymer batteries. Below is a list of pros and cons of every system.

Table 8: Pros and cons for batteries.

	High Voltage Power lines	**Lead Acid**	**Lithium Polymer (LiPo)**
Pros	o Endless power	o Cheap o Easy to charge	o Smallest battery per KWh o light
Cons	o Large inverters o Hazards because of high voltage lines o Larger tether o Most expensive	o Heavy o Larger than LiPo per KWh	o Expensive o Hard to charge

Between these three options we came to the conclusion that Lithium Polymer (LiPo) batteries were the best option. Although high voltage power lines would have more likely been the best option because it would take up less space on the ROV and would last longer than a battery, it would increase they set-up time and required supplies so we chose to use battery operation because of safety issues. Plus, due to size and weight constraints with our robot, we decided Lead Acid batteries were inappropriate. LiPo batteries required us to overcome the charging issues since these barriers require each cell be monitored while charging, but the expense was well worth it. For these batteries, they have to be removed from the bottles to be charged, and they have a special charger that monitors the six cells in the battery in order to ensure they are being charged correctly.

Figure 16: Lithium Polymer batteries. *(Photo: Alex Waschura)*

3.4.3 - Testing

So far, we have used a 6 cell 22.2 C V Lithium Polymer batter to power the camera and thrusters, and it has worked as expected. The test with the thrusters is talked about in more detail in the thruster testing section (3.5.1.3) and motor controller testing section (3.5.2.3). The ROV was tested in Tahoe and, after a 30 minute deployments, we still had half of the charge left. During this deployment, the lights and camera were on and the thrusters were in constant use.

3.5 - Propulsion

The propulsion section includes the thrusters and motor controllers. Together, they control the speed and position of the ROV. It was important that they work with the Arduino microcontroller in the electronics bottles so we can propel Proteus through the water.

3.5.1 - Thrusters

Four thrusters were used on the ROV to propel it through the water. They were chosen from three different options for their depth rating, cost and low power consumption.

3.5.1.1 - Requirements

There are many different options for thrusters that are available on the market, but because this was to be a small hobby class robot it limited us down to a few options. The thrusters were to be low power consumption, low voltage(~24v), powerful enough to direct the ROV, easy to install and use, relatively low cost, and small.

3.5.1.2 - Options

In our research we came across three viable options: a brushless DC thruster made by Crust Crawler called the HFS-L, a brushed DC thruster made by Seabotix and a homemade thruster setup made from bilge pumps. The pros and cons of these different thrusters are below.

Table 9: Pros and cons for thrusters.

	Crust Crawler HFS-L	Seabotix Thruster	Bilge Pump
Pros	o High thrust rating (15 lbs) o Operates within 24V range o Low cost, same as Seabotics thruster	o 500 foot depth rating o Operates within 24V range o Low power consumption o Can use basic motor controllers	o Small o Very low power consumption o Cheap
Cons	o 300 foot depth rating o Requires specific motor controller o Brushless motor maintenance and use	o Not as powerful as HFS-L thruster	o Low Thrust o Depth rating of less than 50 feet

Although the final size and weight of the robot determined what thrusters we could use, we still compared what is available. For our project, we originally considered bilge pumps but quickly looked for alternatives. Although bilge pumps are easy to work with they lack a suitable depth rating and power to move a robot of our size. The next alternative we considered was a brushless motor made by Crust Cralwer, the HFS-L, and although this motor has a high depth rating and is very powerful, we turned away from working with it because it is a brushless motor.

If we were to use a brushless thruster for our project, special care would have had to be taken when choosing the motor controllers since brushless motors need specific controllers. Brushless motors also need more maintenance and are harder to keep serviced. This led to our final choice of thrusters, the Seabotix thruster. We came to this conclusion because it was the best overall thruster of this size. It has a depth rating of 500 ft, operates in the power range we decided on, could be used with basic motor controllers, and has low power consumption. This made it the perfect candidate for the Proteus.

Figure 17: Seabotix thrusters. *(Photo: Alex Waschura)*

3.5.1.3 - Testing

For testing purposes, two thrusters were put onto a prototype PVC-ROV frame with a plastic board on the front of the ROV to increase drag. Using a motor controller (RobotEQ SDC 2130), battery and laptop to send direct commands to the motor controller to control the thrusters, we ran our tests. The test rig was put in the pool with one team member holding on to it for added weight and stability. Using the two thrusters, the ROV was able to drag 170 lbs. At half power we were still able to move, but did not characterize the speed. At full power, the test set up showed a speed of about 1 ft/s. Another test was conducted without added weight, and the speed of the ROV was just over 2 ft/s.

3.5.2 - Motor Controllers

3.5.2.1 - Requirements

Although there are seemingly endless options when it comes to motor controllers, we limited our choices down to three based on the requirements of being a compact size, having power handling capabilities, being easy to use, and being relatively inexpensive.

3.5.2.2 - Options

The three options that were considered were the multiwatt15 (L298n) dual channel motor driver, the RobotEQ SBL1360 and the RobotEQ2130. These were compared and a list of pros and cons are in the tables below for ease of reference.

Table 10: Pros and cons for motor controllers.

	Mutliwatt15 (L298n)	**RobotEQ SDC2130**	**RobotEQ SBL1360**
Pros	o Compact size o Very easy to use o Cost o Dual Channel	o Dual Channel o Powering ratings 30V/20A o Lower cost then SBL1360 o Ease of complex programing	o Brushless motor controller o Power ratings of 60V/30A o Ease of programing
Cons	o Limited power ratings 46V/4A o Lack of monitoring abilities and programing capabilities	o Size o Cost compared to Multiwatt15	o Cost o Size o Single Channel

These three motor controllers were researched and considered. In the end we determined that because of the limited power ratings and lack of monitoring abilities, the multiwatt15 should not be used. We also determined that the RobotEQ SBL1360 should not be used because it is a single channel motor controller and cost more money, plus brushless motors are more difficult to work with. That led us to our final choice of the RobotEQ SDC2130. This motor controller cost less than the SBL1360, handled twice the number of motors per controller, and retained all of the positive benefits like the ability to limit current and monitor the important vital signs such as voltage and temperature.

Figure 18: RobotEQ SDC2130 motor controller. *(Photo: Alex Waschura)*

3.5.2.3 - Testing

The first tests conducted on the motor controllers consisted of using them to communicate with thrusters and supply a set current to the thrusters to adjust their speed. This was performed at full and half speed, limiting the current to 4.25 Amps. The second test was done using an Arduino, rather than a computer like the first test, to send data to the motor controllers. In this test, not only were commands sent, received and performed correctly, but information was also sent back including the battery voltage and temperature of the motor controller.

3.6 - Camera

3.6.1 - Requirements

ROVs provide the opportunity to view the underwater environment. In order to achieve this, we must have an onboard camera capable of providing a live video feed up the tether for the user to see. The camera must output a composite signal, be small, easy to use, low cost, and waterproof up to 500 ft.

3.6.2 - Options

For camera options, we looked at charge coupled device (CCD) cameras, packaged with/out an underwater housing, and a GoPro HD Hero2. A comparison between the options can be seen in the table below.

Table 11: Pros and cons for camera.

Criteria	**CCD board camera**	**CCD packaged**	**GoPro HD Hero2**
Cost ($)	~50	~700	~200
Housing	None	Plug and play	None

We decided to go with a packaged CCD camera. We found the ROVSCO RD-400, at $690. . Despite the lower cost of the GoPro, buying a suitable underwater housing rated for 500 ft would end up costing about the same.

Figure 19: ROVSCO RD-400 camera. *(Photo: Alex Waschura)*

Machining our own housing would have decreased the cost of the camera considerably. We decided against this though, because machining the housing for a camera that needs a clear screen to view outside and required mounting was more than we wanted to take on. We would also have had to test the housing and make sure it was reliable before it would be approved for use. Buying a pre-packaged camera eliminated this necessity, as it was known to be reliable, and if any issue arises, we could contact the manufacturer. In the end, we justified paying a premium for a camera, rather than having to put in the considerable amount of time it would have taken to machine the housing ourselves.

3.6.3 - Testing

The camera was powered by the LiPo battery and connected to a computer monitor via a composite plug, displaying the camera feed on the computer. The connection was also tested over a 100 ft CAT5e cable, the tether material, with little quality loss. The next step was to test the camera underwater across a 500 ft CAT5e tether which was done in Tahoe. For this test, the ROV was completely assembled, the 500 foot tether connected it to the topside control box, and it was submerged in the water. We looked at rock and sediment layers with the small 5” by 7” screen, and there was little quality loss.

3.7 - Lights

There are two lights on the Mini ROV, one to the left and one to the right of the camera. Even though the ROV went to a maximum depth of 500 ft, where light still penetrates through the water, lights in general were a requirement because this vessel is for exploration. Lights make it easier to find and identify things in the marine ecosystem.

3.7.1 - Requirements

The lights that are used on Proteus were required to have low power consumption because we were limited by how much battery power we can have on the ROV. They had to be relatively low cost to maintain our budget of less that $15,000. More requirements for the lights include them being durable, having a variable intensity, having good illumination at our maximum depth, and having a long life cycle.

3.7.2 – Options

With these requirements, three options were analyzed to see which fit the criteria the best. The three options were halogen, high-intensity discharge and LED(ROVSCO SEADragon). Below is a table of the pros and cons of each type.

Table 12: Pros and cons for lights.

	Halogen	HID	LED(ROVSCO SEADragon)
Pros	o Good even lighting o Dimmable	o Low power consumption o Best light intensity	o Low cost o Low power consumption o Resistant to shock/durable o Dimmable o Long life cycle
Cons	o Fragile Bulb o High power consumption o High operating temperature o Short life cycle	o Start up/warm up procedure o More expensive than LEDs o Temperature issues o Non-dimmable	o Not the best intensity compared to other options o Tend to produce backscatter

After looking at all of the possibilities, the LEDs (ROVSCO SEADragon) were the lights that were mounted on Proteus. This was because they met the greatest number of criteria being low cost, low power consumption, durable, dimmable, long-lasting and safe. Although they are not as bright as the other options, it was still enough for our purpose. They are equivalent to a 300-watt halogen light bulb and a 15-watt LED array. Although the halogen is brighter, the ROV in general needed to be durable, and it was a safety hazard as well as costly if the lights were fragile. High operating temperatures also made them a safety concern which was one of the most important requirements for the ROV. The HID lights are also brighter but need to warm up, which went against the requirement for the ROV to be able to deploy easily in under 15 minutes. The cost and temperature issues also made them the wrong choice.

Figure 20: ROVSCO SEADragon Light *(Photo: Alex Waschura)*

The lights have been tested in multiple ways. The first was to verify that they work with no other systems connected. The lights were individually hooked up to a DC power supply and given 22 volts and as much current as they would draw. They both successfully worked and seemed bright enough for the missions we have in store for them. We wanted to be able to turn the lights on and off with the topside console, so we used a solid state relay (SSR) to control the lights. To test this, the Arduino sent a 5 V signal to the SSR, and that allowed current to pass through to the lights. In the future, we would like a dimming capability.

3.8 - Communication

This section includes the elements on the ROV for communication between the robot and the topside console.

3.8.1 - Communication Protocol

3.8.2.1 - Requirements

To communicate with the ROV through the 500 ft of tether, we needed to establish a communication protocol. Passing data through 500 ft lines is not trivial. There are voltage losses, and possibly electrical noise that can corrupt the data. The protocol we used required a minimum number of lines to transmit data, and could do so quickly. If communication was slow, we wouldn't have been able to adequately control the ROV.

3.8.2.2 - Options

We looked at the two industry standards used for ROVs, RS-232, and RS-485. The features of each can be seen in Table 13 below.

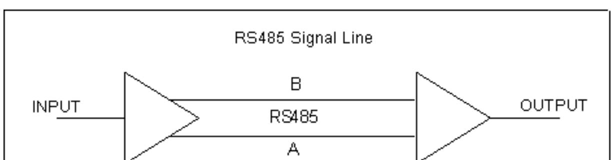
Table 13: RS-232 and RS-485 features

Criteria	**RS-232**	**RS-485 Half Duplex**	**RS-485 Full Duplex**
Data rate	9.6 kbit/s @ 500 ft	660 kbit/s @ 500 ft	660 kbit/s @ 500 ft
Transmission	Simplex (Single line)	Differential (twisted pair)	Differential (twisted pair)
Required lines	3	3	5
Send/receive	Simultaneously	Coordinate between the two	Simultaneously

RS-232 operates as a simplex operation, using a single line to transmit, a single line to receive data, and a ground or reference line. Because of this design, it is very susceptible to data corruption in electrically noisy environments. It is very simple to use, but it is slow compared to what most systems use for the transmission rate.

RS-485 uses differential transmission. It uses twisted pairs to send or receive data. There is an A (-)/inverting line, a B (+)/non-inverting line, and a ground line. Lines A and B are logical opposites. Figure 21 below shows the operation of RS-485.

This differential transmission makes RS-485 great for operating in electrically noisy environments, because both lines will be affected in the same way, but will cancel each other out when determining an output. 485 is also great for long distance transmission and speed. Despite this, it does take more effort to use than RS-232. RS-485 requires that the lines be balanced in order to transmit successfully. This means that a resistor must be placed at each termination, connecting the two lines (A and B). This resistance must be the same as the characteristic impedance of the lines. If the lines are not balanced correctly, the signals received will be distorted and may not be read properly.



INPUT	A	B	OUTPUT
0	1	0	0
1	0	1	1

Figure 21: RS-485 logic table

There are two ways to use RS-485: as half duplex and as full duplex. Full duplex utilizes a twisted pair to send data, and a twisted pair to receive data, or 5 lines in total, including ground. Half duplex utilizes a single twisted pair to send and receive data. This means that it cannot do both simultaneously, and the microcontroller must coordinate between receiving and sending to ensure the message gets through. This is done by switching an enable pin on and off at the appropriate time to ensure that it is in send mode when it needs to, and in receive mode when waiting for a packet.

We decided to use RS-485 half duplex for our project. We chose RS-485 over 232 for its superior transmission speed, longer range, and great noise handling capabilities. Since we had thrusters and lights that have constant on/off switching, they created a source of electrical noise, and we wanted to avoid any issues that they might have caused.

We went with half duplex over full duplex to reduce the number of lines required for transmission. Tethers are expensive, and lines are at a premium. We could lower the cost by removing 2 lines required to communicate, and/or we could free them up to have the possibility to expand functionality. These are both very valuable. All of the RSL robots are required to have a “speak when spoken to” protocol, thus we are only required to communicate one way at a time. The benefits provided by half duplex outweigh the need to implement software to enable switching between transmitting and receiving, which we believe will require considerable effort to ensure proper communication.

3.8.2.3 - Testing

Information on testing the protocol can be found in the testing section of the Processing section (3.9.3).

3.8.2 -Tether

3.8.2.1 - Requirements

A tether connects the topside console to the ROV for communications purposes. It had to transfer information 500 feet down to the ROV as well as a live camera feed and sensor data from the ROV back to the topside controller. However, it did not have to transfer power because there are batteries on the ROV. Some more requirements for the tether included that it must be low drag, low weight, low cost, and preferably neutrally buoyant to not effect ROV performance.

3.8.2.2 - Options

With these requirements, four options were considered; fiber-optic, copper, copper with no buoyancy and CAT5e.

Table 14: Pros and cons for tethers.

	Fiber Optic	**Copper**	**Copper(no buoyancy)**	**CAT5 Ethernet cable**
Pros	o Thin o Fast transmission o Low weight	o Neutrally buoyant, less effect on ROV o Industry standard o Easy to terminate o More durable than fiber optic	o More durable than fiber o Easy to terminate connections o Cheaper than neutral buoyancy	o 8 transmission lines o More noise immunity o Require less programming o Less drag, lighter, smaller o Least expensive
Cons	o Difficult to terminate connections o Expensive o More fragile than copper	o More expensive than no buoyancy o Thicker than fiber	o Can dominate ROV handling/perform ance o Thicker than fiber	o No protective jacket o Not neutrally buoyant

For Proteus, after weighing the pros and cons, the best option was CAT5 Ethernet cable. This tether has less drag due to its small size. Although it is not neutrally buoyant, testing showed that this was not a problem when in the water. It was much less expensive than the other options, being less than $100 for 500 feet. Because of the greater number of lines, we were able use the RS-485 with less programming and more flexibility.

Figure 22: Tether

The CAT5 cable was used while testing the different components of the ROV and did not corrupt/interrupt the data. We also tested how strong the cable was, just in case we had to use it to pull the ROV through the water. It successfully held 160 lbs. and did not break any connections.

3.9 - Processing

3.9.1 - Requirements

The processor of the ROV controlled all parts of the robot so it was required to be able to execute all possible actions and be able to cooperate with all electronics, motors, and sensors on the ROV.

3.9.2 - Options

The two possible options for the processor of the robot were the Arduino and the Raspberry Pi. They are two different pieces of hardware, each with its own benefits and disadvantages.

Table 15: Pros and cons for processing system.

	Raspberry Pi	**Arduino Mega**
Pros	More functionality Small	Always ready to go Less expensive Small
Cons	Requires operating system Requires time to boot-up More power consumption More expensive	Less functionality

The Arduino is a micro- controller with less functionality than a Raspberry Pi. The Pi is a mini-computer and requires an operating system. Because of this, it requires time to boot up, about 30 seconds, while the Arduino, once the program is installed the first time, can start working immediately. They are around the same size, but the Pi requires 20 times more power than the Arduino.

Figure 23: Arduino Mega *(Photo: Alex Waschura)*

Although the functionality is limited with the Arduino, it was the product that was used to control Proteus. The three most important reasons were that Proteus did not need all of the functionality that the Raspberry Pi offers, the Raspberry Pi needs time to boot-up and a customer requirement was that the ROV be quick to deploy, and our customer, the Robotics Systems Laboratory, preferred the lab standard Arduino over the Pi unless the system demanded the computing capabilities of the Pi, which it did not. This was because many of their other robots run via an Arduino so it would be easier to stick with a system that is known and understood by the RSL and can therefore be easily changed, if needed, down the road.

3.9.3 - Testing

The first test had the ROV Arduino connected to the motor controller and compass. The top Arduino was connected to a laptop and two potentiometers simulating joysticks. The joysticks are two axis potentiometers so it was a good simulation. Between the Arduinos there was a 100 ft CAT5e tether. To convert from Arduino logic (TTL) to RS-485, there was a MAXIM 488E chip at each end.   We were able to get full duplex RS-485 communication to work. We also hooked up the Arduino on the ROV straightto the laptop and by sending a command packet, we were able to control thrusters and a receive data packet back.

The testing for the Arduino in the topside box was testing communication between the ROV Arduino and the topside console one. While testing, we sent down packets of code that were preassembled to the ROV Arduino, and we received a sensor data packet. Next, joysticks sent commands down the tether to the Arduino, and data was returned up the tether and was displayed on the laptop. Lastly, we tested the sensors, thrusters, and camera together. We then did a lot of troubleshooting to get all of the data to display correctly on the LCD screen, to make sure packets of information were sent, and played around with the joysticks to map the commands we wanted.

3.10 - Sensors

Most ROVs have the capability to transmit data from the ROV to a topside console; we planned to implement the same capabilities. The ranges of sensors and instruments that can be used are endless. Our ROV has the capability to expand and add more sensors as needed due to an auxiliary port; however, there are some sensors included on the basic ROV. These sensors included a depth, temperature, humidity, and magnetometer sensor.

3.10.1 - Requirements

The requirements for these sensors were that they must reliably and easily interface with an Arduino Mega microcontroller, and be low cost. The depth sensor had to be rated for more than 222 PSI (500 ft). The temperature sensor had to be rated for temperatures ranging from -10 to 40 ° Celsius; this covered most water temperatures the ROV was intended to operate in.

3.10.2 - Decisions

For the depth sensor, we used a Digi- Key MLH250PSL09A pressure sensor. This is a high quality sensor, rated to 250 PSI. It had been used in the RSL before, and thus there was documentation, and resources were available on how to use it with an Arduino.

The water temperature sensor we used is a Texas Instruments LM35 temperature sensor. It has a range of -55 to 150 ° Celsius. It is very inexpensive (~$2), and widely used with Arduino’s, thus there are many online resources for using it. It also produces a linear output voltage, making it very easy to interpret the data.

Figure 24: Texas Instruments LM35 Temperature sensor. *(Photo: Alex Waschura)*

The humidity sensor was mounted in the main electronics bottle in order to sense a bottle leak. Three types of humidity sensors were considered, the DHT22, SHT15 and HIH-4021-003.

Table 16: Pros and cons for humidity sensor options.

	DHT22	SHT15	HIH-4021-003
Pros	o Low cost ($12.50) o Smaller than quarter o Arduino code already written o Max current 2.5mA o Temperature range of -40 to 80degrees Celsius o Measures temperature	o Smaller than a dime o Low power consumption o Better accuracy o High precision o Measures temperature	o Size between other two options o Medium cost ($18) o More accurate that DHT22 o Low power .5 mA o Easily used with arduino, code available
Cons	o Data read /2 seconds	o More expensive($28)	

From the pros and cons above, the HIH-4030 fit the requirements the best. Cost wise, it is average; however, it requires less power than the least expensive option, is more accurate and is smaller. The HIH-4021-003 also has code available to use with the Arduino Mega microcontroller.

To collect yaw, pitch and roll data, we used a Devantech CMPS 10 magnetometer. The yaw, pitch, and roll are the rotations around the x-axis, y-axis, and z-axis. The information is used to figure out the orientation of the ROV underwater and can be used in the future to help characterized the system dynamics of Proteus. The CMPS 10 is a lab standard at the RSL, and was used for PVC ROV. There has been significant work that has been done on it, and many resources available to help with the implementation. It is low cost (~$35) and is easy to interface with an Arduino. The only downside to this magnetometer is that it is very sensitive to electrical noise from thrusters/lights/power lines. For this reason, it is mounted far away from these components to give satisfactory data.

Figure 25: Devantech CMPS 10 magnetometer *(Photo: Alex Waschura)*

3.10.3 - Testing

Before the bottles were assembled, the sensors were tested before being mounted in the bottles and before the ROV was put in the water to make sure they worked. These tests were performed above water and the data received was accurate to the location of the sensors. The temperature sensor was tested using an Arduino program to run it, and it registered a room temperature of 23 degrees Celsius. The compass registered the change in yaw, pitch and roll when it was moved and rotated. The humidity sensor was also hooked up to the computer, and data was collected from it. In the room, it read a humidity of 27%. The pressure sensor was tested and it sent back information that was consistent with the sensor being above water.

3.11 - Topside Console

The topside control box was designed to control the ROV and display sensory information. The box itself it a Seahorse hard case that is weatherproof and waterproof, since it will most likely be used in environments where it could come in contact with water. The ROV tether connects to the back of the case using rugged military style connectors. The inside of the box contains the electronics necessary to control the ROV.

The control box can be powered using a 12 VDC adapter with a 2.5 mm plug, as well as using an internal battery. These power modes can be selected using a switch on the control panel. There is large red killswitch that cuts power to the control box, as well as an on/off rocker switch that does the same.

The live video coming up the tether is fed directly into an RCA jack mounted on the back of the case, allowing any RCA cable to fit the connector.

Figure 26 : Topside Control Box. *(Photo: Alex Waschura)*

An Arduino Mega was the microcontroller of choice in order to keep microcontrollers on the project standard. The tether communication lines go through a MAX488 chip, just like what is on the ROV. This converts the communication protocol from RS-485 to an Arduino friendly TTL. The data received is displayed on a 4 x 20 LCD screen, and is labeled appropriately. Thruster commands are determined using two 2-axis joysticks. Camera and lights commands are determined using a series of switches mounted on the control panel.

There is an LCD screen mounted in the topside control box to display sensor outputs. These include the yaw, pitch, roll, water temperature, depth, battery voltage, humidity in the electronics bottle, battery voltage and the motor controller temperature.

Figure 27: LCD display. *(Photo: Alex Waschura)*

Extra switches were added to the control panel to make adding auxiliary functionality easy. An Arduino Uno was also included in the box with its programming port mounted outside. This Arduino has access to the tether lines by flipping a switch. The Arduino Uno was specifically added in order to connect with the tablet interface developed by the Computer Engineering senior design team advised by Dr. Figueira. This port can also be used with a laptop interface in order to test autonomous controllers developed by students at the RSL or to try out different control interfaces.

Chapter 4 - System Integration, Tests, and Results

4.1 - System Integration

Before our ROV was completely ready for launch, many different tests had to be completed in order to ensure that everything was functioning properly. First, the waterproof housings were submerged in a water tank for extended periods of time (1 day) to make sure they were waterproof at the surface. We also integrated all of the thrusters, electronics, sensors, camera, and lights to make sure they all would receive and send data to the topside consol. These tests are highlighted in the testing sections of each component.

4.2 - Tests and Results

Figure 28: Testing ROV off of the dock. *(Photo: Robert Heinevetter)*

4.2.1 - Field Tests

For the full validation test of our ROV, we went to Lake Tahoe. This testing consisted of several aspects. The ROV was used as a backup to Triton when deploying.

The Robotic Systems Lab was helping Rich Schweikert, a geologist working with the University of Nevada – Reno, Dr. Jim Moore from the US Geological Survey, and Dr. Winnie Kortemeyer. They wanted to continue surveying Lake Tahoe using ROVs from the RSL, an activity that has been conducted annually for the past decade.

Before using it as a back-up, we tested the 500 foot tether to make sure everything still worked, which it did, and that the video feed looked good on our monitor. Next the ROV was tested close to Camp Richardson in South Lake Tahoe. After the first test of about 30 minutes, there were a few things that we realized needed work. The compass needed to be calibrated since the readings it gave were off, only ranging from 50 to 180 degrees. The floatation and buoyancy needed to be adjusted in order to get the ROV very slightly positively buoyant, and weighed down in certain locations in order to adjust the trim (pitch) of the ROV. Besides that, the thruster flow in reverse was hitting the back plate, which impeded the ROV when turning and going in reverse. The thrusters were then mounted on the outside of the side panels, allowing water to flow freely. The floatation was moved forward and fishing weights were added to make the ROV neutrally buoyant and level (12 oz each side).

There was a concern that the LED lights on the ROV would not be powerful enough to light up a dark environment in deep water. To test this, the ROV was put in the water at night in the Tahoe Keys and the lights were turned on to assure that the light was adequate, and they were.

The next day, the ROV was deployed in Emerald Bay. The ROV went down to a depth of 75 ft. The compass onboard was still giving incorrect readings. The new thruster mounting location allowed the ROV to drive much better than before. After about 30 min, the ROV surfaced and was brought back on the boat. The main electronics bottle developed a leak and had some water inside. The marine plugs were retightened on the end cap to attempt to stop the leak, but the effort was unsuccessful. It was; however, slow enough to not damage the electronics. The battery life was still well charged after 30 min of continuous testing and a few smaller dives throughout the day.

All in all, the Tahoe trip gave us full validation of our system. It handles like we wanted it to, and we were able to actually conduct scientific missions with the ROV. It was extremely simple to set up and operate, taking only about 10 minutes and required only 2 people to deploy.

4.2.2 - Sensor Data and Verification

The compass, after being calibrated, worked correctly. We think we may have damaged the temperature sensor, because it was reading a temperature of 120 degrees in Lake Tahoe. The pressure sensor correctly relayed the depth to the topside console and the humidity sensor was able to tell us if there was water in the bottles, which there was.

Chapter 5 - Standards and Constraints

5.1 - Engineering Standards and Constraints

5.1.1 - Health and Safety

The health and safety pertaining Proteus falls into three categories, these being user, robot and environment. A main goal for Proteus was to be an underwater ROV that can be easily used by one to three people. This constrained the weight of the system to something one person could lift without hurting themselves or the robot in the process. Also, the frame was designed with handhold that were tested to make sure Proteus was easy to carry and fingers could not get stuck in the hand holds. The frame is also light but strong to make it easier for the user to pick up.

Secondly, the system has been designed to be low voltage so there is less danger for the user when it comes to setting up the robot for deployments. Having a higher voltage system increases the danger for the user and requires experts to operate. We limited our ROV to 24 volts for these safety reasons. There also an emergency shut off switch on the topside console to protect the user and the ROV. Because this underwater ROV will be used by students to come, it was very important that we designed a robot that was safe and easy to use to decrease the chance of harm. All systems have been well documented so if there are problems, the solutions are theoretically easy to find.

The robot must be safe from itself, whether that is in the programming or in a physical sense. For example, we do not what it to cut its own tether so the thrusters are shielded. The emergency shut off switch also helps the robot's safety because if there is anything wrong with coding and the robot freaks out, we can shut it off before too much damage is done.

Health and safety for the environment comes into play when deploying the robot. All materials will not start to degrade over time and possibly cause problems for the underwater ecosystems it is observing. Also, the enclosed propellers decrease the change of harming the environments as well as controlling the robot so it does not run in to or damage anything underwater.

5.1.2 - Environment

Marine ROV's are crucial to underwater research and exploration; however, while they can provide invaluable information on how to protect our aquatic life, they can also have destructive effects if not handled properly. Our submersible contains materials that are harmful to wildlife and the marine environment, but with proper care and maintenance, those materials should never have the chance to affect the environment. The batteries contain toxins, however, the toxins are sealed inside of the battery compartment, which is sealed inside a deep ocean waterproof container with very small change of getting out.

The frame is also composed of a plastic composite, but with proper maintenance, none of the plastic wills behind in the ocean. All pieces are secure, so nothing should be falling/breaking off. With deep sea exploration there is always the risk of damaging the surrounding environment and ecosystems. Our ROV is equipped with lights and a camera so we can see where we are driving under the water. This helps us avoid smashing into rocks or coral and disrupting the ecosystems.

As long as the ROV is well maintained, with no leaks in the waterproof container that holds the battery, our deployments should not affect the environment in the slightest. Also, resources are available (camera/lights) so the operator of the ROV is aware of his/her surroundings. Therefore, there should be almost no destruction to the marine ecosystems when using Proteus.

Environmental Disturbance Time (EDT) is defined as the amount of time that a measurement is taking place and disrupting the local wildlife and ecosystem. Our ROV is designed to quickly setup and deploy, efficiently using time and resources. Instead of needing a boat and 5 people to deploy, we can easily deploy on shore with minimal manpower. Our system will reduce the EDT in scientific missions, allowing the local wildlife to more quickly return to their normal actions.

5.1.3 - Politics

Deploying remotely operated vehicles in public bodies of water requires the handling of several legal considerations. Certain agencies and park managements require the craft to be deployed to undergo rigorous inspection before being given approval to deploy. Permits usually need to be obtained before the craft can touch the water. Due to environmental concerns, any craft that has been deployed in a body of water containing invasive marine species may not be deployed in a non-contaminated body of water for at least thirty days. Since our ROV can be deployed from shore, or off a dock, there's not a necessity to have a large vessel inspected and approved. Deploying off a dock might still require certain permits though.

5.1.4 - Manufacturability

Underwater ROVs, or any type of ROV for that matter, tend to be very complex and very expensive. That being said, it was important to keep in mind the price of the different materials that we used while trying to minimize our expenses as much as possible. The large price of our ROV also means that it will not be something that can be easily manufactured or mass produced. One of our main goals is to provide future undergraduates with a reliable ROV that they can work to improve so manufacturability isn't our largest concern. The frame will most likely be made out of laser-cut metal with syntactic foam for buoyancy. These are both relatively easy materials to work with and should not require too much effort to reproduce. However, waterproofing all of the electronics, especially at large depths, will be quite challenging. Overall, any type of remotely operated robotic system will be very hard to manufacture but this can be made a little easier by carefully analyzing the workability of each material.

5.1.5 - Usability

Our ROV can be used students and scientists, young and old, experienced or not. Our goal for Proteus was to be easy to use and it is. This means our ROV is extremely user friendly. The structure itself has handles to show what the optimal carrying position is. We also design the ROV so that minimal work is required to be done in order to get it in an operating mode and functioning in the water. The tether is plugged into our topside control box, which is design with simple controls for driving and lights with a screen that is plugged into the box so the user can see the live feed from the camera. There is also a simple LCD screen on the box that allows the user to know the output put of the variety of sensors onboard Proteus. This includes the water temperature, depth, humidity (to know of if the electronics bottle is leaking), battery voltage, temperature of the motor controller and values relating to the position of the ROV in the water.

Due to the battery we used and wiring issues, the ROV is turned on and off by a switch in the battery bottle. Before launching, this switch must be turned on and the battery bottle sealed. The tether needs to be plugged into the ROV, into a well designated plug and this is the only plug that needs to be connected to the ROV, the last things to put in are the purge plugs and they are attached to the frame so it is hard to forget. The ROV At this point, the topside console can be turned on by the switch on the box and Proteus is ready to go.

The ROV is to be maintained by students. This means that we documented all of the parts, wiring diagrams, schematics, etc. so that future generations can understand the inner workings.

Chapter 6 - Business Plan

Our goal would be to sell our ROV to educational institutions and the marine research industry for $14,000. As long as we are selling two or more units per year, we can make a profit. We will have to take out a small load for initial costs like space and tools. Because of how portable our unit is, we will advertise it by giving demonstrations. We will also rent out units of help programs share one if they cannot afford one individually.

6.1 - Background

The product we are trying to sell is a competitively priced small scale remotely operated underwater robot. This product is targeted at the marine research industry and educational institutions with the design of our robot allowing for not only use as a fully operational robot for exploring the ocean depths, but also as a test bed for control system based learning experiments. The way we designed our product allows us to market it in various ways since it is such a versatile instrument. Small and large companies can purchase this product at a fraction of the cost compared to other systems that are on the market.

6.2 - Business Goals and Objectives

Our main goal as a company is to break into an existing market with a product that is designed from the ground up like it should have been from the beginning. Our product is not new in theory, but in practice it is much different than those on the market today. It will include features such as an onboard microprocessor and highly adaptable frame as well as significantly lowering the price from the get go. We do not want to reinvent the wheel, just make sure that everything is working together perfectly and at a price point that would get more involved in the wonderful world of marine robotics.

6.3 - Elevator Pitch

The product we are selling is an underwater remotely operated vehicle and all necessary operating equipment. An ROV (Remotely Operated Vehicle) is a highly maneuverable robotic system that can be operated remotely. They are linked to the operating location using a tether, which passes commands and telemetry back and forth.

The ROV is designed to weigh less than 50 lbs, be easy to deploy by two individuals, and operate entirely on batteries for portability and ease of use. The ROV has an onboard camera and lights giving the operator an underwater view. Also included are temperature, depth, and heading sensors. It is also possible to build upon the base ROV and add sensor packages, more thrusters, a robotic manipulator or a water sampler.

The ROV will come standard with a 500 foot long tether and can be used in fresh or saltwater environments with soft currents.

6.4 - Potential Markets

There are several different industries where ROVs are used, with the three main ones being construction, military and port authorities, and science. ROVs are frequently used in the construction industry as inspection systems in underwater constructions, as well as for some light work when the ROV is fitted with a manipulator.

Militaries and port authorities like navies, coast guards, and police departments are using ROVs more and more everyday, mostly for search and rescue missions. The military use them to stalk enemy territory, patrol local harbors and explore ocean floors to detect environmental hazards. ROVs are particularly useful for search and rescue missions where the diving conditions are dangerous to people due to debris, low visibility, and long hours needed for missions.

ROVs are used extensively by the science community to study the oceans. ROVs used come in many different sizes depending on their application. From large and expensive ROVs used for deep sea applications to small ones with only a camera used for recording video and surveying the ocean floor. Sensor packages and payloads are usually tailor suited to the mission at hand, and ROVs are built for a specific type of science. Several deep sea animals have been discovered using ROVs. Oceonographic institutes such as the Monterey Bay Aquarium Research Institute (MBARI), Woods Hole Oceanographic Institute (WHOI), and University of Rhode Island / Institute for Exploration (URI/IFE) all use ROVs as part of their research. Most of these ROVs are large and expensive, in the millions of dollars.

There are also ROVs being used for educational outreach. These tend to be smaller, more hobby class ROVs. Several universities and institutes are trying to get student, many in middle and high school, to get interested in engineering. Having them work on ROVs is a good project to help them develop engineering skills.

ROVs are becoming increasingly popular in broadcasting. Their ability to be submerged for long periods of time in adverse conditions, makes them well adept at filming underwater documentaries. Small, maneuverable ROVs are particularly desirable in this industry.

There is another growing industry segment, which is the hobby ROV segment. Old and young people have built their own ROVs, usually made using PVC tubing and low cost electronics. These are very low cost and can usually go down to 50 – 100 feet and sometimes more (up to about 300 feet). These ROVs are mostly tested in calm waters and do not perform well with waves or currents.

6.5 - Sales and Marketing Strategies

Our marketing strategy is to appeal to two crowds, one being educational institutions and the other being marine researchers. Our ROV was designed in mind to be changed and altered and to be used as a teaching platform. Through this, we would highlight how versatile the frame is and how it allows for additions with the auxiliary port. We would also highlight how simple the electronics are and how much room there is for changes and improvements. In general, we want to stress how open-ended Proteus is that anyone could learn from it whether it is changing the frame orientation for less drag or designing and implementing some program that controls the ROV.

To appeal to these programs, we could offer demonstrations or even offer to let them borrow it for a short period of time. With how light and compact Proteus is, with would be easy to drive, ship or fly Proteus anywhere.

As for the sales team, it would be two people, that is how many people are required to operate Proteus in a demonstration but one of them could easily talk one the ROV is in the water, being controlled by the other salesperson. We would try to make sure people have a chance to operate the ROV to see how easy and fun it is.

If cost was a problem, we may rent out an ROV or create a sharing program where multiple schools could buy one and share it throughout the year.

Distribution would occur from Santa Clara and the product would be shipped with assembly instructions. This assembly would include putting the frame together, attaching the lights, camera and thrusters and installing the bottles in the holes in the frame. All permanent electronics would already be assembled but anything that can be unplugged will be unplugged and labelled.

6.6 – Competition

Table 30: Table of competition for selling Proteus

Product	**Price**	**Depth**	**Weight**	**Dimensions**	**Tether**	
Seabotix (vLBV300)	$88,000	1000 ft rating	40 lbs	24.6 x 15.4 x 15.4”	820 ft neutrally buoyant	Camera, lights, magnetometer, pressure sensor
VideoRay (Explorer X3)	$14,000	250 ft rating	10 lbs	12 x 8.5 x 9"	130 ft tether	Camera that can be remotely repositioned, lights, depth sensors
OpenROV	$895 for component kit, must build it	300 ft rating	5.5 lbs	5.9 x 7.9 x 11.8”		Camera and lights, laser distance calculator
Aquabotix-hydroview	$5,500 - 8000	150 ft	8 lbs		75 ft tether	lights, HD camera, controlled by iPad or laptop or included topside box, price based on sensors included

6.7 - Manufacturing Plans

For manufacturing, we would work at a small scale, having components made out of house but assembled by us. When designing the ROV, we tried to make sure every component on the ROV was off the shelf so that if anything broke, it was easy to replace. We would try to make a deal with the companies we got components from for Proteus to hopefully decrease price. Plus, we would be buying things in bulk rather than quantities of two or three. Right now, it would take five people 16 hours, or 80 man hours, to assemble the completed ROV and run simple test to make sure it is working correctly. If two people are out of town giving a demonstration that basically doubles the time required. It will take a year or so to start selling the product on a regular basis in by which we would stop assembling it ourselves and move towards hiring a team to decrease the time it takes.

6.8 - Product Cost and Price

The retail price of the system would be $14,000, we would need to sell at least two ROVs per year to make a profit due to annual costs. It would cost us less than $10,000 to purchase materials and components by buying in bulk, decreasing the price per unit to an assumed $8,000. At 80 man hours, paying $12 dls/hour the personnel cost for manufacturing one ROV would be $960. It takes an average of 4 hours to cut the frame material using a water jet cutter. At $0.20/min this comes out to a total cost of $48. Renting a small office/space in Santa Clara costs about $450/ month for 244 SQFT. The equipment cost would be an annual $600 payment to purchase/replace cutting tools, soldering irons and various tools. If one unit is sold, then there is a loss of $1000 but if two are sold, there is a profit of $3984. Below is a graph of profit if 13 units are made. If more are made, the annual cost of $600 dollars for tools will increase.

Our price puts us in the middle when it comes to the price of available ROVs in the market with the highest being $88,000 and the lowest being $900. This large range of price is due to the capabilities of the ROVs. The higher priced robots have the ability to reach greater depths, have repositionable cameras, etc. We know that Proteus is limited to 500 feet by the thrusters; however, you need to include the educational value. Students can learn from it, design around it and improve it. Out of all the competitors we mention, none of them have the ability to be built on or added to.

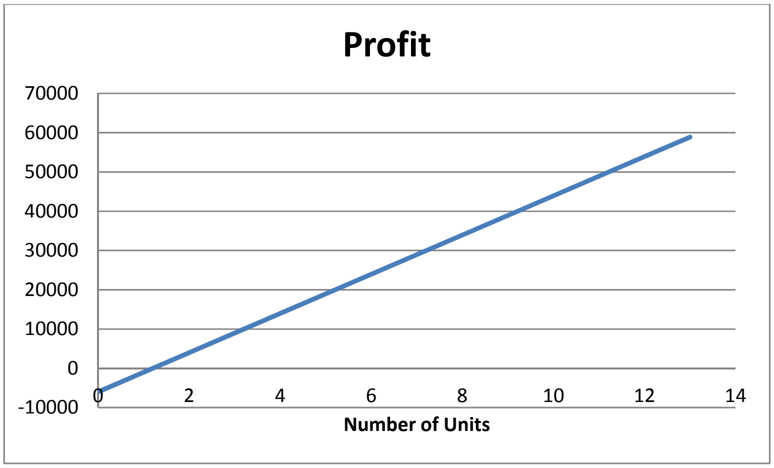


Figure 29: Graph of profit per unit.

Our most similarly priced competitor is the VideoRay Explorer X3, priced at $14,000. Proteus is rated to 500 ft versus 250 ft for the VideoRay, and comes with a 500 ft tether vs a 130 ft tether. The VideoRay however is much smaller and more lightweight than ours, coming in at 10 lbs vs our 50 lbs for the submersible. What we gain with ours is a magnetometer/heading sensor, as well as the ability to add auxiliary functionality and a greater payload capability (>5 lbs vs 2 lbs). Instead of continuous power, we provide power via a battery, limiting our drive time to approximately one hour per charge.

6.9 - Service and Warranty

We would offer a short warranty for the product, between 30 and 60 days, incase there are any manufacturing defects (leakage, shorts, in general does not work). After 60 days, any damage on the ROV is due to use and is the operators fault, any manufacturing problems would have surfaced before then. With this, we will include detailed instructions and precautions to decrease the change of the operator causing lasting harm to the ROV. When it comes to the Renting program, there would be some form of a deposit and for the sharing program, the ROV could be evaluated every once and a while to see who has done damage and who should be replacing what.

6.10 - Financial Plan

We would need to take out a small loan for initial costs for space and tools, but if we sell units, we will be able to quickly pay it back.

Chapter 7 - Project Summary and Conclusion

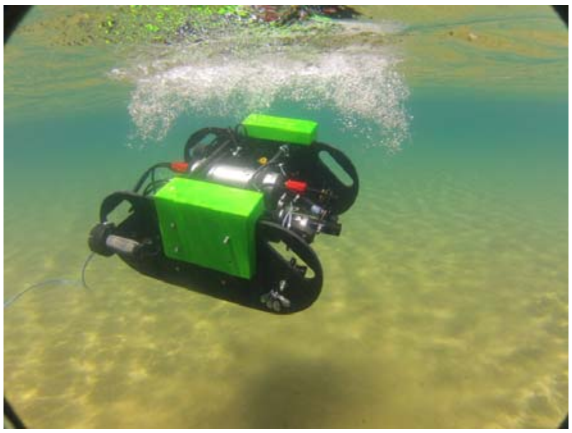


Figure 30: Testing Proteus *(Photo: Jorge Guerra)*

7.1 - Summary of Work

The goal of this project was to design and build a low cost, easy to use, portable, safe, and reliable ROV capable of being used for scientific research, while being run by students. An in depth survey was conducted with potential users, experienced users, and industry experts in order to understand what was required in an ROV and what to keep in mind when developing one. We developed several sketches of possible designs for our ROV, and built several prototypes, getting feedback from our customer on each design. We tested resulting components of our system when appropriate before integrating the full system, ensuring a successful build.

The system was used for real research missions in Lake Tahoe at the end of the year, validating the success of the ROV.

In the end, the ROV Proteus met almost all of the requirements we set forth in the beginning. We designed and built an ROV that can be used by students. Proteus reached a depth of 75 feet while sending depth, temperature and heading readings as well as the live feed from the camera to the topside console. The camera and lights worked correctly. The thrusters were able to maneuver the ROV around in the water after they were repositioned. Although some of the sensors malfunctioned, they can be easily replaced. The ROV was deployable by two people in ten minutes when our requirement was three people in fifteen. It was under the required weight of 75 lbs with an actual weight of less than fifty. It is greater than 62 linear inches by 3 inches, but this increased size was considered necessary in order to accommodate all required components. The electronics bottle leaks; we believe this because the bottle was machined and not within tolerance. Like the sensors, this can be replaced.

7.2 - Future Work

There is room for future work left for our ROV. First of all, there are some improvements to the current design. The most critical improvement is to fix the leak in the main electronics bottle. It must be determined where the leak is developing before we know how to stop it. There are a few sensor readings that were off and need to either be calibrated or have the sensors replaced. The temperature sensor consistently gives incorrect readings in the hundreds of degrees Celsius when in the water. This could be due to water shorting some pins. The compass was originally mounted in the sensor package that was encased in potting compound. The compass however, stopped working when it was fully encased. The CMPS10 compass is notorious for acting faulty when it comes in contact with certain substances. For now, there is one mounted inside the main electronics bottle. The compass still needs to be calibrated which can be done using an Arduino code.

Right now, the lights can be switched on or off using a mechanical relay. Ideally, the lights would have dimming functionality. Students can work to achieve this functionality.

There is also the possibility for students to redesign the frame as needed. The main change would be to improve the water flow through the ROV by removing unnecessary frame material in the front and back plates of the frame. This would reduce weight and would help reduce drag, improving the driving dynamics of the ROV.

Part of our design specifications was to add at least one auxiliary port with lines connecting directly to the main electronics bottle. This allows students to build upon our ROV by adding more functionality. There are a total of 14 lines available in the main electronics bottle that can be used for peripheral equipment. Students at SCU can develop sensing packages for specific science missions, as well water samplers, a high definition camera or a laser range finder. There are many possibilities for peripheral equipment. Our advisor Dr. Kitts has been in contact with a colleague in Villanova University who is interested in creating their own marine robotics program. In order to expose themselves to this field, they have expressed interest in developing a peripheral for our ROV. This could consist of a manipulator that is mounted on our ROV. And thanks to the modular design of our frame, mounting it would be simple.

Down the line, we hope that students will study and characterize the system dynamics and eventually use this information to develop autonomous controllers for the ROV. This could be in the form of heading control, where the ROV can maintain a certain heading while driving so the scientist can spend more time on the science rather than knowing where they are. The ROV could also be reproduced fairly easily and used to test multi robot control techniques, which are commonly used in the RSL.

There is another idea we have talked about with Dr. Kitts, and that is the possibility of having the ROV become a “product” that the RSL sells. This could mean having a few of them built for a specific purpose and sold to programs that are in the market for an ROV, or it could also mean selling the services. The ROV we have could be disassembled, flat packed and shipped to wherever is needed, and operators could fly out and help manage the ROV deployments. Santa Clara University could use the ROV to help other universities or research programs looking for a low cost, high performance ROV.

7.3 - Conclusion

In conclusion, we designed and built a functioning ROV that can be used for marine research or as a test bed/learning tool for universities including Santa Clara University. There are adjustments to be made to Proteus; however, they can be made by future students as a way to learn more about ROVs or can be changed by the students in an attempt to make it better.

Bibliography

Buecher, David. *Design and Manufacture of a Low Cost Underwater Remote Operated Vehicle (ROV)*. Thesis. Santa Clara University, School of Engineering, 2004. Print.

Harrington, J.V.; Parker, A.M.; Waligora, T.M.; Walworth, M., "Design of an inexpensive waterproof housing," *OCEANS 2003. Proceedings* , vol.5, no., pp.SP2880,SP2887 Vol.5, 22-26 Sept. 2003

"Human Settlements On The Coast". *UN Atlas of the Oceans.* Web. 23 Oct 2013.

Murphy, Robin R., et al. "Marine Heterogeneous Multirobot Systems at the Great Eastern Japan Tsunami Recovery". *Journal of Field Robotics* 29.5 (2012): 819-831. EBSCO. Web. 16 Oct 2013.

Ocean." National Oceanic and Atmospheric Administration. Web. 14 Nov. 2013. <http://www.noaa.gov/ocean.html>.

"Ocean Resources - MarineBio.org". MarineBio Conservation Society. Web. Monday, November 25, 2013. <http://marinebio.org/oceans/ocean-resources.asp>.

Scaradozzi, David, , Giuseppe Conte, and Laura Sorbi. "Assisting Micro-ROV Operators During Surveys in Fragile Environments." *Sea Technology* 53.4 (2012): 56-60. *Santa Clara Library Applies Science & Technology*. Web. 17 Oct. 2013.

"SeaBotix Inc. - VLBV300." *SeaBotix Inc.* SeaBotix Inc., n.d. Web. 29 May 2014. <http://www.seabotix.com/products/vlbv300.htm>.

Vessels and Vehicles - R/V Zephyr." Vessels and Vehicles - R/V Zephyr. Monterey Bay Aquarium Research Institute, 07 Mar. 2013. Web. 14 Nov. 2013. http://www.mbari.org/dmo/vessels_vehicles/Doc_Ricketts/specifications.html>

"VideoRay Explorer ROVs." *VideoRay Explorer ROVs*. VideoRay, n.d. Web. 29 May 2014. <http://www.videoray.com/homepage/economy-configurations/videoray-explorer.html#1296-Explorer>.

Vlahos, Michael. *PVCROV2: Improved Low-Cost Rugged ROV for Control Theory Experiments.* Graduate Capstone Report. Advisor: Kitts, C. Santa Clara University, 2014

Wareham, Martin. "ROVs Continue to Develop Capacity for Deepwater Operations." Offshore Magazine Apr. 2013: 108-09. EBSCO. Web. 17 Oct. 2013.

Weast, A., Laurence, A., Lwin, C., Cook, J., Wigle, T. *The Triton Remotely Operated Vehicle (ROV)*. Undergraduate Thesis. Advisors: Ota, J. and Hight, T. Santa Clara University, 1999.

Appendix A - Customer Raw Data

Below is how we decided on the requirements for Proteus. We had ten categories preference fell under. We included what kind of person made which preference as a way to weight the importance of each with the key customer being the most important/influential.

KC	Key customer
EU	Experienced user
PU	Potential user
IE	Industry Expert

Attachments	Performance	Operation	User interface
Good camera, video is critical (EU, PU)	Enough power to drive required payload (PU)	1-3 people operating (KC)	Live video feed (EU, PU)
Manipulator (not critical) (KC, PU, EU)	Follow compass heading (PU)	Deploy from shore, dock, boat (KC, PU)	Data overlay of heading, depth, date, time (PU)
Any TBD instruments (KC)	Thrust lines through center of gravity (IE)	Multiple missions per day (KC, PU)	Video split and automatic video recording (EU)
Laser scaling system (EU, PU)	Ability to hover (depth lock) (PU)	Operate for more than 1 hr and less than 8 hr at a time (KC)	Analog/digital control (KC)
Positioning data (x,y,z coordinates) (PU)	Variable, easy ballasting (EU, PU)	Used in lakes, estuaries (low current), MBARI test tank (KC, PU)	
Water profiler (PU)	Trim (thrust to compensate buoyancy) (EU)	Multiple operation modes (analog/digital/no compass/no depth/etc) (KC)	
	500 ft depth rating (KC, EU, PU)	Top side option for battery/direct connection (KC)	
	Tether management on ROV side for efficient use of power and good driving dynamics (IE)		

Portability	**Purpose**	**Simplicity**	**Safety**
Transported in the back of a car (KC)	License ROV to the RSL and work with other universities (KC)	Inexperienced user can understand and operate with not too much effort (KC)	Low voltage (<40V) (KC)
ROV can be handled and moved by 1 person (KC)	Used as an educational test bed for future students and research into multi robot systems (KC)	Easy to maintain / built in diagnostics testing equipment (KC)	Shrouded propellers to protect tether (KC)
Small, lightweight (KC)	Incorporate ROV into MECH 180: Marine Ops class (KC)	Tether management system (EU, PU, IE)	Small, light for safety (KC)
Handles for carrying ergonomics (EU)	Must be maintained by the Robotic Systems Lab (KC)	Well documented parts list (EU)	Carrying handles (EU)
Winch hook (KC, EU)	Backup to Triton ROV (KC)	15 min set up time (KC)	Have electrically safe system (EU)
Easy to get in/out of water and make desired changes (KC)	Possibly using the ROV for a control systems class lab (KC)	Minimal electronics on ROV (IE)	
15 min set up time (KC)	Serve as a student development project (KC)	Minimal number of plugs in ROV (IE)	

Robustness	**Cost**
Not many extraneous parts (KC)	$10,000 to $15,000 not including labor (KC)
Strong and durable (KC)	
Good marine plugs (EU)	
Multiple operation modes (analog/digital/no compass/no depth/etc) (KC)	
Minimal electronics on ROV. Keep most of the controlling topside (IE)	
Designing for robustness will save you in the long run (IE)	
Good wiring cable management (KC)	

Appendix B - Product Design Specifications

Below is a table of the different parameters and goals we had for our design for Proteus. We were able to stay within the design target range for almost all characteristics. We were unable to stay within the target for the linear size (65 inches and we were shooting for 62), the fabrication of the frame had to be done by MBARI when we had hoped to do all manufacturing in our machine shop with the laser cutter.

Characteristic/ parameter	**Parameter Units**	**Design Criticality**	**Design Target**	**Benchmark 1 Range**	**Benchmark 2 Range**	**Benchmark 3 Range**
Price	USD	1	<$15,000	~$300,000	~$1,500	~$88,000
Size	Linear inches	1	62	92	36 w/ arms collapsed	55.4
Weight	Pounds	1	<100	~250	~12	40
Depth Rating	Feet	1	<500	1,000	50	1,000
Supply voltage	Volts	1	<48 DC	120-240 AC	11.1 DC	100-240 AC
Deploy-ability	# of people	1	2-3	5	2	2-3
Run time	Minutes	2	~60	Continuous supply	~120	Continuous supply
Buoyancy source	Material	2	Foam	Syntactic Foam	Sealed ABS	Foam
Frame Material	Material	2	HDPE	Aluminum	PVC	HDPE
Forward thrust to weight	Kgf/kg	3	0.1-0.3	~0.26	~0.1	0.99
Sensors	Magnetometer & depth sensor	1	Magnetometer	Magnetometer, depth	Magnetometer	Magnetometer, depth
Modularity(auxiliary port)	Yes/no	1	Yes	Yes	no	Yes
Power	On-board/ topside	1	On-board	topside	On-board	topside
Manufacturing	Facility	3	SCU machine shop	SCU/DOE shop	SCU maker Lab	Seabotix
Transportation	Car/SUV/ trailer	1	SUV	Trailer	SUV	SUV
Maintenance	Student/outsource	1	students	Outsourced (DOE)	students	Outsourced (seabotix)
Camera	Yes/no	1	Yes	Yes	no	Yes
Fabrication process	Machine, weld, form	2	Machine, laser	Machine, weld	Light Fabrication	Machine, thermoform
Thruster configuration	Number, direction	3	2 fwr, 2 vertrans	2 fwr, 2 vertrans	2 fwr, 2 vertical	4 vectored, 2 vertical
Environment	Fresh/salt water	2	Fresh and salt	Fresh and salt	Fresh	Fresh and salt

Appendix C - Bill of Materials and Budget

Below is the bill of materials for out project and our total budget.

Mini ROV	Budget							
Subsystem	Item	Description	#	Vendor	Cost/ part	Bought /Donated	Estimated	Actual w/ Donations
Frame & Flotation								
	Frame	48" x 48", 1/2" thick HDPE	1	McMaster	$136.00	-	$136.00	$136.00
	Polyurethane Foam	1 ft^3	2	General Plastics	$180.00	Donated	$360.00	$0.00
	Foam sealant	Paint	1	Home Depot	$50.00	-	$50.00	$50.00
	Subsystem Total						$546.00	$186.00
Sensing								
	Camera	ROVSCO RD400	1	ROVSCO	$690.00	-	$690.00	$690.00
	Magnetometer	CMPS 10	1	Devantech	$37.00	-	$37.00	$37.00
	Pressure sensor	MLH250PSL09A	1	Digi Key	$131.44	-	$131.44	$28.00
	Temperature sensor	TMP36	1	TI	$1.50	-	$1.50	$1.50
	Humidity Sensor	HIH-4021-003	1	Digi Key	$28.76	-	$28.76	$28.76
	Lights	ROVSCO SeaDragon	2	ROVSCO	$650.00	-	$1,300.00	$1,300.00
	Subsystem Total						$2,188.70	$2,085.26
Electronics								
	Motor Drivers	RobotEQ SDC2130	2	RobotEQ	$175.00	-	$350.00	$350.00
	Micro controler	Arduino Mega	2	SCU	$35.00	Donated	$70.00	$0.00
	RS485 Transceiver	MAX488E	2	MAXIM	$1.00	Donated	$2.00	$0.00
	Electronics housing	Al bottles + delrin end caps	2	DOE	$850.00	Buy/don	$1,700.00	$1,300.00
	Relay	FRS08	1	HSC	$1.75	-	$1.75	$1.75

	Solid State Relay	Teledyne	1	Jameco	$5.00	-	$5.00	$5.00
	Thrusters	Seabotix thruster	4	Seabotix	$600.00	-	$2,400.00	$2,400.00
	Subsystem Total						$4,528.75	$4,056.75
Power								
	DC-DC Converter	12V converter	1	Jameco	$17.00	-	$17.00	$17.00
	Batteries	22.2 V 6S lipo (10 Ah)	1	Quadrocopter	$245.00	-	$245.00	$245.00
	Subsystem Total						$262.00	$262.00
Wiring & Misc								
	Umbilical	500 ft CAT5e	1	Monoprice	$70.00	-	$70.00	$70.00
	Marine Plugs	Subconn connectors	1	Subconn	$1,700.00	-	$1,700.00	$1,700.00
	Potting Compound	2131 Scotchcast	1	3M	$92.00	-	$92.00	$92.00
	Switches & fuse	Variety	1	Anchor Electronics	$30.00	-	$30.00	$30.00
	Connectors	Variety	1	HSC	$30.00	-	$30.00	$30.00
	Mounts	Variety	1	Home Depot				
	Wire	Variety	1	HSC	$25.00	-	$25.00	$25.00
	LCD Screen	4x20	1	Amazon	$12.00	-	$12.00	$12.00
	Hard case	Topside Box	1			-		
	Subsystem Total						$1,862.00	$1,862.00
Tahoe Trip								
	Lodging	nights	3		$300.00	-	$900.00	$900.00
	Food/drink	4 days	1		$400.00	-	$400.00	$400.00
	Car Expenses	161 mi + permits	1		$350.00	-	$350.00	$350.00
	Subsystem Total						$1,650.00	$1,650.00
Total							$11,037.45	$10,102.01

Appendix D - Mass

Watching the mass of Proteus was extremely important because of our customer requirements. Our estimated total mass was 46.88 lbs and when we weighed Proteus, it was 49 lbs, below our requirement of less than 75 lbs.

Subsystem	Item	Description	#	Mass/item (lbs)	Total Mass(lbs)	Source
Structure						
	Foam	6" x 10" x 18" (24 lb/ft^3)	1	-	-	Estimate
	Frame	24" x 36", 1/2" thick HDPE (35 lb/ft^3)	1	14	14	Estimate
	Bottle housing	4.25 in diameter Al	2	1	2	Estimate
	End Caps	Delrin	4	1	4	Estimate
	Foam sealant	Zolatone and primer	1	0.5	0.5	Estimate
	Nuts and bolts	Pack	1	2	2	Estimate
	Marine plugs	Subconn connectors	1	3	3	Estimate
	Potting Compound	Marine epoxy	1	3	3	Estimate
	Mounts		1	2	2	Estimate
	Winch hook		1	1	1	Estimate
	Subsystem Total				31.5	

Sensing						
	Camera	ROVSCO RD400	1	0.33	0.33	Spec
	Magnetometer	CMPS 10	1	0.02	0.02	Spec
	Pressure sensor		1	0.25	0.25	Estimate
	Temperature sensor		1	0.02	0.02	Estimate
	Conductivity Sensor		1	0.05	0.05	Estimate
	Lights	ROVSCO SeaDragon	2	0.48	0.96	Spec
	Subsystem Total				1.63	
Electronics						
	Motor Drivers	RobotEQ SDC2130	2	0.2	0.4	Spec
	Microcontroler	Arduino Mega	1	0.12	0.12	Spec
	Video signal amplifier		1	0.02	0.02	Estimate
	RS485 converter	Hossen MAX3485	1	0.0185	0.0185	Spec
	Mounting board		1	0.3	0.3	Estimate
	Wiring		1	0.1	0.1	Estimate
	Subsystem Total				0.9585	
Power						
	Thrusters	Seabotix thruster	4	1.4	5.6	Spec
	Batteries	22.2 V 6 cell lipo (7500 Ah)	1	2.2	2.2	Spec
	Subsystem Total				7.8	
Other						
	Boards		1	5	5	Estimate
	Subsystem Total				5	
	ROV Total				**46.8885**	

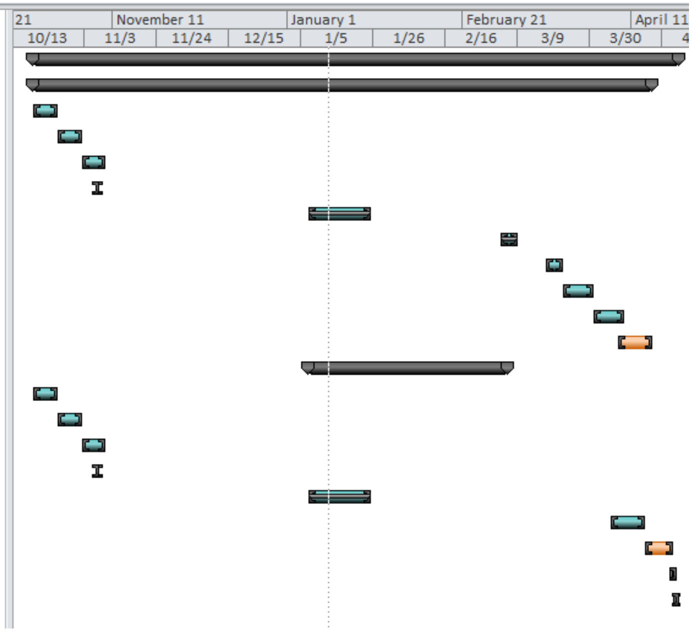
Appendix E - Power

We were limited in power by size requirements for the ROV so below is a breakdown of the current, voltage and power for each component.

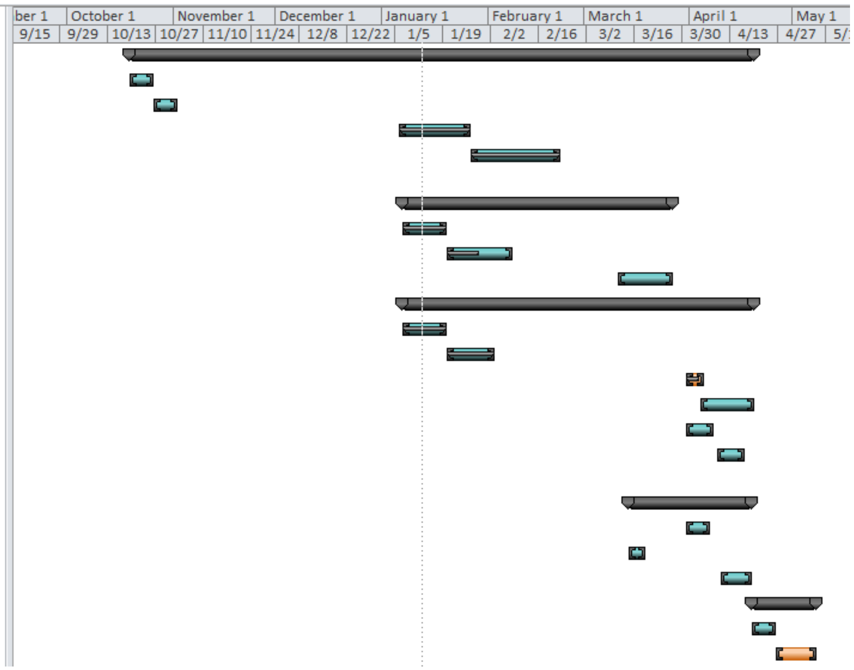
Component	Current (A)	Voltage (V)	Source	Power (W)	Idle (W)	Dive/Surface (W)	Driving (W)
Z Thrusters (2)	8.5	19.1	Spec/test	162.35	0	162.35	81
X Thrusters (2)	8.5	19.1	Spec/test	162.35	0	0	162.35
Camera (ROVSCO RD400)	0.1	24	Spec	2.4	2.4	2.4	2.4
Lights (2) (ROVSCO SeaDragon)	1.44	24	Spec	34.56	34.56	34.56	34.56
Magnetometer (Devantech CMPS10)	0.025	5	Spec	0.125	0.125	0.125	0.125
Pressure Sensor	0.05	5	Spec	0.25	0.25	0.25	0.25
Temperature Sensor (LM35)	0.05	5	Spec	0.25	0.25	0.25	0.25
Conductivity Sensor	0.05	5	Spec	0.25	0.25	0.25	0.25
Arduino Mega	0.05	12	Spec	0.6	0.6	0.6	0.6
Motor Driver (RoboEQ SDC2130)	0.1	24	Spec	2.4	2.4	2.4	2.4
RS485 converter	0.025	3.3	Spec	0.0825	0.0825	0.0825	0.0825
Total (W)					40.9175	203.2675	284.2675
	Watt hours required	**Battery pack voltage**	**Amp hours required**				
Worst case (Driving) for 1 hr	267.595	22.2	12.0538288288288				

Appendix F - Timeline

Task Name	Duration	Start	Finish
Structure and Flotation	**134 days**	**Sat 10/19/13**	**Thu 4/24/14**
Frame	**128 days**	**Sat 10/19/13**	**Wed 4/16/14**
Brainstorm	6 days	Sat 10/19/13	Fri 10/25/13
Preliminary Design	6 days	Sat 10/26/13	Fri 11/1/13
CAD Drawings	6 days	Sat 11/2/13	Fri 11/8/13
Build Prototype	1 day	Wed 11/6/13	Wed 11/6/13
Revised Design	14 days	Tue 1/7/14	Fri 1/24/14
Order Test material	5 days	Tue 3/4/14	Sat 3/8/14
Test material	5 days	Mon 3/17/14	Fri 3/21/14
Order final material	7 days	Sat 3/22/14	Sun 3/30/14
Machine	7 days	Mon 3/31/14	Tue 4/8/14
Assemble	8 days	Mon 4/7/14	Wed 4/16/14
Flotation	**42 days**	**Tue 1/7/14**	**Wed 3/5/14**
Brainstorm	6 days	Sat 10/19/13	Fri 10/25/13
Preliminary Design	6 days	Sat 10/26/13	Fri 11/1/13
CAD Drawings	6 days	Sat 11/2/13	Fri 11/8/13
Build Prototype	1 day	Wed 11/6/13	Wed 11/6/13
Revised Design	14 days	Tue 1/7/14	Fri 1/24/14
Machine	7 days	Sat 4/5/14	Mon 4/14/14
Assemble	6 days	Tue 4/15/14	Tue 4/22/14
Mount Brackets	2 days	Tue 4/22/14	Wed 4/23/14
All Hardware Mounted	2 days	Wed 4/23/14	Thu 4/24/14



Task Name	Duration	Start	Finish
Electronics	**130 days**	**Sat 10/19/13**	**Sat 4/19/14**
Brainstorm	6 days	Sat 10/19/13	Fri 10/25/13
Preliminary Design	6 days	Sat 10/26/13	Fri 11/1/13
Order Parts (Camera, lights)	16 days	Mon 1/6/14	Sun 1/26/14
Test/integrate Camera and Lights	20 days	Mon 1/27/14	Fri 2/21/14
Sensing	**57 days**	**Tue 1/7/14**	**Wed 3/26/14**
Buy Parts	10 days	Tue 1/7/14	Sun 1/19/14
Integrate with Arduino	15 days	Mon 1/20/14	Fri 2/7/14
Bench Test	12 days	Tue 3/11/14	Wed 3/26/14
Power	**74 days**	**Tue 1/7/14**	**Sat 4/19/14**
Finalize Research	10 days	Tue 1/7/14	Sun 1/19/14
Order Battery/Converter	11 days	Mon 1/20/14	Sun 2/2/14
Test/integrate	5 days	Mon 3/31/14	Fri 4/4/14
Electronics Bench Test	12 days	Fri 4/4/14	Sat 4/19/14
Full Electronic System	6 days	Mon 3/31/14	Mon 4/7/14
Integrate Tether Communication	6 days	Wed 4/9/14	Wed 4/16/14
Wiring	**26 days**	**Fri 3/14/14**	**Fri 4/18/14**
Wiring Diagram	6 days	Mon 3/31/14	Sun 4/6/14
Order Plugs	3 days	Fri 3/14/14	Tue 3/18/14
Wire/pot	7 days	Thu 4/10/14	Fri 4/18/14
Testing	**13 days**	**Sat 4/19/14**	**Wed 5/7/14**
Full system test	6 days	Sat 4/19/14	Fri 4/25/14
water testing	9 days	Sat 4/26/14	Wed 5/7/14



Appendix G - Calculations

Below is the excel sheet that was used to calculate the amount of flotation required to keep Proteus slightly positively buoyant. The first table is the breakdown of weight for each component with the total being 16.83 lbs.

Component	Volume	Weight	Bouyancy	Apparent Weight	Other Mass	
Bottle	198.8039	9.1585	7.1821889	1.9763111	1.3	
Thrusters	40.88979	5.6	1.4772255	4.1227745	9	
Lights	14.43169	0.96	0.5213737	0.4386263		
						Total
						16.837712

The second table is the flotation properties and a list of a variety of weights of the ROV and how much flotation is required at that weight. We initially used the total weight found above to determine the size of flotation required; however, we overestimated how heavy the components were so it was too positively buoyant. We then took that flotation off and weighed Proteus in the water and found the actual weight and used that to find that it required about 0.1425 cubic feet of flotation.

Foam Properties	**R-3315**				**Bouyancy per volume**
Density (kg/m3)	240				0.036127
Compressive (psi)	679	*350 psi liquid penetration resistance			
					Water Density
Psi at 500ft	216.6				62.42796
ROV Weight lbs	**kg**	**Volume R-3315 (m3)**	**ft^3**		
0	0	0	0		
5	2.26796	0.0029842	0.1053846		
10	4.53592	0.0059683	0.2107693		
15	6.80388	0.0089525	0.3161539		
20	9.07184	0.0119366	0.4215386		
25	11.3398	0.0149208	0.5269232		

Appendix H - PDS and Experimental Protocol

Description	Baseline
Cost of parts	< $15,000
Dimensions	~ 62 linear inches (L+W+H)
Mass	< 75 lbs
Deployment personnel	2 people
Portability	Entire system fit in a personal vehicle
System Voltage	< 48 Volts
Depth rating	~ 500 ft
Battery life	> 1 hour
Buoyancy of ROV	Slightly positive buoyancy
Payload	~ 5 lbs
Auxiliary port	1 auxiliary port connected to microcontroller
Camera	Live feed
Pressure Sensor	$\pm$ 6 ft depth accuracy
Compass	$\pm 30°$ heading accuracy
Set up time	~ 15 min

Test	Location/Time	Equipment	Accuracy	Trials	Expected outcome	Assumptions	Man hrs
Pressure sensor	MBARI. Full ROV test, May 1	ROV, Control Box, MBARI tank	±6 ft	3	Record 33 ft depth at bottom of tank	Pressure to depth conversion at 14.7 PSI per 33 ft	5
Buoyancy	MBARI. Mar 1	ROV, MBARI test tank	-	3	ROV will slowly rise when turned off at depth		2
Thruster capabilities	Off campus pool. Completed	Stopwatch, body, prototype ROV	±0.5 ft/sec	4	Can pull 180lbs through water with ease	Assume higher drag and weight than actual system	2
Compass heading	AMES. April 21	Arduino, potted compass, computer	±30°	4	We identify separate quadrants	We know reference directions	2
Waterproof bottle	MBARI. April 23	Test tank	-	1	Dry insides	Leaving bottle in tank for 0.5 hrs will mean waterproof	3
Dimensions	RSL. April 24	Measuring tape, ROV	±1 in	3	62 linear inches		1
Mass	RSL. April 24	Spring scale, ROV	±2 lbs	3	<75 lbs		1
Payload	Off campus pool. April 28	Pool, ROV, weights	-	3	Can move 5 then 10 lbs in water		2
Portability	SCU. April 25	ROV, car	-	1	Can fit system into personal vehicle		1
Set up time	SCU. April 25	ROV, Car, control box	±10 sec	2	Get ROV out of car, set up, fully operational in <15 min by 2 people		2

H.1 - Experimental Protocol

Pressure Sensor: Put it down to a known depth and see the response of the sensor.

Buoyancy: We want the ROV to be positively buoyant in case we lose power or it gets disconnected. That way it will float and we will not lose the ROV. Tested by stop controlling at bottom of tank and wait for it to rise.

Compass heading: We want accuracy of plus minus 30 degrees. We want to know we are around the correct quadrant. We will test with the sensor mounted on the ROV with and without running motors to see the effect of it.

Waterproof bottle: By placing a paper towel in the bottle and dropping bottles to depth we can see if there is a leak when we open it up.

Payload: We want to add weight so that we had the opportunity to add an auxiliary attachment in the future. We must be sure that we can still pull the weight.

Portability: We want to be able to transport the ROV in a personal vehicle (sedan style). We want to fit the ROV, control box, tether and any additional equipment and still have room for 2 to 3 people to deploy.

Set up time: Get ROV equipment out of car, set up control box, monitor. Turn on ROV and have it fully operating in less than 15 minutes. There will be 2 people conducting the deployment. If 2 people can not deploy under 15 minutes, we will try with 3 people to see how much it speeds up.

Appendix I - Code for Electronics

I.1 - Topside Control Code

```
/*****************************************************************
*********************
Author: Jorge Guerra
Date  : May 21, 2014

Based on work by Chase Trafficanti and Mike Vlahos. This is the software to
interface with
and control the Mini ROV - Proteus in a topside control box.
******************************************************************
********************/
#include "RSLpacket.h"
#include <SoftwareSerial.h>
#include <LiquidCrystal.h>

//40 40 62 7A 00 08

// Define Constants

#define SSerialRX        10     // Software Serial RX
#define SSerialTX        11     // Software Serial TX

SoftwareSerial RS485Serial(SSerialRX, SSerialTX); // Name software serial port
RS485Serial

// initialize the LCD screen library with the numbers of the interface pins
LiquidCrystal lcd(2, 3, 4, 5, 6, 7);


//set up ROV serial adress and comm port
/*
a:     97
b:     98
c:     99
d:     100
*/
RSLpacket rslHw(RS485Serial, 122); //this arduino is adress 'z' or 7A in HEX

char pitch[4], roll[4], yaw[4], batt[4], tempc1[4], tempc2[4], watertemp[4],
pressure[4], humidity[4];   // creates character arrays to store data received
```

```
int k,y;

const int Joy1y = A0;        // joystick 1 Y axis
const int Joy1x = A1;        // joystick 1 X axis

const int Joy2y = A3;        // joystick 2 Y axis
const int Joy2x = A4;        // joystick 2 X axis

const int camera = 35;  //Switch to turn camera on/off
const int lights = 31;  //Switch to turn lights on/off
const int dimmer = 33;
const int aux1 = 37;
const int pshbtn1 = 39;
const int pshbtn2 = 41;

char commands[12];
char array[20];


void setup()
{
    // set up the LCD's number of columns and rows:
  lcd.begin(20, 4);

  Serial.begin(38400);

  RS485Serial.begin(38400);

  //Welcome message
  lcd.setCursor(1, 0);
  lcd.print("Mini ROV - Proteus");
  lcd.setCursor(0, 2);
  lcd.print("Robotic Systems Lab");
  lcd.setCursor(9, 3);
  lcd.print("SCU");

  delay(3000);
//  lcd.clear();
//  lcd.setCursor(0, 0);
//  lcd.print((char)34);
//  lcd.print("The sea, once it");
//  lcd.setCursor(0, 1);
//  lcd.print("casts its spell,");
//
//  delay(5000);
//  lcd.clear();
```

```
//  lcd.setCursor(0, 0);
//  lcd.print("holds one in its net");
//  lcd.setCursor(0, 1);
//  lcd.print("of wonder forever.");
//  lcd.print((char)34);
//  lcd.setCursor(0, 3);
//  lcd.print("Jacques Y. Cousteau");
//
//  delay(5500);
  lcd.clear();


}




void loop()
{
   delay(50);
  // read and scale the two axes of each joystick:
  int y1reading = analogRead(Joy1y)/2;
  int x1reading = analogRead(Joy1x)/2;
  int y2reading = analogRead(Joy2y)/2;
  int x2reading = analogRead(Joy2x)/2;


//  Serial.println("one"); //Debug statement

  // Joystick 1 y forward, ROV forward
  if(y1reading > 259){
    commands[1] = y1reading-260;
    commands[0] = 0;
    commands[3] = y1reading-260;
    commands[2] = 0;
    //If going left on joystick, ROV turns left so slow down left motor and reverse
direction
    if(x1reading > 264){
      commands[1] = commands[1] - x1reading+260;
      if(commands[1] < 0){
        commands[0] = -commands[1];
        commands[1] = 0;
      }
    }
    //If right on joystick, ROV turns right so slow down right motor and reverse
direction
    else if(x1reading < 256){
```

```
      commands[3] = commands[2] - 251+x1reading;
      if(commands[3] < 0){
        commands[2] = -commands[3];
        commands[3] = 0;
      }
    }
  }

  //If joystick 1 y is back, ROV goes in reverse
  else if(y1reading < 252){
    commands[1] = 0;
    commands[0] = 251-y1reading;
    commands[3] = 0;
    commands[2] = 251-y1reading;
    //If joystick goes left, ROV turns right by slowing down motor then reversing
direction
    if(x1reading > 264){
      commands[2] = commands[2] - x1reading+260;
      if(commands[2] < 0){
        commands[3] = -commands[2];
        commands[2] = 0;
      }
    }
    //If joystick goes right, ROV turns left by slowing down motor then reversing
direction
    else if(x1reading < 256){
      commands[0] = commands[0] - 251+x1reading;
      if(commands[0] < 0){
        commands[1] = -commands[0];
        commands[0] = 0;
      }
    }
  }

  //No input on joystick 1 y axis
  else{
    commands[0] = 0;
    commands[1] = 0;
    commands[2] = 0;
    commands[3] = 0;
    //If joystick pushed left, ROV does pure rotation to left
    if(x1reading > 259){
      commands[0] = x1reading-260;
      commands[3] = x1reading-260;
    }
    //If joystick pushed right, ROV does pure rotation to right
```

```
    else if(x1reading < 252){
      commands[1] = 251-x1reading;
      commands[2] = 251-x1reading;
    }
  }

    // Joystick 2 y forward, ROV up
  if(y2reading > 259){
    commands[4] = y2reading-260;
    commands[5] = 0;
    commands[6] = y2reading-260;
    commands[7] = 0;
    //If going left on joystick, ROV rolls left so slow down left motor and reverse
direction
    if(x2reading > 259){
      commands[4] = commands[4] - x2reading+260;
      if(commands[4] < 0){
        commands[5] = -commands[4];
        commands[4] = 0;
      }
    }
    //If right on joystick, ROV rolls right so slow down right motor and reverse
direction
    else if(x2reading < 252){
      commands[6] = commands[6] - 251+x2reading;
      if(commands[6] < 0){
        commands[7] = -commands[6];
        commands[6] = 0;
      }
    }
  }

  //If joystick 2 y is back, ROV goes down
  else if(y2reading < 252){
    commands[4] = 0;
    commands[5] = 251-y2reading;
    commands[6] = 0;
    commands[7] = 251-y2reading;
    //If joystick goes left, ROV rolls right by slowing down motor then reversing
direction
    if(x2reading > 259){
      commands[5] = commands[6] - x2reading+260;
      if(commands[5] < 0){
        commands[4] = -commands[5];
        commands[5] = 0;
      }
```

```
        }
        //If joystick goes right, ROV rolls left by slowing down motor then reversing
direction
        else if(x2reading < 252){
          commands[7] = commands[7] - 251+x2reading;
          if(commands[7] < 0){
            commands[6] = -commands[7];
            commands[7] = 0;
          }
        }
      }

      //No input on joystick 2 y axis
      else{
        commands[4] = 0;
        commands[5] = 0;
        commands[6] = 0;
        commands[7] = 0;
        //If joystick pushed left, ROV does pure rotation roll to left
        if(x2reading > 259){
          commands[5] = x2reading-260;
          commands[6] = x2reading-260;
        }
        //If joystick pushed right, ROV does pure rotation roll to right
        else if(x2reading < 252){
          commands[4] = 251-x1reading;
          commands[7] = 251-x1reading;
        }
      }

     //Read if switch is turned on, if so, turn camera on. '0' means off, '1' means on.
      if(digitalRead(camera) == LOW){
         commands[8] = 0;
         Serial.print("off");
      }
      else{
         commands[8] = 1;
         Serial.print("on");
      }

       //Read if switch is turned on, if so, turn lights on. '0' means off, '1' means on.
      if(digitalRead(lights) == LOW){
         commands[9] = 0;
         Serial.println("  off");
      }
      else{
```

```
    commands[9] = 1;
    Serial.println("  on");
  }
//  Serial.println("two"); //Debug line
//Serial.print(byte(commands[0]));
//Serial.print("  ");
//Serial.print(byte(commands[1]));
//Serial.print("  ");
//Serial.print(byte(commands[2]));
//Serial.print("  ");
//Serial.print(byte(commands[3]));
//Serial.print("  ");
//Serial.print(byte(commands[4]));
//Serial.print("  ");
//Serial.print(byte(commands[5]));
//Serial.print("  ");
//Serial.print(byte(commands[6]));
//Serial.print("  ");
//Serial.print(byte(commands[7]));
//Serial.print("  ");
//Serial.print(byte(commands[8]));
//Serial.print("  ");
//Serial.print(byte(commands[9]));
//Serial.print("  ");
//Serial.println(x1reading);


  //send commands message to ROV adress "a". commands has a length of 6 bytes
  rslHw.sendMessage(0x61,commands,10);
  delay(100);
//  Serial.println("three"); //debug line

  if (rslHw.available()>0 )
   {
     //get data from ROV
     //rslHw.getMessage reads data on a software serial port and saves it
     //to an object called message
//      Serial.println("four"); //Debug line
     rslHw.getMessage();
     //  Read untill we are out of bytes or get a proper message.
//      Serial.println("4"); //Debug line
     while((rslHw.ReadFail!=0) && (rslHw.available()) )   {rslHw.getMessage();}
     //ONLY send data if a proper message was recieved
     if(rslHw.ReadFail==0)
       {
//         Serial.println("five"); //debug line
```

```
//Get data from packet and display it on an LCD screen
//Zero out character arrays to be displayed on LCD screen so that characters do
not overlap
  k=0;
  y=0;
  int i;
  for (i=0; i<3; i++){
  yaw[i] = 0x20;
  pitch[i] = 0x20;
  roll[i] = 0x20;
  batt[i] = 0x20;
  tempc1[i] = 0x20;
  tempc2[i] = 0x20;
  watertemp[i] = 0x20;
  pressure[i] = 0x20;
  humidity[i] = 0x20;
  }

 //Data string is set up to be numbers and commas "123,345,3,23,4,"
 //So read until you find a comma and save it into the appropriate array,
 //then go the next data set
  while(byte(rslHw.message[k]) != 0x2C){
   yaw[y]=byte(rslHw.message[k]);
   k++;
   y++;
  }

  k++;
  y=0;
  while(byte(rslHw.message[k]) != 0x2C){
   pitch[y]=byte(rslHw.message[k]);
   k++;
   y++;
  }

  k++;
  y=0;
  while(byte(rslHw.message[k]) != 0x2C){
   roll[y]=byte(rslHw.message[k]);
   k++;
   y++;
  }

  k++;
  y=0;
  while(byte(rslHw.message[k]) != 0x2C){
```

```
  batt[y]=byte(rslHw.message[k]);
  k++;
  y++;
}

k++;
y=0;
while(byte(rslHw.message[k]) != 0x2C){
  tempc1[y]=byte(rslHw.message[k]);
  k++;
  y++;
}

k++;
y=0;
while(byte(rslHw.message[k]) != 0x2C){
  tempc2[y]=byte(rslHw.message[k]);
  k++;
  y++;
}

k++;
y=0;
while(byte(rslHw.message[k]) != 0x2C){
  watertemp[y]=byte(rslHw.message[k]);
  k++;
  y++;
}

k++;
y=0;
while(byte(rslHw.message[k]) != 0x2C){
  pressure[y]=byte(rslHw.message[k]);
  k++;
  y++;
}

k++;
y=0;
while(byte(rslHw.message[k]) != 0x2C){
  humidity[y]=byte(rslHw.message[k]);
  k++;
  y++;
}

} //void loop
```

```
//        Serial.println("six"); //Debug line

   }

//   Serial.println("seven"); // Debug line

   //Print data on LCD screen
   lcd.setCursor(0, 0);
   lcd.print("Yaw: ");
   lcd.print(yaw);
   lcd.setCursor(8, 0);
   lcd.print((char)223);
   lcd.setCursor(10, 0);
   lcd.print("Ptch: ");
   lcd.print(pitch);
   lcd.setCursor(19, 0);
   lcd.print((char)223);
   lcd.setCursor(0, 1);
   lcd.print("Rll: ");
   lcd.print(roll);
   lcd.setCursor(8, 1);
   lcd.print((char)223);
   lcd.setCursor(10, 1);
   lcd.print("Tmp: ");
   lcd.print(watertemp);
   lcd.setCursor(18, 1);
   lcd.print((char)223);
   lcd.print("C");
   lcd.setCursor(0, 2);
   lcd.print("Z: ");
   lcd.print(pressure);
   lcd.setCursor(6, 2);
   lcd.print("ft");
   lcd.setCursor(10, 2);
   lcd.print("Vlt: ");
   lcd.print(batt[0]);
   lcd.print(batt[1]);
   lcd.print(".");
   lcd.print(batt[2]);
   lcd.setCursor(19, 2);
   lcd.print("V");
   lcd.setCursor(0, 3);
   lcd.print("RH: ");
   lcd.print(humidity);
   lcd.setCursor(7, 3);
   lcd.print((char)37);
```

```
//   lcd.print("Ch1:");
//   lcd.print(tempc1);
//   lcd.setCursor(7, 3);
//   lcd.print((char)223);
//   lcd.print("C");
   lcd.setCursor(10, 3);
   lcd.print("MC: ");
   lcd.print(tempc2);
   lcd.setCursor(17, 3);
   lcd.print((char)223);
   lcd.print("C");

}
```

I.2 - ROV Code

```
/*****************************************************************
**********************
Author: Jorge Guerra
Date  : May 21, 2014

Based on work by Chase Trafficanti and Mike Vlahos. This is the software to
interface
sensing capabilities, thrust control, and video feed for use with the Mini ROV -
Proteus.
******************************************************************
*********************/
#include <Wire.h>
#include "RSLpacket.h"
#include <SoftwareSerial.h>

//40 40 62 7A 00 08

// Define Constants

#define SSerialRX        10     // Software Serial RX
#define SSerialTX        11     // Software Serial TX

#define ADDRESS 0x60          // Defines address of CMPS10

SoftwareSerial RS485Serial(SSerialRX, SSerialTX); // Name software serial port
RS485Serial
```

```
const int  length = 12;       // Length of the outgoing message
// control gain sacles from 255 to 240 to account for saturations, and then to +-
1000 for motorcontrollers
const int GAIN=(1000/255.0)*(240/255.0);

//set up ROV serial adress and comm port
/*
a:      97
b:      98
c:      99
d:     100
*/
RSLpacket rslHw(RS485Serial, 97); //this arduino is adress 'a' or 61 in HEX

//Declare global variables
const int temp_sensor = A0;  //Temperature voltage reading pin
const int pres_sensor = A1;  //Pressure voltage reading pin
const int RH_sensor = A2;   //Humidity voltage reading pin
const int camera_relay = 31;  //Pin to turn camera relay on/off
const int lights_relay = 33;  //Pin to turn lights relay on/off
const int STR_LEN=15; //maximum length of string buffer for RobotEQ
commands.
char X1_cmd[STR_LEN];
char X2_cmd[STR_LEN];
char Z1_cmd[STR_LEN];
char Z2_cmd[STR_LEN];
char MC_data[30];
char junk[30];
char attitude [length +20];
char temperature[8];
char pressure[8];
char humidity[8];

int pitch, roll, yaw;   // creates pitch, roll and yaw values
int temp1, temp2;
int camera, lights;


void setup()
{
   Wire.begin();    // Conects I2C

//  pinMode(20,  INPUT_PULLUP);
//  pinMode(21,  INPUT_PULLUP);
```

```
  Serial.begin(38400);
  Serial2.begin(115200);
  Serial3.begin(115200);

  RS485Serial.begin(38400);

  //Initialize relay pins as outputs and set to low
  pinMode(camera_relay, OUTPUT);
  pinMode(lights_relay, OUTPUT);
  digitalWrite(camera_relay, LOW);
  digitalWrite(lights_relay, LOW);
}



void loop()
{
 delay(20);


 //cehck if there are bytes to read, if so start read loop.

 if (rslHw.available()>0 )
   {

 //get command from simulink
 //rslHw.getMessage reads data on a hardware serial port and saves it
 //to an object called message
  rslHw.getMessage();

//  Read untill we are out of bytes or get a proper message.
  while((rslHw.ReadFail!=0) && (rslHw.available()) )   {rslHw.getMessage();  }

  //ONLY send data if a proper message was recieved
  if(rslHw.ReadFail==0)
    {

    //get string and length of string for MotorControler data
    int dataLen=0;
    GetMicrocontrollerStatus(MC_data);
//    sprintf(MC_data," hello");              //debug line
    while(MC_data[dataLen]!=0x00){dataLen++;}
//    Serial.println(MC_data);               //debug line

    //get compass data
    CMPS10();
```

```
        int attLen=0;
//   sprintf(attitude,"180,90,45");                  //debug line
        while(attitude[attLen]!=0x00){attLen++;}
//   Serial.println(attitude);                       //debug line

        //get temperature data
        WaterTemp();
        int tempLen=0;
        while(temperature[tempLen]!=0x00){tempLen++;}
//   Serial.print(temperature);
//   Serial.print("  ");

        GetPressure();
        int presLen=0;
        while(pressure[presLen]!=0x00){presLen++;}
//   Serial.print(pressure);
//   Serial.print("  ");

        GetRH();
        int humLen=0;
        while(humidity[humLen]!=0x00){humLen++;}
//   Serial.print(humidity);
//   Serial.println("  ");

        //combine the 2 strings and add a comma
        sprintf(attitude,"%s,%s,%s,%s,%s,", attitude, MC_data,
temperature,pressure,humidity);
//   Serial.println(attitude);              //debug line
//   Serial.println(attLen+dataLen +1);        //debug line

        //send message to computer adress "z". the +5 is for the added length of the 5
commas

rslHw.sendMessage(0x7A,attitude,attLen+dataLen+tempLen+presLen+humLen +5);



        //create the formated motor commands
        // The rslHw.message is a CHAR array, this causes havok with math, so it is
converted to bytes first.
        // there is a gain applied to scale the incomming values (0-255) to the max for
the motor controller (1000)
        // the command [ff 0] is full forward; [0 ff] is full reverse.
        temp1=  GAIN * byte(rslHw.message[0]);  temp2= GAIN *
byte(rslHw.message[1]);
        R_EQ_FORMAT(X1_cmd, temp1, temp2, 1 );
```

```
        temp1=  GAIN * byte(rslHw.message[2]);  temp2= GAIN *
byte(rslHw.message[3]);
        R_EQ_FORMAT(X2_cmd, temp1, temp2, 2);

        temp1=  GAIN * byte(rslHw.message[4]);  temp2= GAIN *
byte(rslHw.message[5]);
        R_EQ_FORMAT(Z1_cmd, temp1, temp2, 1);

        temp1=  GAIN * byte(rslHw.message[6]);  temp2= GAIN *
byte(rslHw.message[7]);
        R_EQ_FORMAT(Z2_cmd, temp1, temp2, 2);

       // write the motor commands to the correct motors
       Serial2.println(X1_cmd);
       Serial3.println(Z1_cmd);

       Serial2.println(X2_cmd);
       Serial3.println(Z2_cmd);

         //turn camera on/off
       camera = byte(rslHw.message[8]);
       if(camera == 1){
         digitalWrite(camera_relay, HIGH);
         Serial.println("on");
       }
       else{
         digitalWrite(camera_relay, LOW);
         Serial.println("off");
       }

       //turn lights on/off
       lights = byte(rslHw.message[9]);
       if(lights == 1){
         digitalWrite(lights_relay, HIGH);
     //     Serial.println("on");
       }
       else{
         digitalWrite(lights_relay, LOW);
     //     Serial.println("off");
       }
        }//if(rslHw.ReadFail==0)
       } // if (rslHw.available()>0 )

     } //void loop
```

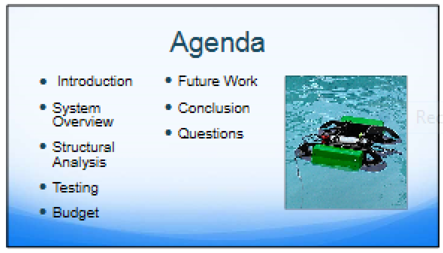
```
char* R_EQ_FORMAT(char c_array[], int b1, int b2, int motor_num){
  // creats a formated command in c_array. The basic format is "!g motor# val."
  // Given a packet of [0, num] for each motor, the order determins if  the
  // commanded value is positive or negative.
   if (b1>b2){
     sprintf(c_array, "!g %d %d", motor_num, -b1);
     }
  else {
     sprintf(c_array, "!g %d %d", motor_num, b2);
  }
  return c_array;
}

void CMPS10(){
  // This is the function that asks the compass for data over the I2C bus.
//  Serial.print("1");

  // clear out old data.
  while(Wire.available()>0){Wire.read();}
//  Serial.print("2");
   // highByte and lowByte store high and low bytes of the bearing
   byte highByte, lowByte;
   //starts communication with CMPS10
   Wire.beginTransmission(ADDRESS);
   // ask for register #2 (2-3=yaw, 4=roll,5=pitch).
//  Serial.print("3");
   Wire.write(2);
   //required end transmission. For some reason part of cmps10 protocol.
//   Serial.print("4");
   Wire.endTransmission();
   // Request 4 bytes from CMPS10
//   Serial.print("5");
   Wire.requestFrom(ADDRESS, 4);
//  Serial.print("6");
   /* Wait for bytes to become available. Waiting for Wire.available() bytes does
not work,
   as the compass ocassionally does not respond. This would leave us stuck in a
blocking
   read with no way out. There is no I2C read timeout in Arduino, so that is not an
option.
   A manual timeout could be created, but the simple delay below is 2 times more
that it should take
   for data to be recieved, and works with very little issue.*/
//   delay(18);
```

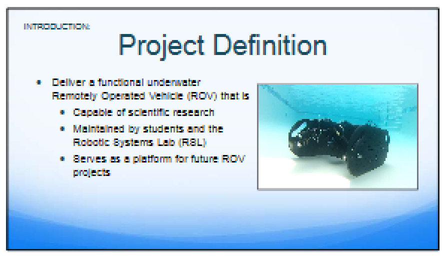
```
   delay(20);

   // we should get 4 bytes back.
   if(Wire.available() >= 4){
//       Serial.print("7");
      highByte = Wire.read();
//      Serial.print("8");
      lowByte = Wire.read();
//Serial.print("9");
      pitch = Wire.read();
//Serial.print("10");
      roll = Wire.read();
//Serial.print("11");
      // Calculate full yaw
      yaw = ((highByte<<8)+lowByte)/10;
//Serial.print("12");
      //store yaw, pitch, roll as attitude
      sprintf(attitude,"%d,%d,%d",yaw,pitch,roll);
//       Serial.print("13");
   }
   // close out connection to the compass.
   Wire.endTransmission();
//   Serial.println("14");

}



void GetMicrocontrollerStatus(char mc_buffer[]){
// This asked the RobotEQ bord for the current battery voltage and the temp of
each channel

//clear out old bytes in the serial buffer
 while(Serial2.available()>0){Serial2.read();}
// while(Serial3.available()>0){Serial3.read();}

Serial2.println("?v 2");
 Serial2.println("?t");
// Serial3.println("?t");

 int n=0;
 delay(5); //delay for serial turnaround

 /*the basic message is: ?v 2V=110
 ?tT=27:27
 yes there should be a carriage return (0x0d) in there*/
```

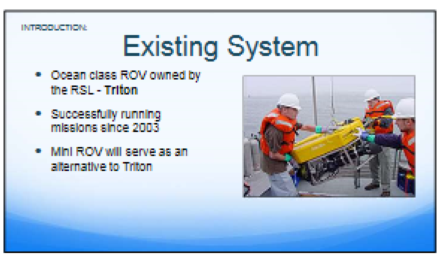
```
 //read untill we get a "=" to start the first return
 readUntill(Serial2, '=', junk, 10);
 // read untill the end of this message, which is a 0x0d (a carriage return)
 readUntill(Serial2, 0x0d, junk, 10);

//use string writes to get the data into a output string
 sprintf(mc_buffer,"%s",junk);

 //read untill we get a "=" to start the second return
 readUntill(Serial2, '=',junk,10);
 //read untill : to mark end of fist variable
 readUntill(Serial2, ':',junk,10);
//use string writes to get the data into a output string
 sprintf(mc_buffer,"%s,%s",mc_buffer,junk);
//read second half of variable, should be no more than 2 bytes
readUntill(Serial2, 0x0d,junk,2);
 sprintf(mc_buffer,"%s,%s",mc_buffer,junk);


// readUntill(Serial3, '=',junk,10);
// readUntill(Serial3, ':',junk,10);
// sprintf(mc_buffer,"%s,%s",mc_buffer,junk);
// readUntill(Serial3, ':',junk,2);
// sprintf(mc_buffer,"%s,%s",mc_buffer,junk);
// Serial.println(mc_buffer);

}



void readUntill(Stream &ser, char stop_char, char array[], int max_bytes){
 /* This is a function designed to read the ASCII output of
 a robotEQ and help with processing it. after the stopcharacter is
 found, the string will be null terminated. Becasue this returns a null terminated
 string, it will not work with binary data. You have been warned.
 */

int k=0;
char c;

//c=stop_char+1;
c=ser.read();
while ( (k<max_bytes) && (c!=stop_char)  && (ser.available()>0) )
    {
     array[k++]=c;
     c=ser.read();
```

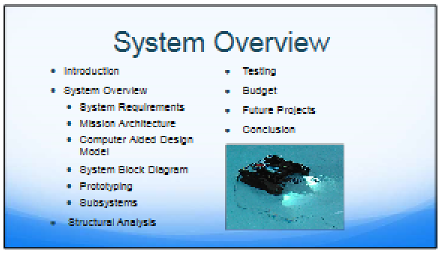
```
    }

    array[k]=0x00;
  }

  void WaterTemp(){  //Read sensor voltage coming in and convert to degrees celcius, then print into "temperature" string
   float value = analogRead(temp_sensor);
   float temp_voltage = value/1024*5000;
   int temp = (temp_voltage - 750)/10 + 25;
   sprintf(temperature,"%d",temp);
  }

  void GetPressure(){  //Read sensor voltage coming in and convert to feet, then print into "pressure" string
   float presval = analogRead(pres_sensor);
   float pres_voltage = presval/1024*5000;
   float pres = (pres_voltage - 500)*62.5/1000;
   int depth = pres/14.7*33;
   sprintf(pressure,"%d",depth);
  }

  void GetRH(){  //Read sensor voltage coming in and convert to % relative humidity, then print into "humidity" string
   float humidityval = analogRead(RH_sensor);
   float humidity_volt = humidityval/1024*5000;
   int RH = (humidity_volt - 802)/30.1;
   sprintf(humidity,"%d",RH);
  }
```

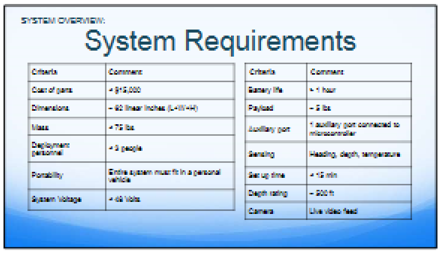
Appendix J - Senior Design Slides

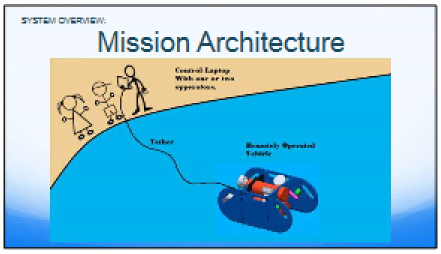


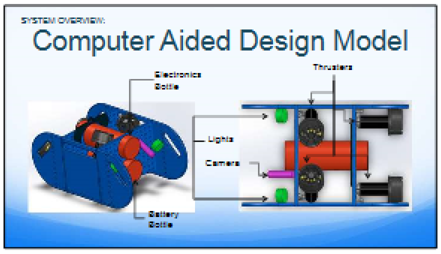


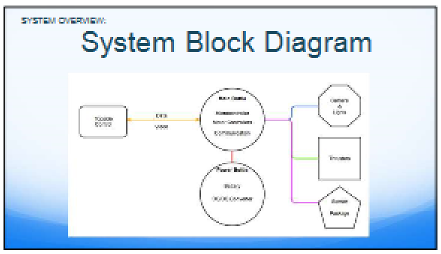



System Overview
• Introduction
• System Overview
• System Requirements
• Mission Architecture
• Computer Aided Design Model
• System Block Diagram
• Prototyping
• Subsystems
• Structural Analysis
• Testing
• Budget
• Future Projects
• Conclusion


SYSTEM OVERVIEW:
System Requirements
Criteria | Comment
Cost of parts | < $15,000
Dimensions | ~ 60 linear inches (L+W+H)
Mass | < 75 lbs
Deployment personnel | < 3 people
Portability | Entire system must fit in a personal vehicle
System Voltage | < 48 Volts
Criteria | Comment
Battery life | ~ 1 hour
Payload | ~ 5 lbs
Auxiliary port | 1 auxiliary port connected to microcontroller
Sensing | Heading, depth, temperature
Set up time | < 15 min
Depth rating | ~ 300 ft
Camera | Live video feed


SYSTEM OVERVIEW:
Mission Architecture
Control Laptop With one or two operators.
Tether
Remotely Operated Vehicle

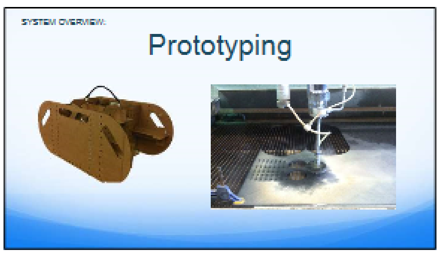

SYSTEM OVERVIEW:
Computer Aided Design Model
Electronics Bottle
Thrusters
Lights
Camera
Battery Bottle

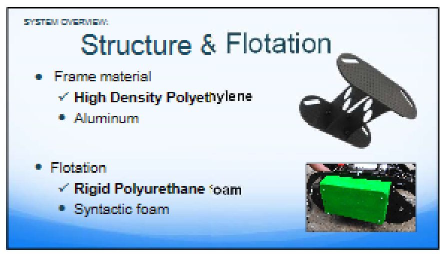

SYSTEM OVERVIEW:
System Block Diagram


SYSTEM OVERVIEW:
Topside Control
• 3 Modes of Operation
✓ Analog Control Box
• Laptop
• Tablet interface (COEN team)

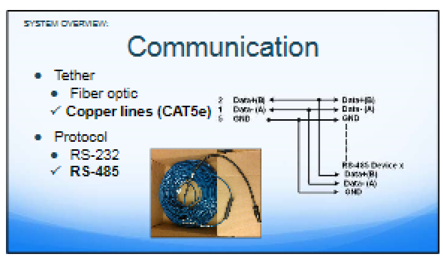

SYSTEM OVERVIEW:
Topside Control
3 Modes of Operation
Analog Control Box
✓ Laptop
Tablet interface (COEN team)


SYSTEM OVERVIEW:
Topside Control
3 Modes of Operation
Analog Control Box
Laptop
✓ Tablet interface (COEN team)


SYSTEM OVERVIEW:
Prototyping

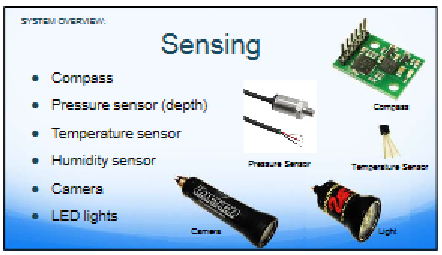

SYSTEM OVERVIEW:
Structure & Flotation
Frame material
✓ High Density Polyethylene
Aluminum
Flotation
✓ Rigid Polyurethane 'oam
Syntactic foam

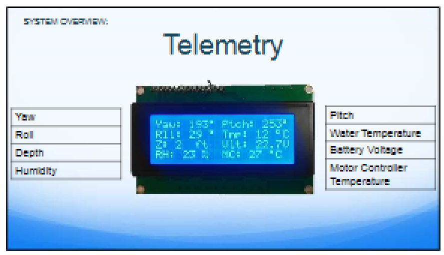

SYSTEM OVERVIEW:
Electronics and Housing
Arduino Mega Microcontroller
Standard in the Robotics Systems Lab
Aluminum Bottles with End caps
✓ Air filled
Oil filled


SYSTEM OVERVIEW:
Communication
Tether
Fiber optic
✓ Copper lines (CAT5e)
Protocol
RS-232
✓ RS-485

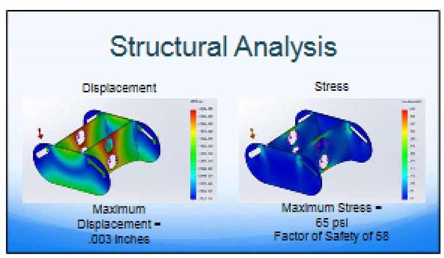

SYSTEM OVERVIEW:
Power
Lithium Polymer Battery
6 cell (22.2 V) – 10 Ah
Runtime ~ 1 hr
DC/DC converter (12 V) to aid noise immunity
Control electronics and sensors

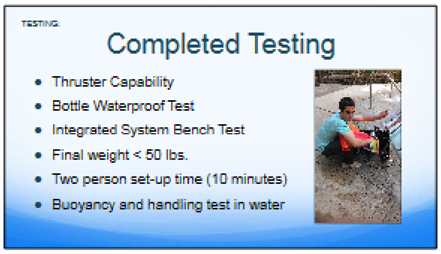

SYSTEM OVERVIEW:
Propulsion
Seabotix ROV thrusters
Brushed DC
Rated to 500 ft
RobotEQ SDC2130 Motor Driver
2 channel
Voltage and temperature monitoring

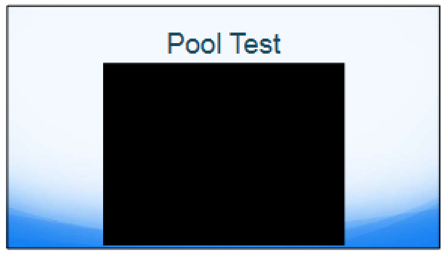

SYSTEM OVERVIEW:
Sensing
Compass
Pressure sensor (depth)
Temperature sensor
Humidity sensor
Camera
LED lights
Compass
Pressure Sensor
Temperature Sensor
Camera
Light

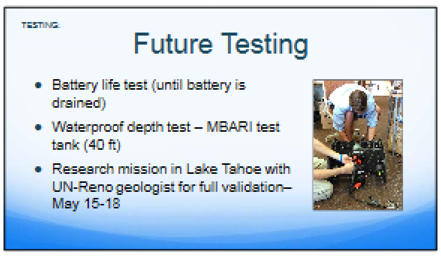

SYSTEM OVERVIEW:
Telemetry
Yaw
Roll
Depth
Humidity
Pitch
Water Temperature
Battery Voltage
Motor Controller Temperature

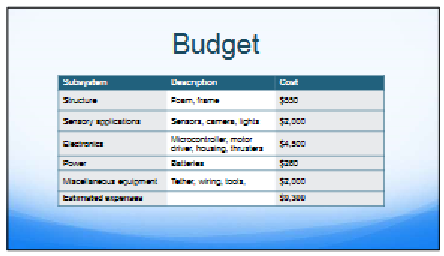

SYSTEM OVERVIEW:
Demonstration


Structural Analysis
Displacement
Stress
Maximum Displacement = .003 inches
Maximum Stress = 65 psi
Factor of Safety of 58

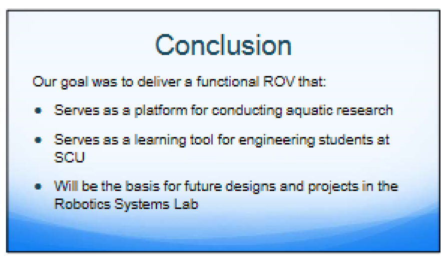

TESTING
Completed Testing
• Thruster Capability
• Bottle Waterproof Test
• Integrated System Bench Test
• Final weight < 50 lbs.
• Two person set-up time (10 minutes)
• Buoyancy and handling test in water


Pool Test


TESTING
Future Testing
• Battery life test (until battery is drained)
• Waterproof depth test – MBARI test tank (40 ft)
• Research mission in Lake Tahoe with UN-Reno geologist for full validation– May 15-18


Budget
Subsystem | Description | Cost
Structure | Foam, frame | $550
Sensory applications | Sensors, camera, lights | $2,000
Electronics | Microcontroller, motor driver, housing, thrusters | $4,500
Power | Batteries | $250
Miscellaneous equipment | Tether, wiring, tools, | $2,000
Estimated expenses | | $9,300

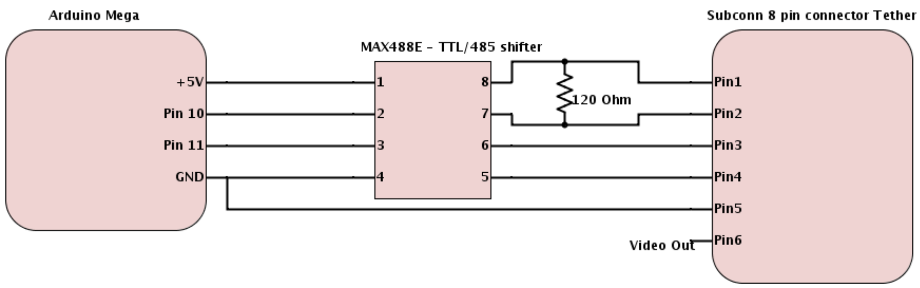

Future Projects
• Characterization of system dynamics
• Auxiliary functionality
• Manipulator
• Sensor packages


Conclusion
Our goal was to deliver a functional ROV that:
• Serves as a platform for conducting aquatic research
• Serves as a learning tool for engineering students at SCU
• Will be the basis for future designs and projects in the Robotics Systems Lab

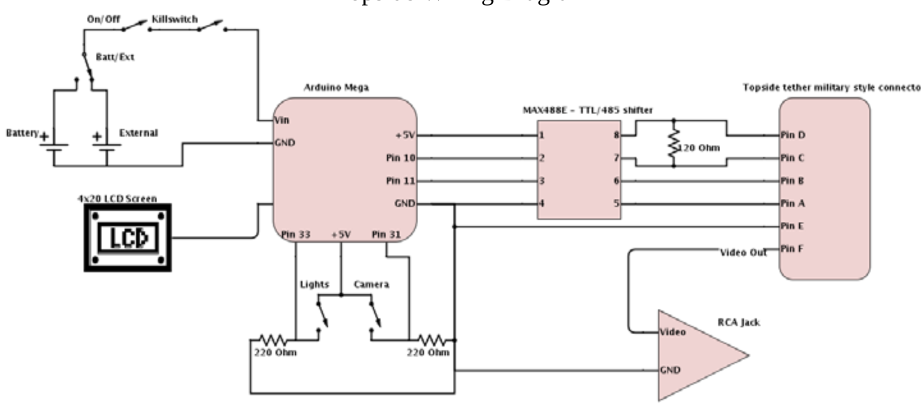

Questions?

Appendix K - Full Circuit Diagram

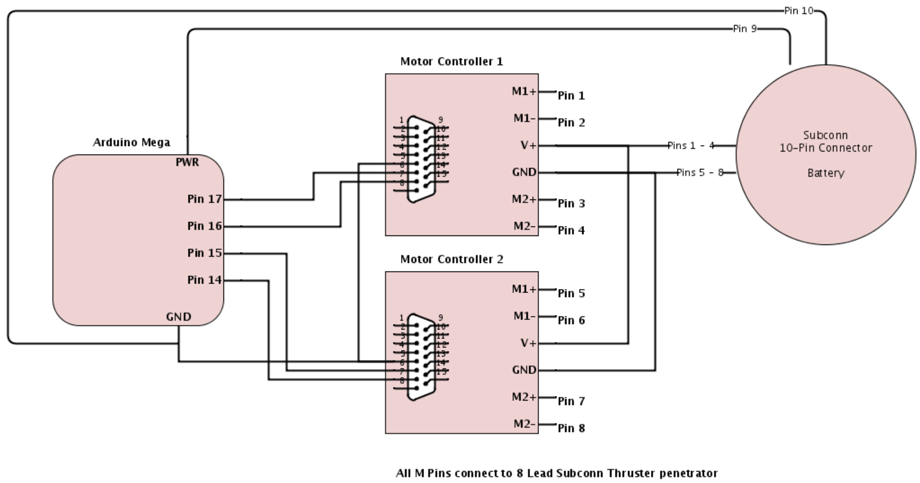
Communication Wiring Diagram



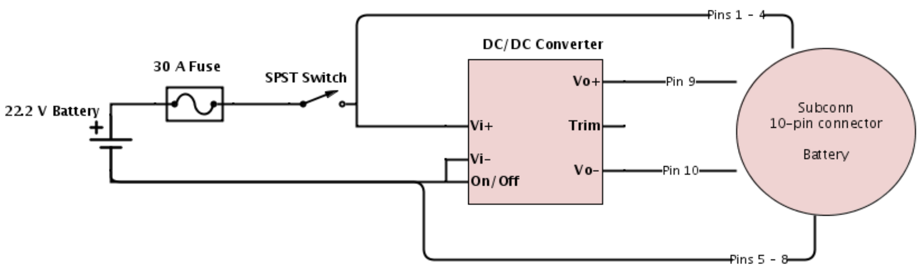
Topside Wiring Diagram



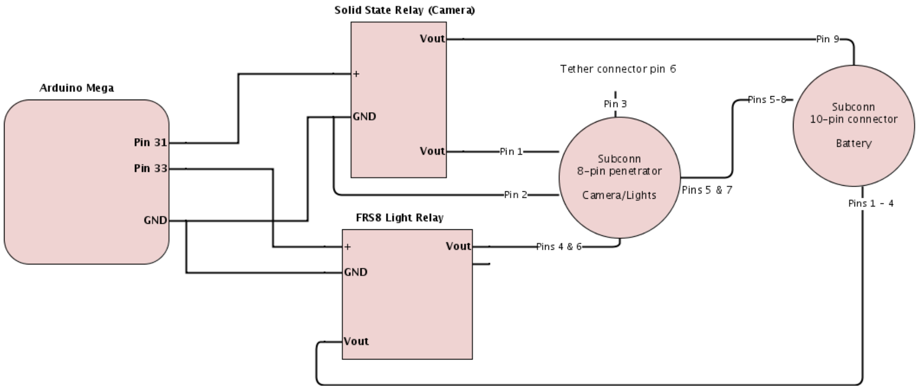
Motor Controller Wiring Diagram



Battery Bottle Wiring Diagram



Camera and Lights Wiring Diagram



Tether Pin Connectors

Subconn 8 pin	**CAT5e**	**Topside connector**
1	Orange	A
2	Orange/White	B
3	Green	C
4	Green/White	D
5	Blue	E
6	Blue/White	F
7	Brown	G
8	Brown/White	H

Appendix L - Manufacturing and Drawings

L.1 - Discussion

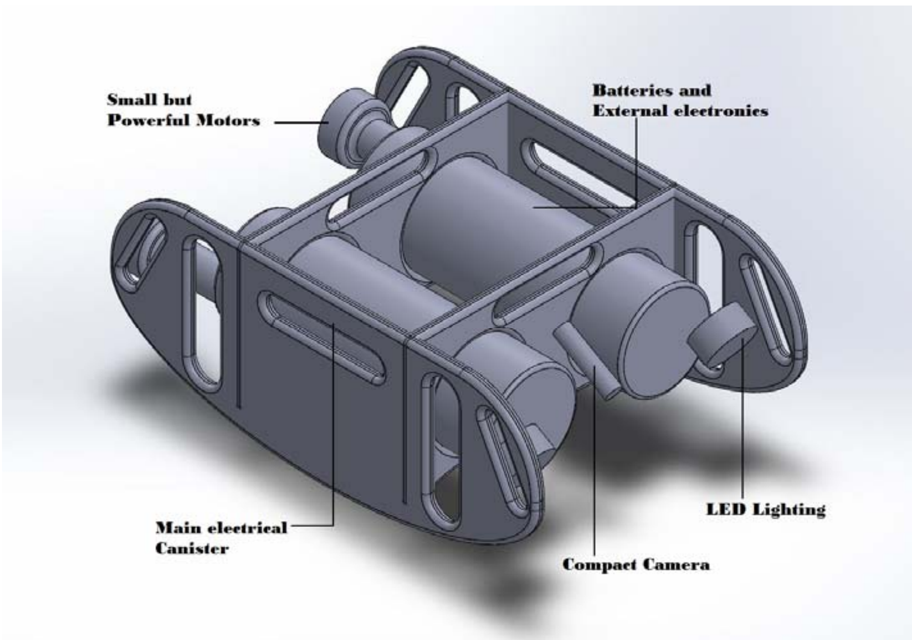
When building a robot you must design for the completion of the project. And from the beginning that is what we did. We knew that coming up with a design would be simple, but we focused instead on making something that would be easier for production. Everything from the frame to the electronics was talked about and reviewed for ease of use. Our project is not designing an object and focusing on that one thing but instead the integration of complex electronics and assembly of different parts from a variety of venders.

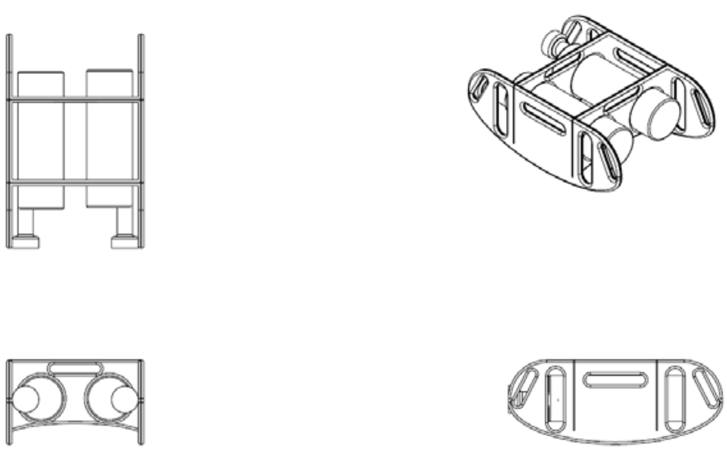
To start off we looked at what we needed: frame, microcontroller, motor drivers, lights, camera, thrusters, water proof plugs, electronics housing, foam and a frame to hold it all. We then started to break it down and choose components that we were familiar with and that we knew would be able to work. We planned ahead for the inevitable problems that would occur when gathering parts and trying to get them to work together. We took our time and designed everything so that we would have as few problems as possible when it came to actually producing the product. The best example of this is the frame. Most other robots that we have in the Robotics Systems Lab at Santa Clara University are aluminum which is strong but painstaking to manufacture. We looked at this and decided that this procedure was far too complex and spent time searching for a better alternative. We came up with simply water jetting the frame. This allows precision cuts and guarantees that it will be right the first time with minimal problems.

Our manufacturing process is different than many other groups seeing as we took components, modified them and enabled them to work together to finish the robot. Starting with the big picture of finishing the robot, everything was looked at and verified that it would work together.

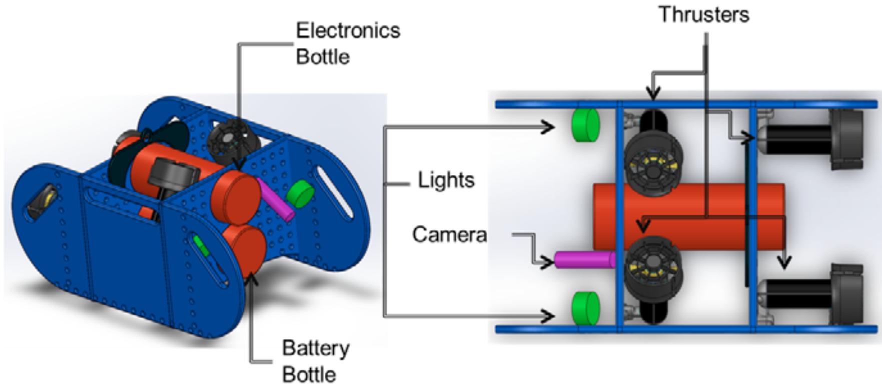
L.2 - Preliminary Drawings

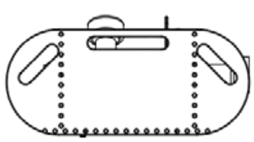
L.2.1 - Version 1

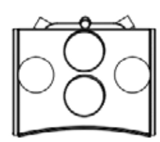


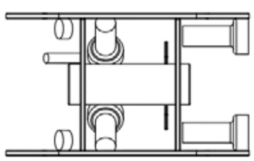


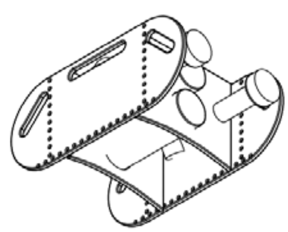
L.2.2 - Version 2



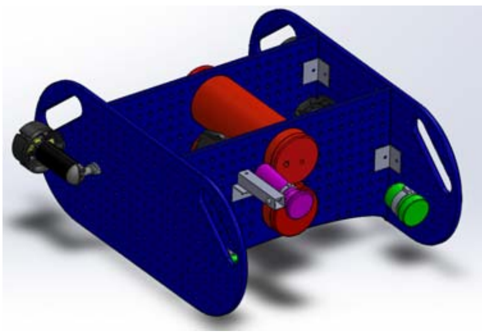


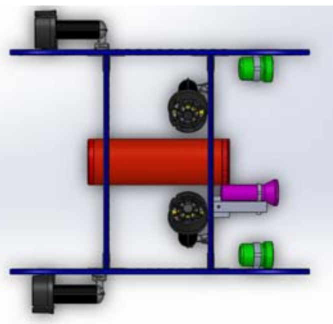


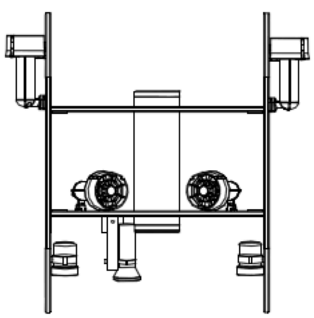


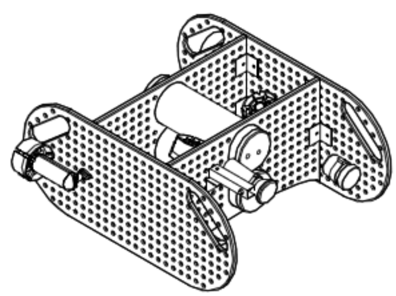


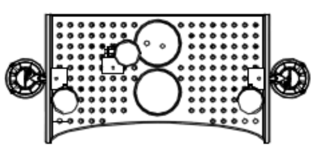
L.2.3 - Final Version

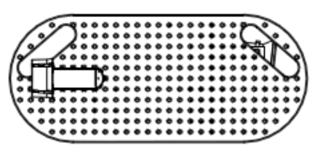












L.3 - Final Drawings

Appendix M - Instruction Manual

M.1 – Pre-Deployment Checklist

- Connect tether to main electronics bottle and attach the handle carabineer to strain relief sleeve on tether
- Ensure all plugs are connected and locking sleeves are tight
- Ensure all components are secured to ROV
- Connect other end of tether to the control box and connect video monitor
- Turn on ROV battery switch in the bottom bottle
- Turn on control box, either battery powered or through power jack, test functionality
    - Spin all thrusters in both directions
    - Turn on camera, get live feed
    - Turn lights on/off
    - Receive sensor data
- Ensure ROV battery is charged (full charge > 22.2 V)
- Grease end cap O-rings and cap the electronics and battery bottle
- ***Plug the purge/vent holes*** on each bottle by using the plugs that are attached to the frame
- Test functionality once more
- ***Check purge/vent holes***

M.2 - Operating Procedure

- Lift ROV using the handles, one person on each side
- Make sure there is enough loose tether so that the ROV can safely be put in the water without the tether pulling
- Place ROV in water
- While one operator is driving the ROV, another will be on tether duty
    - Thether duty consists of holding on to the tether and giving is slack or pulling it in so that the ROV never has too much tether underwater. ***Do not pull.*** If tether is reaching the end of the line, let driver know so they can stop and the tether can be reeled in
- Let those deploying know when you are diving or surfacing the ROV
- When driving the ROV, if driving seems off, let go of joysticks for 2 seconds then resume. Most commonly occurs when going from forward/reverse to rotation
- Monitor ROV battery voltage is **> 19.5 V**. If under, surface ROV and swap battery
- ROV can surface using thrust or by pulling the tether. When pulling the tether, do so in soft, slow and consistent motions. No sudden jerking of the line.

- When reeling the ROV and tether in, make sure the tether is being laid down in loops so that it will not tangle
- One or two people can pull the ROV out of the water using the handles, always look out for the thrusters hitting anything