Bubbles Under The Sea

Jacob LaVoy, Madison Melton, Avery Mills, Joshua Nichols

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

Abstract — As a number of activities revolve around water, it's no surprise that humans live near bodies of water such as rivers and oceans. Among the activities that can be done on or in the water, jet skiing, sailing, wakeboarding, fishing, and scuba diving are among the most prominent. This means that for most water-based activities, a lot of time is spent seeing what is above or on the water's surface. Our motivation in creating a remotely operated submarine is to observe what's under the surface, and without the expensive gear typically required. For research or for fun, our remotely operated submarine allows users to explore the underwater world with the vision capabilities of being there themselves. This paper aims to summarize the functions of our remotely operated submarine, focusing on its construction and its features.

Index Terms — Aquatic robots, Machine-to-Machine Communication, Controller PCB, Drone PCB, PoE, Motorized Craft, Battery operated

I. INTRODUCTION

Our senior design project is named Bubbles Under The Sea. The purpose for our project is to make an underwater exploration device that can be easily used and does not require a lot of expensive gear or knowledge on underwater exploration. This way, we can further expand the ability of the general public to explore and see environments under water.

Our project has several features to make this underwater exploration easier including the following: a panning and tilting camera for vision, headlights for darker environments, motors for movement along the entire horizontal axis, a bladder system for movement along the vertical axis, a sonar system for rear object warning, and a depth sensor. All of these features can be controlled via a hand held controller and display so that the user can easily control the movements of the motors, bladder, and camera while simultaneously being able to see a live feed of the camera on the monitor.

II. SYSTEM OVERVIEW

A. Hardware

The Bubbles Under the Sea project is focused on expanding upon the capabilities offered by existing sub-$1000 submarines. The submarine's duties are to facilitate the movement of the submarine itself and provide more control options, especially for the camera. A physical realization of the submarine was created not only to carry out these duties, but also to bear the physical appearance of traditional human-sized submarines as seen in Figure 1. The remainder of this section will describe the hardware for the submarine, with a block diagram of our system shown in Figure 2.

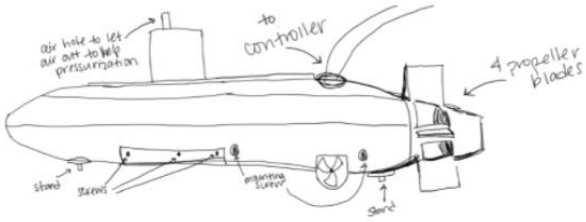

Figure 1: Initial design sketch

As is common with other similar products, we have separate land and sea components. On the land is a 60V battery (hereafter known as the "land battery"), which powers the entire system. The power is fed to all land-side items first including the controller, which is the user's primary means of interacting with the submarine. The controller is attached to a Raspberry Pi 4 which handles the transmission of commands from the controller, and the image packets that come in from the submarine are displayed on an liquid crystal display which is attached to the input and output Pins of the Pi. On land is also a printed circuit board responsible for powering the controller from the energy supplied by the land battery, and takes this power along with data from the controller to be forwarded to the submarine using a Power over Ethernet (PoE) source circuit.

The power and data are transmitted over an Ethernet cable that enters the submarine through the conning tower, and connects to a receiving Power over Ethernet powered device which then breaks the cable output back into Ethernet and power. The power is then used to power the Raspberry Pi, while the Ethernet data is provided to the Pi's networking systems. In addition to the power provided by the PoE from the umbilical cable is also the power provided by the three 11.1V batteries. These specially power the motors and pumps independently, and serve as a way of extending the lifespan of critical systems. All small devices such as the camera, pan/tilt, depth sensor, and sonar however ultimately get their power through the Pi which is distributed from the on-board PCB.

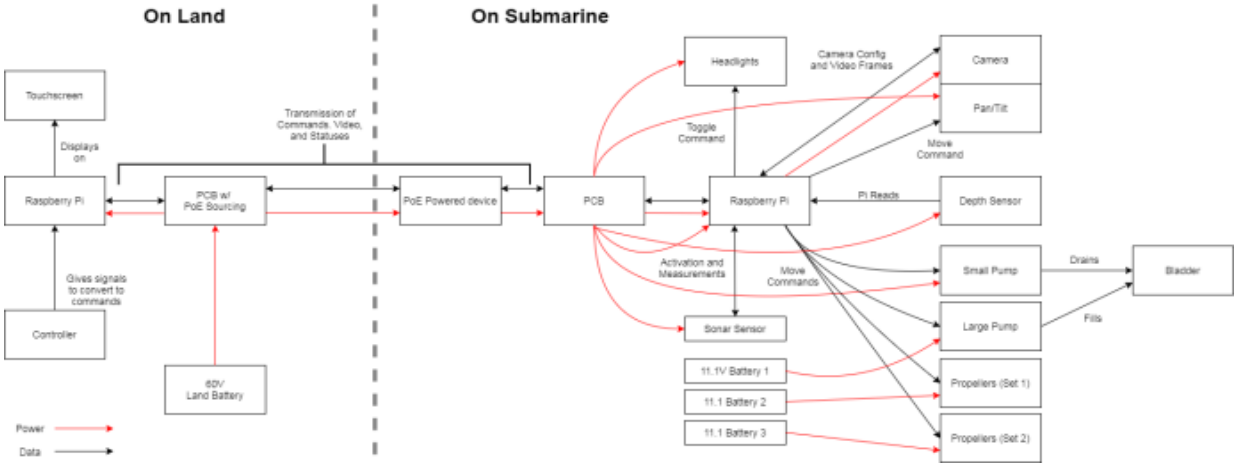

Figure 2: Overall Block Diagram

B. Data Flow

The software used to control the submarine was developed using C++ on the Raspbian operating system. Throughout development of the software, Linux Mint 19.3 and Windows via PUTTY were also used to provide an overall smoother workflow between more powerful desktop computers and the submarine Raspberry Pi. Immediately on system start, both Raspberry Pis will begin to send data to each other using the User Datagram Protocol (UDP). Each Pi will initialize two threads, one dedicated to video and another dedicated to the transmission of commands and sensor data. The on-board Pi then initializes the camera and sensors within their respective threads. The submarine takes controller inputs and determines what motors need to be activated along with their appropriate speed and direction.

III. INPUT

A. Camera

The camera we chose was the Raspberry Pi Camera Module V2 as seen below in Figure 3.

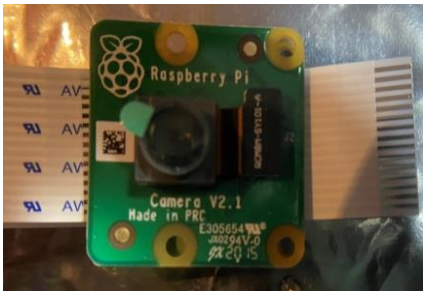

Figure 3: Raspberry Pi Camera Module

It was chosen for a number of reasons including its low price point, 1080p resolution, small profile, community libraries, and the use of a non-GPIO interface. It uses the IMX219 sensor, and as such has a full resolution of 3280 by 2464 Pixels. DesPite having a smaller static viewing angle of 48.8 degrees vertically and 62.2 degrees horizontally [1], this is not an issue when mounted to the pan/tilt as the user can use that to look around. The pan/tilt range is shown in Figure 4.

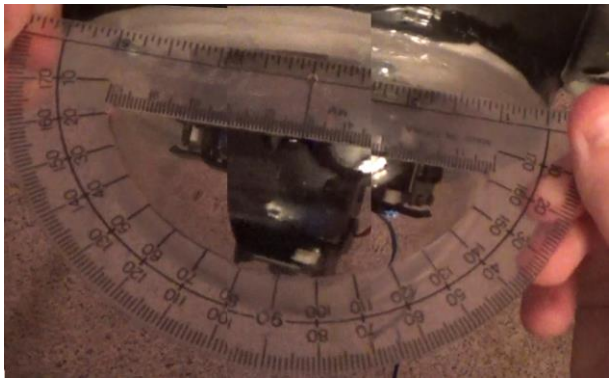

Figure 4: Horizontal Motion Image Collage

Processing of the camera is done through use of the OpenCV and RasPicam libraries. They are used to break down the captured images into packets which are then sent out using standard C++ sockets. At the target resolution of 480 * 800, the controller received images at a steady rate of 100 frames per second. The average measured latency from the time the last packet was sent to the time the last packet was received was 7.156 ms. These results show that the user will experience livestream viewing. A similar test was done to measure the delay of a user action such as between the left stick and the motors running. The average time between the stick moving and the motors running was 17.633 ms.

B. *Sensors*

The user will be given information from several sensors to let the user know the status of the submarine. The most important sensors are the sonar, and the depth sensor. Like a car sensor, the sonar will be used for rear object detection, to detect anything in the way of the submarine when reversing. The depth sensor is more special, in that it is actually a pressure sensor with temperature sensing abilities. This means that the temperature of the surrounding water can be measured as well as the pressure on that area of the submarine. The relation between depth and pressure is used to derive the current depth value.

The sonar chosen was the DFRobot SEN0208, as seen below in Figure 5, chosen for its operating range, detection cone, and weight.

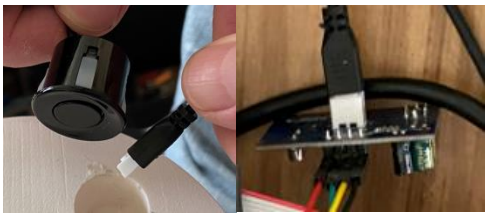

Figure 5: DFRobot SEN0208

The depth sensor chosen was the MS5837-30BA as seen below in Figure 6, and packaged by BlueRobotics. It was Picked for its usage of I2C (which won't block other GPIO devices), high resolution, and being high-pressure resistant (30 bar).

The Raspberry Pi uses I2C to read the data captured by the sensor. This process involves sending a command to the appropriate register, waiting out the repeated starts and other I2C signals, then reading from the result register. To ease development, libraries from the Pi's chip manufacturer along with a Github repository were used as opposed to raw I2C.

Figure 6: MS5837-30BA Pressure Sensor

IV. BALLAST

In order for the user to control the movement on the vertical axis we needed to have a ballast system to either increase or decrease the weight of the submarine, allowing it to either sink or float respectively.

The ballast was determined based upon the weights calculated for the solid hull parts and their respective densities. The weights of all the internal components like batteries and Raspberry Pi were added to this first sum. This was then compared against the weight of the water displaced assuming the submarine was one homogenous solid. This got us a weight and subsequently the volume that the bladder needed to hold to work.

Our ballast design was to have a large bladder that could be filled with water or drained of water to increase the weight in the submarine. In order to do this we have two water pumps as seen below in Figure 7 that are powered separately. On the intake pump we also have a solenoid valve that will allow us to block incoming water, and on the exfil value we have a back flow preventer to stop water from getting in through the exit.

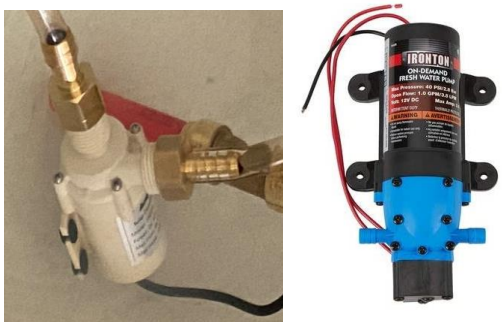

Figure 7: Infil and Exfil pumps

V. PCBS AND POWER

Although our design has a simple concept, many signals had to be traced and power needed to be traced. For our design to work, we needed to have a power source that was not very heavy inside the sub, so we decided to use Power over Ethernet (PoE) rather than weigh down the sub with extra batteries, as that would make it harder for the motors to propel. Power over Ethernet is a great concept where the extra two pairs of cabling inside an Ethernet cable are to be used rather than sit idle. These other two pairs are usually used when there is a high volume of data being transmitted, but for our purposes, they would not be used. Instead, we are now using the original Tx/Rx wires as positive and negative voltage passing lines. The boards designed

for power distribution can be seen in Figures 8,9, and 10.

There are four different ratings for PoE, consisting of 802.3af, which can supply 12.95 W max; 802.3at, which can supply 25.5 W max; 802.3bt, which can supply 51 W, and another version of 802.3bt, which cam supply 71 W. Type 1 is called “PoE”, Type 2 is called “PoE+”, and Types 3 and 4 are called “4PPoE” or “PoE++.” Through all four ratings, all of them need to have a higher tolerance to account for the voltage/current loss in the length of the Ethernet cable connecting. Category 5 and Category 6 Ethernet cables are capable of supplying the Type 4, which is what we used.

There are two separate PCBs in our design. One PCB is on land and is called the “Controller PCB”, whereas the other one is inside the drone and is called the “Drone PCB”. The 60V Greenworks battery is attached to the Controller PCB to power most things. Power will be transmitted to the Drone PCB via an Ethernet cable using the concept known as “Power over Ethernet”. On land, the power will be transmitted from the Controller PCB to the Raspberry Pi via a USB-C cable, and to the Drone board via the long Ethernet cable. Data will be transmitted over the long Ethernet cable as well and through a smaller Ethernet cable to the land Raspberry Pi. This is how the signals will be transmitted for guidance and navigation as well as sensing information.

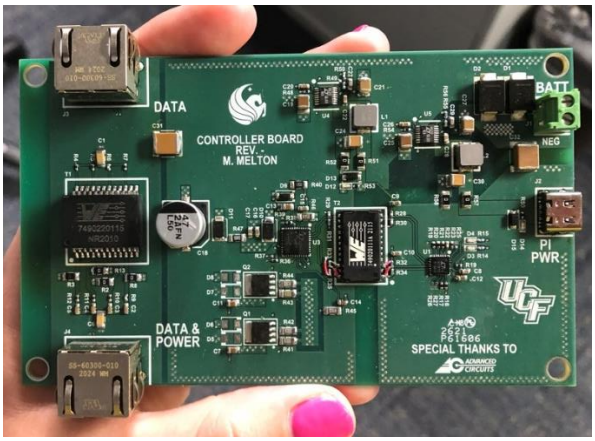

Figure 8: PCB connecting controller to sub

Once the data signals and power have been transmitted through the long Ethernet 802.3bt standard Ethernet cable, the power will flow to a Power over Ethernet Powered Device board that receives the 58 volts coming in through the Ethernet cable and reduces it down to 5 volts. We chose the powered device board that regulated down to 5 volts, because all of the devices that are being powered from this PCB require 5 volts.

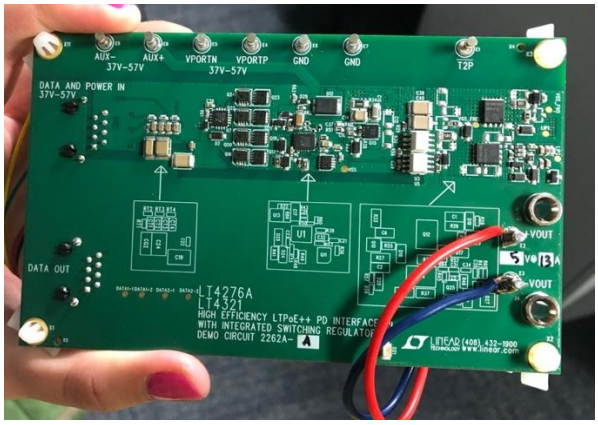

Figure 9: Back of drone PCB

The Ethernet cable connects to this powered device board to supply the power to the board, and then there is a jumper Ethernet cable from the powered device board to the drone board to supply the data to get to the drone board, which will be funneled to the Raspberry Pi via Ethernet as well.

The two wires Pictured above (red and black) supply the 5 volts to the board attached to this powered device entering in through a terminal block. Once the 5 volts gets to the Drone board, it begins to power up the add-ons, such as the sonar sensor, drone Raspberry Pi, and the little pump.

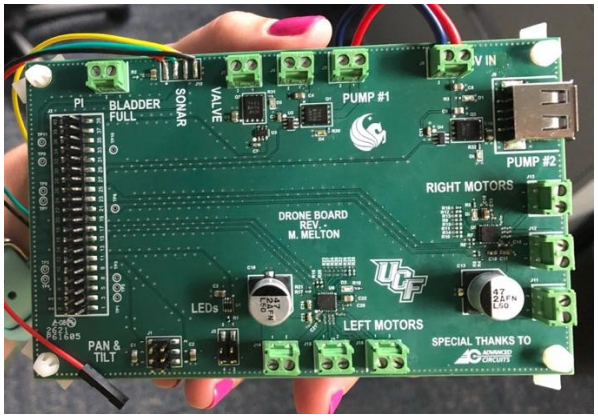

Figure 10: Front of drone PCB

On the drone board, there are several terminal blocks to allow for all of the connections to the individual Pieces. Most of the connections are just red and black power cables, so this type of connector made the most sense. On the right side is an Ethernet port where the data will be transmitted to the Powered Device board

directly below this board. There are several LEDs on this board that helped with testing the circuits to make sure that power was getting where it needed to be. Without an indicator light, it would have been much harder to diagnose during testing. Where the sonar sensor is in the top left corner of this board, we originally had a 4-Pin header there, but they were all sold out through every vendor, so we had to make a make-shift header with four connector wires soldered onto the board.

Also, three exterior batteries connect to this board that power the motors and the large pump respectively. These three batteries are 11.1 V 5Ah Youme batteries that are quite light. Each battery only weighs 1.5 pounds, which was important in our decision. We did not want to make the drone too heavy for the little four motors to propel. One battery powers the right set of two motors, another battery powers the left set of two motors, since each motor can only handle about 2.5 Amps. The last battery powers the large pump. All of these connections are joined on this board, so that they would all be in a central place and the controls from the Raspberry Pi would be able to work, as well as there is a Pulse Width Modulation circuit on this board to allow for the motors to work.

There is a 40-Pin header Pictured above that connects to the drone Raspberry Pi to help with making the connections and commands. This is how the signals can be controlled to each device to either turn on or turn off.

All parts for both PCBs were designed in Altium by Madison and bought and hand-soldered on by Madison as well.

VI. CONTROL

A. Raspberry Pis

The Raspberry Pi 4B+ was chosen for its communication and video display capabilities and it being the smallest physical profile of the competing boards. The Raspberry Pi offered a number of add-on modules and communication modes that enhanced its capabilities, such as sensors, displays, and camera modules (as mentioned earlier).

B. Game Controller

The controller selected was a USB direct connect Xbox One controller. The reason behind the selection was due to a comfortable layout of buttons and a software driver that was compatible with the Raspberry Pi software allowing it to have the inputs directly tracked but C/ C++ code. A custom designed controller was originally called for, but to accommodate the touch screen required the use of too many GPIO pins. To accommodate touch screen capability the Xbox controller was be attached to the Pi via USB.

C. Display

The Display selected is a Waveshare 4-inch LCD display with touch screen capability. This allows for a perfect fit with and Xbox phone mounting bracket that can hold the microcomputer in place. While giving the user a perfect view of the screen they will have direct view of the controls so that any unfamiliarity can be put at ease while using the submarine.

VII. OUTPUT

The signals from the controller can result in several notable outputs on the submarine, including the following:

A. Propulsion

Four LICHIFIT motors provide the lateral propulsion used to move the submarine. They were chosen due to requiring significantly less power and them coming at a lower price than their competitors. We examined a few pre-built speed controller modules for use but ultimately needed to incorporate h-bridge integrated circuits to drive their speed and direction. To control the speed, the Pi uses pulse width modulated signals from two digital control inputs**.** Inside the Drone PCB, there is a Pulse Width Modulation circuit designed to allow these signals to pass through and control the motors. Each pair of motors moves in a CW/CCW direction, so we arranged them in a way to allow for the most water to be pushed through when the motors are turned on.

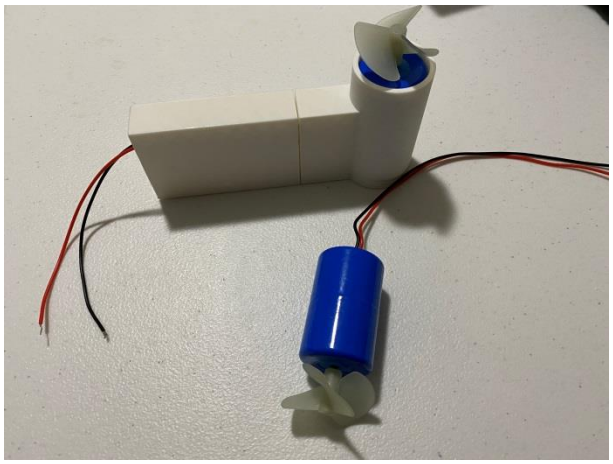

Figure 11: LICHIFIT Motors

B. Lights

While lumen is the oft quoted unit of measurement for light, it is a number measured in comparison to the uniform source of one candela. A different measurement, that of millicandela (mcd), is "a standard unit for measuring the intensity of a light source" [2], and made sense due to its direct influence on lumen. The tradeoff between viewing angle and millicandela rating guided the selection of the C513A-WSS LEDs. It is also believed that the small physical profile will help the LEDs to resist the pressures at the user's target depth. They have a 55 degree viewing angle, and when paired with an 8930 mcd rating, can provide light to a dark room easily. They are driven by a dedicated circuit embedded into the submarine PCB.

VIII. SOFTWARE

A. Submarine-Side

When the submarine is powered on, a short bash script is used to automatically launch any additional software along with the main program. This includes any daemons or services that the program may rely upon to work properly, such as the PigPio daemon which facilitates GPIO interactions.

As described earlier, the software of the submarine is based on the principle that running multiple threads allows speedup of the program through dedication of resources. Since the Raspberry Pi 4 is quad-core, each core can support one thread of execution, and the processing of video and the reception and execution of commands can be performed in parallel. A dedicated thread of execution for the video can mean that the user has a more up-to-date Picture of the environment that the submarine is in. Similarly, for commands this means that the submarine can more quickly react to the actions undertaken by the user. There is also an additional thread used for the sensors, which also allows the user the ability to react quicker to the local conditions due to the data being received on the controller faster.

When a command is received, the command thread logs it into a list which is then pulled from by a helper thread. The helper thread then interprets what the command does and executes the functionality inside its relevant section. This functionality can vary from controlling the pulse-width modulated signals used to control the pan/tilt to using I2C to read the depth that the submarine is at.

A notable output is of course the video that is scanned section by section into UDP packets. Upon initialization of the program, a calculation is made for the optimal number of packets with the largest size. We felt that sending packets with an increased size would result in the controller-Pi unpacking more information faster. Along with this image data is the return side of the command thread. The command thread has the second purpose of sending sensor data back to the controller, such as the depth, temperature, and bladder statuses.

On the C++ level, all packets are read and packed as strings, which simplified the planning for command interpretation due to being more human readable. The primary third-party libraries used include OpenCV, RasPicam, PiGPIO, and the BCM library associated with the chip the Raspberry Pi uses.

B. Controller-Side

When the controller is powered on, a short bash script is used to launch two programs that both handle communication with the submarine. Also, alongside with the controller booting up with a bash script there will be three executables to run on the desktop should there be a crash on the software. The executables will be capable of stopPing the programs on the controller, stopPing the program, and then restarting it, and run both programs in the proper order should they not have been started or were cancelled by the user. With the three options available the user will be able to utilize the touchscreen and debug the controller software on the spot without needing to pull the submarine out of the water.

The first process to be ran is the controller program that acts as a listener for the remote controller. The program reads in the command that is being used on the controller then runs it through a case check that performs a specific task depending on the binding that is pressed. For example, if the left or right joystick is moved around then the controller program interprets the direction and intensity of the movement then sends the formatted command as a UDP packet to the Submarine. With some buttons the controller sends a UDP packet only containing the positive variable of a button press rather then the zero from when it releases.

The secondary command that is ran on startup on the controller Pi is the Imagery code that was designed to interpret the values that are sent from the sub such as its depth, temp or if the water bladder is full or not. The program first receives a UDP packet that contains an

image from the submarines camera and then displays it in the window. After the image pathway has been initialized and set up, a secondary thread is set up for the submarine info. Once the controller receives the packet it then separates out the info and shifts them to the variables controller-side. Those variables are then used to be displayed along with the frames per second.

A function using a FIFO was created to allow for the two programs on the controller to communicate a button command that allows for a user to toggle the information display on sub feed. This variable display lets a user see more of the camera feed if they do not believe they need to see the sub data. It also allows the user to constantly keep the sub data showing without taking attention from the subs control.

IX. HULL

The hull is arguably the most important part of the submarine, as it defines the aesthetics and protection of the entire project. Ensuring that the hull would be strong enough to resist the pressures at the target depth while within our budget and manufacturing abilities was a major focus.

The dome was the center of the hull design. It is used as the viewing window to allow the camera to see the environment outside. It was perceived as being the weakest link of the submarine, with initial research indicating that the manufacturing process could result in the dome being as thin as 1.5875 mm (1/16 in). This led to efforts to calculate a first-order approximation of the strength of all parts of the submarine, and defining the submarine to be composed of four shell sections. These are a half-dome, ogive, cylinder, and truncated cone.

The required dimensions for the dome were calculated assuming a fixed diameter of 127 mm (5 in.) and due to a rigorous pressure-analysis being outside the scope of this project, the classical equation for a thin elastic half-dome below was used.

$$(1)\ P = \frac{2*E}{\sqrt{3*(1-v^2)}} * \frac{h^2}{R^2}$$

Where v is the Poisson's ratio for the material, E is the modulus of elasticity, h is the thickness, and R is the radius of the shell's mid-surface (a point between inner and outer radius). Due to the wide range of possible values for the Poisson's ratio and modulus of elasticity, averages were used from the MatWeb online material database. Further note that although equation is used to represent elastic shells, assuming the hull of the submarine to be rigid should increase the pressure it can withstand and subsequently improve the depth that the submarine can survive at. We finalized our measurement based on the depth resulting from the calculated pressure.

The plexiglass dome was manufactured by EZ Tops World Wide, and a white-paper they published in collaboration with the Puget Sound Naval Shipyard assuaged concerns that it may fail. In fact, the report shows it may be our strongest part, as a smaller dome they tested in a specialized pressure chamber on a dome of similar dimensions only failed when under a pressure of 41.39 bar (600 psi), which is a depth of 1200 feet and well outside our depth range [3].

Similar calculations for pressure were made using the following algebraic equations:

$$(2)\ P = \frac{2.42*E*(\frac{t}{2*a})^{\frac{5}{2}}}{(1-v^2)^{\frac{3}{4}}*(\left(\frac{L}{2*a}\right)-0.447(\frac{t}{2*a})^{\frac{1}{2}}}$$

$$(3)\ P = \frac{0.92*E*\gamma}{(\frac{L}{pbar})*(\frac{pbar}{t})^{\frac{5}{2}}}$$

Equation 2 was used for the buckling pressure of a cylindrical shell, where t is wall thickness, a is the mean radius of the shell, E is Young's modulus, v is Poisson's ratio, and L is the unsupported length of the cylinder. Equation 3 was used for the truncated cone shell [4], with E being Young's modulus, gamma being an estimated correlation factor, L being the slant length of the cone, and t being the wall thickness. P-bar uses this equation to determine the average radius of the cone:

$$(4)\ pbar = \frac{r_1+r_2}{2*\cos(\alpha)}$$

where r1 and r2 are the radii of the small end and large ends of the cone respectively, and alpha is the semi-vertex angle of the cone.

Due to the expenses of professional 3d printing, the hull of the submarine was changed from being printed from PLA to using a purchased schedule 40 PVC Pipe from Charlotte Pipes. The Pipe is rated for handling 42 bar (160 psi), and is by far the heaviest part of the submarine. All of the other parts were designed in Autodesk Inventor 2019, and 3D printed with PLA plastic on a Creality Ender 3 printer on loan from a team friend.

X. TESTING

To Put the submarine to the test the first check to hit was component testing, where all the components were tested before being assembled. During the component testing the small subsystems such as the ballast system and the controller submarine communication line were analyzed to ensure they would work correctly once inside so that should an error arise, the components would be in a safe position for extraction. Once all core components of the submarine were assembled and tested, they were then loaded into the framing and tested as a whole system above land. For the land test the submarine was propped up into a mount that allowed the propellers to run without impediment. While propped up commands such as forward movement, turning movement where able to be demonstrated at varying speed meaning the commands where properly coming through. The pan tilt was demonstrated using the controller's right joystick and followed the mapped plan of looking up, down, left, right. With the full system running and demonstrating the capability of a fully operational system the submarine could be relocated for water testing. For water testing the submarine was placed on the surface of the water in a pool with a depth of 3 meters. Once placed in the water forward movement was demonstrated as the submarine could travel through water with have the propulsion being active. Once movement was demonstrated the ballast system was tested next by drawling in water and submerging to the bottom and then water was released to cause the submarine to float to the top. The camera stream and Pan/Tilt were noted for working via fps counter and the ability to pan the camera.

XI. CONCLUSION

For the time that the group has been together and formed, much time has gone into carefully planning each step of the process and selecting each component has been unanimous. Without forming a group wide decision as the project progressed the formation of the Submarine may have been less then anticipated. The group worked as one coherent cohort and welcomed the difficulties that was challenging the open water and allowing the risk of everything being destroyed. With the submarines last seal being put on it was the groups final goodbye and leap of faith into testing what was created.

THE ENGINEERS

Jacob LaVoy is a 23-year old electrical engineer who is eager to start working in the field of green energy for a cleaner, more powerful, future.

Madison Melton is a is a 22-year-old Electrical Engineering student and also an intern at Aerojet Rocketdyne Coleman Aerospace, where she will continue working full-time after graduation. She enjoys working with missiles and wants to continue learning about their components and layouts in depth.

Avery Mills is a 21-year old ambitious engineer in the making ready to get into the field and be introduced to the hardships of the world. He plans to follow an internship with a local engineering company and join the ranks as an engineer.

Joshua Nichols is a 22-year old graduating Computer Engineering student who hopes to get a job at a large company such as AMD or Microsoft. He has recently expanded his horizons with machine learning and remotely operated vehicles.

ACKNOWLEDGEMENT

We want to thank Dr. Wei first and foremost for his assistance in guiding us to success as our coordinator. We also wish to thank our review committee, for joining us for the Q&A session on July 28th. We wish to thank Advanced Circuits for their fabrication services. We would also like to thank John Raab for allowing us to use his 3d printer to model our Submarine and print it ourselves. Most importantly, we wish to thank our family members and friends, especially Stephen Nichols, Crispina Nichols, Robert Mills, and Michael Bristol for their assistance and motivation. The project could not have been done without all the people listed above.

REFERENCES

[1] https://www.RaspberryPi.org/documentation/hardware/camera/

[2] https://learn.sparkfun.com/tutorials/light-emitting-diodes-leds/get-the-details

[3]https://www.eztopsworldwide.com/EZ-TOPS%20Dome%20Test.pdf

[4] https://ntrs.nasa.gov/aPi/citations/19690014753/downloads/19690014753.pdf