Gonzaga University Robotics: RoboSub 2021

Aaron Weber, Alan Poblette, Alec Wilson, Alexander Delgado, Audrey Stevenson, Blaine Atkins, Brady Norrish, Brandon Takahashi, Carter Mooring, Drew Bogdan, Drew Feucht, Emily Ellwein, Eric Av, Ethan Cady, Ethan Higa, Gage Bockenstedt, Graham Laird, Jalen Tacsiat, Jasmine Fisher, Jason Lunder, Jason Yager Jeffrey Henion, Joci Anderson, Joey Macauley, Loren Anderson, Mathew Hasting, Nathan Frojelin, Nick Linthacum, Olivia Schaefer, Preston Ernst, Ryan Hardinge, Ryan Leahy

***Abstract*—The Gonzaga University Robotics team has competed in the international Robosub competition since 2018, both in person and virtual due to COVID-19. Every year we have been able to discover new goals and areas of improvement for ourselves. Leading up to this year's 2021 competition, the goal was to surpass previous limitations to prepare our sub for a more advanced approach to the Robosub competition. Our team is looking forward to our best year yet when we are able to showcase our sub, Terrapene, in person again. This report documents an in-depth technical review of the work GU Robotics accomplished as a team on the mechanical, electrical, and computer science systems during the 2020-2021 school year.**

I. Competition Strategy

Our competition strategy continues to be reliability through simplicity. We've been working on low-level battery management, leak sensing and component layout to make pool time more reliable. We have focused on increasing reliability on tasks we accomplished in person in 2019. We acquired a RealSense depth of field sensing camera which when combined with long-range utilization of the IMU should allow us to pass through the gate on every try instead of just 50% of the time. We've continued to develop our image recognition technology in an attempt to detect and track buoys from further away. We've also worked on implementing a hydrophone system to increase the number of tasks our sub could accomplish. A hydrophone system would allow us to surface in the octagon and attempt the torpedo task. Thus, we also began work on a torpedo launching system. Going in to competition this year, our goal was to reliably pass through the gate with a coin flip, and bump in to a buoy, preferably with the chosen image. If we finished the hydrophone system we would also attempt to surface in the octagon and complete the torpedo task.Prior to COVID-19 we were prioritizing creating a pre-qualification video so that we could use all our time in the competition pool to focus on testing tasks instead of just qualifying.

II. Vehicle Design

A. Mechanical

As of right now we have been building off of the sub, Terrapene, that was originally created in 2019, and using the knowledge of this system to work in additional components that will strengthen our competition strategy. The strengths of the box design included in-water stability, reliable waterproof sealing, ease of access to electronic components, and increased space for upgraded computer systems. Although the rectangle is not an ideal hydrodynamic shape, the sub travels at such low velocities that drag forces are negligible. Our 2021 mechanical team primarily focused on the research, design, and implementation strategies of several new systems. This includes a new camera, two extra thrusters to give us multiple axis of movement, and a torpedo launcher with competition appropriate torpedo's. Due to COVID-19 restrictions, we were unable to test these systems to our satisfaction and decided not to add them to our submarine for this year.

1) Hull: The 2021 hull design features an off-the-shelf, IP68 rated, underwater enclosure from Polycase. An on-campus laser cutter was used to create a custom electronics tray with interlocking acrylic parts, and tolerances are within .05 mm of what we expected. This tray is inside the hull and houses the batteries, and the mounted electronics on the middle level. The tray has a handle for easy removal and access to the components. It also serves as a structural brace to protect the sides of the box against deflection from water pressure. Although the sides and bottom of the box are opaque polycarbonate, the lid is clear so indicator LEDs can be seen. Wires from the components are routed through the lid of the box using Blue Robotics Cable penetrators. We designed and 3D printed custom brackets that allow the lid to clip onto the side of the box while we work on the internal electronics without causing damage to the wires or someone having to hold it open. Initially, a 10"x10"x12" box was used, but testing revealed that all components could fit in an 8"x8"x10" box and score bonus points by minimizing ballast weight. If future developments require more space, the larger box could be used and a new electronics tray could be cut to those proportions.

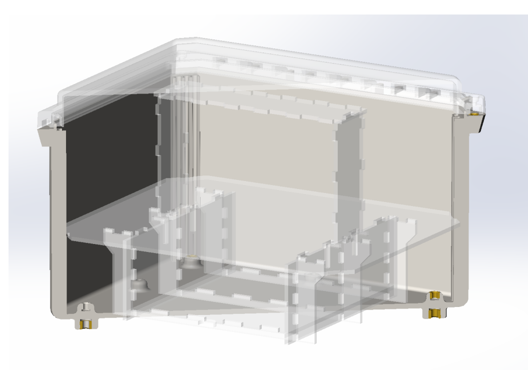

Fig. 1. Custom Electronics tray in the box

2) Frame: The 2021 design of the frame features plasma cut aluminum side panels that allow for precise positioning of the motors and the hull, as well as providing extra protection for components if the sub bumps into obstacles. The custom hole pattern was modeled specifically to work with the 80/20 aluminum t-slot mounting brackets and the custom 3D printed motor mounts to allow for near-universal motor positioning. Pool test experience revealed that the aluminum corrodes significantly over time. Thus, the team installed a sacrificial zinc anode to the frame in order to prevent this problem.

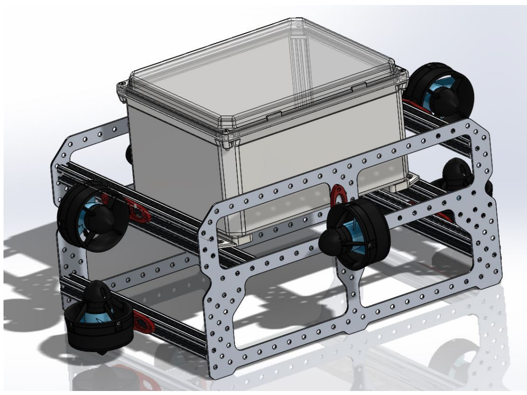

Fig. 2. Fully Assembled Frame

B. Electronics Systems

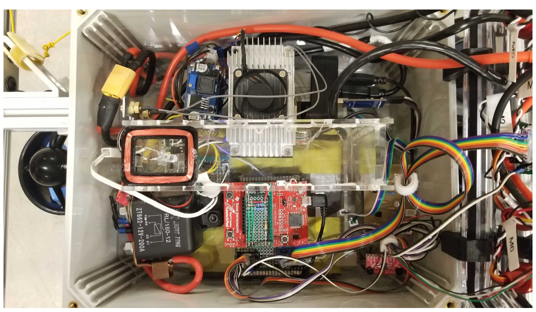

Fig. 3. Electrical System

1) Monitoring System: Previously, all of our monitoring capabilities have been a part of the mission computer code running on the Jetson, and only visible through the GUI when connected to Terrapene. Thus if Terrapene was in autonomous mode or the Mission code or TI crashed, we lost all monitoring. To remedy this, the electrical team built a separate arduino-based system exclusively for monitoring. It interfaces with a 16x2 LCD screen that will be mounted under the transparent handle so that it's easily visible through the lid. The screen displays rotating information about battery voltage level and current draw, and more information can be added to the cycle. It also connects to the leak sensor and will flash a message that a minor or critical leak has been detected. The arduino may eventually be used to trip relays on a leak event as well (or this may be handled with transistors). Communication between the TI and arduino is in progress to gather more information when the TI is active.

2) Leak Sensor upgrades: The leak sensor board we built has two independent probe headers which activate two onboard LEDs and a signal header when current is detected across probe leads, indicating the presence of water.

Previously the probes had been wired in parallel, and connected to the bottom of the sub requiring them to be disconnected whenever the tray was removed (a common occurrence). This year we wired each probe to a separate circuit but on a shared ground. The ground probe was mounted to the bottom of the tray with the minor leak probe beside it, while the critical leak probe was mounted to the side of the tray half an inch above the bottom. This way if some water gets in, it will bridge the ground and minor leak probes, while if a significant amount of water gets in, it will activate the critical probe. This will eventually allow us to take different action depending on each case (for example: issue a visual warning, or surface and cut power).

Additionally, development was started on adding the SN74LS279ADR SR latch to the leak sensor circuit. This will keep the circuit in the detected state until it is reset with a button press, in case water bridges the contacts and then sloshes away. This is also our first experiment with a surface mount device, which should allow us to shrink our future circuit boards.

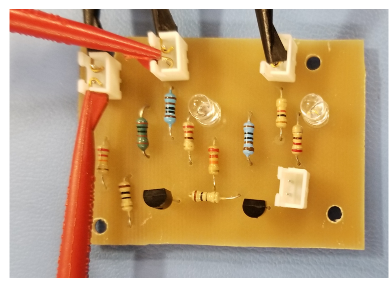

Fig. 4. Leak Sensor Board in testing

3) Hydrophones: [This project has been slowed by Covid-19 due to limited pool access]. The Hydrophone System would allow Terrapene to locate the location of a ping in water and travel to the location of the ping. The hydrophones themselves are piezoelectric mic elements put into a waterproof enclosure. Each one was made in house as opposed to being bought in order to reduce the cost of the project. One hydrophone would be put in each corner of Terrapene. This would lead to having four hydrophones in total. Each hydrophone would out put its signal to Texas Instruments lm386. Each lm386 was setup to have a theoretical gain of 200V/V. This was done so that the signal can be process by the microprocessor.

This year we have been focused on filtering, and began experimenting with the MAX 286 integrated circuit. The adjustible center-frequency of this device will enable greater flexibility in tuning the circuit from software which may help eventually earn more points with a random frequency during competition.

4) Hull Illumination: Our procedure for deploying Terrapene after the hull has been open involves slowly placing it underwater and visually checking for leaks. Due to lighting conditions, it is often difficult to see streams of water coming

Fig. 5. Hydrophone Assembly

Fig. 6. Final Hydrophone Build

down the side, so work has been started on directing white LEDs to all sides of the hull to better illuminate them for visual leak detection

5) Hardware battery indication: Several attempts were made to create a battery monitoring system similar to that of portable batteries with 4 LEDs to indicate battery level. Initially a circuit based on zeener diodes was attempted, but additional voltage drop across LEDs or transistors made zeener diodes not transition states at a precise enough voltage. Comparators were also considered, but the number necessary created a physically larger circuit than we were hoping for. Perhaps in the future we can use SMD components to shrink the size and build this circuit, but for now the monitoring LCD is accurate enough.

C. Mission Software

1) Mission Computer: The Mission Computer software team focuses on controlling the actions of the sub and communication between the different systems running on the sub. The center of the mission computer is a Java program that is executed on an NVIDIA Jetson TX2. This mission computer program communicates with the TM4C123GH6PM microcontroller to control motors and receive sensor data, and with a Python program to interpret camera data using OpenCV. The microcontroller communications are done through UART, and the Python communications are done through a UDP server. In both cases, the Java program sends data through a communication protocol using a set of enums called SendTypes and ReceiveTypes. These enums are part of a system we developed that sends a specific character id for what type of data is being sent (such as a PWM value for a specific motor or a desired depth value) along with the data. Whenever a SendType is sent the data being sent and the timestamp of when it was sent are recorded, so that they can be viewed after testing to find what commands were sent at what time.The microcontroller and Python send data to the Java program in the same format, and the Java program has a list of ReceiveTypes that tell what type of data the character id corresponds to, and these are recorded in the same manner as the SendTypes.

The Mission Computer program was designed to handle all decisions related to autonomy, relying on the microcontroller for sensor input and the OpenCV code for visual input. By moving the code for interfacing with the camera and sensors to different programs, the Mission Computer can focus on autonomously controlling the sub and request sensor and camera data as needed. For testing and debugging, there is also a graphical user interface written in C# that runs on a separate computer and can connect to the Mission Computer program over ethernet to display sensor data and manually control the sub.

2) Mission Control: The mission control aspect of the sub is currently handled through parsing and executing JSON scripts. These scripts contain a series of steps and actions that the submarine will take, given the right condition is met. This script can be best thought of as a linked-list where the mission computer only traverses to the next node when all of its exit conditions have been met. This mission script allows us to quickly modify the behavior of our submarine while allowing it to autonomously execute a set of instructions. Mission scripts consist of a set of nodes, with each node having actions and exit conditions. Actions are values that are sent to the microcontroller or Python program, such as motor PWM values or setting and enabling a PID loop. Exit conditions are the conditions that must be met before the mission can move on to the next node, which can include simple conditions like a certain amount of time elapsing or more complex conditions such as holding a certain depth or heading for a period of time.

D. Embedded Systems

The goal of the Embedded Systems team is to provide an interface for our Mission Computer to communicate with our motors, sensors, etc. To do this, we have a Texas Instruments microcontroller that uses protocols such as I2C, UART and PWM to communicate with the sensors and motors while providing feedback to the Mission Computer.

1) Microcontroller Unit: The microcontroller used is the TM4C123GH6PM. This unit was chosen for its widespread support. This made it possible to prototype and develop functionalities in a timely manner. The microcontroller's capabilities were accessed through the widely supported TivaWare drivers. The drivers made it easy to use the various peripherals provided without extensive knowledge in the microcontroller's architecture. The flexibility of the Nested Vectored Interrupt Controller allowed for a responsive system. The use of interrupts provide an illusion of concurrency which is a key component of the embedded system.

2) Control Loops: The interrupt service routines provide 3 main control loops. The main function loop, the UART receiving interrupt service routine, and the real-time interrupt service routine. The UART receiving interrupt service routine is triggered when a character is received on the UART channel. The main function loop controls prototyping and specific function testing, while the real time interrupt service routine executes the PID control loop that alters motor values to achieve the given set point. The main program flow is illustrated below.

a) UART Interrupt Service Routine: This UART Interrupt is triggered when Mission Computer sends a communication string to the microcontroller. These communication strings are 6 bytes long; each containing 1 byte as an identifier character, 4 bytes as a standard IEEE 752 floating-point number and the final byte as our predetermined "end of transmission" character: '∼'. If the identifier character is '*', then the command is forwarded to the Main Function Loop and will be processed there. Otherwise, the this Interrupt Service Routine will process it and perform the specified task. These tasks include changing the set-points for the system's PID controllers.

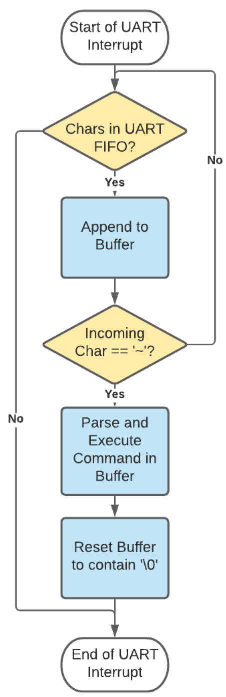

Fig. 7. The UART ISR Diagram

b) Real-Time Interrupt Service Routine: This routine is a periodic interrupt that is triggered every 100ms. Once it is triggered, the routine will toggle the TM4C123GH6PM on-board LEDs to indicate that the system has entered the interrupt. After toggling, the interrupt will then set flags to allow the Main Function to perform the primary system tasks such as reading peripherals and calculating PID values.

c) Main Function Loop: The loop is comprised of four sections: processing Mission Control commands, reading system peripherals, computing PID output values, and updating Mission Control with the current state of attached peripherals (see Fig. 7).

Upon receiving commands from Mission Control via the UART, the Main Function Loop receives the commands from the UART Interrupt Service and enacts them. The commands are used to request the microcontroller to perform tests that involve it sending Mission Control the current states of the peripheral being tested. Thus, these commands will only be sent for debugging and prototyping purposes, only.

The microcontroller will then read from the following devices when triggered by the Real-Time Interrupt Service Routine: the TM4C123GH6PM builtin 10-bit ADC, the IMU, and the depth sensor. These readings are used to monitor the battery voltage, and monitor the system state when achieving the target heading and depth during autonomous operation.

The next section of the Main Function Loop updates the PID control system output values to be used to manipulate motor speed. Currently, we use our PID control system to maintain a specific depth and heading of the sub during autonomous movement. We chose to use a PID system due to its ease of implementation as well as the ease of tunability of the control system parameters. Figure 8 depicts how the PID control system works. Based on the current PID output values and the computed sensor values, new PID output values are computed and used to manipulate the motor RPM to get closer to the target value.

The last section of the Main Function loop updates Mission Control regarding the current state of the system. The microcontroller will communicate, via UART, floats representing peripheral readings and target states of PIDS. This is used in conjunction with the Mission Control GUI (refer to page 3 under the "Mission Computer" subsection).

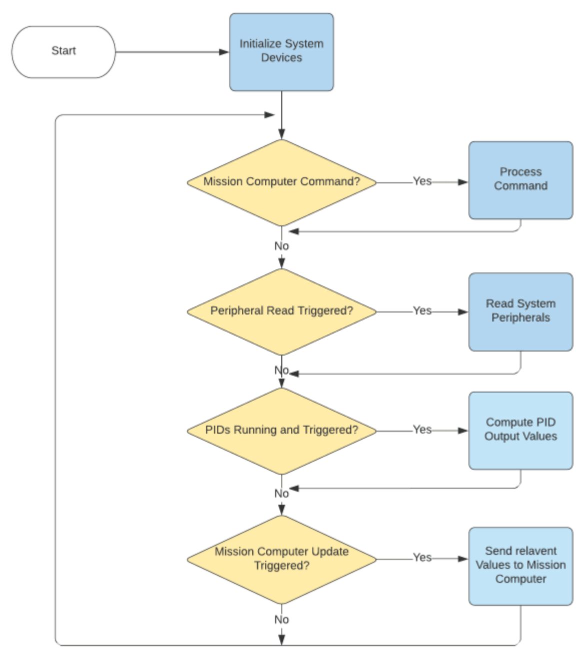

Fig. 8. The main control loop of the microcontroller

3) Sensors: The TM4C123GH6PM also interfaces with a large portion of the sensors on the submarine. These include the depth sensor, accelerometer, gyroscope and magnetometer. The BNO055 inertial measurement unit (IMU) is used to

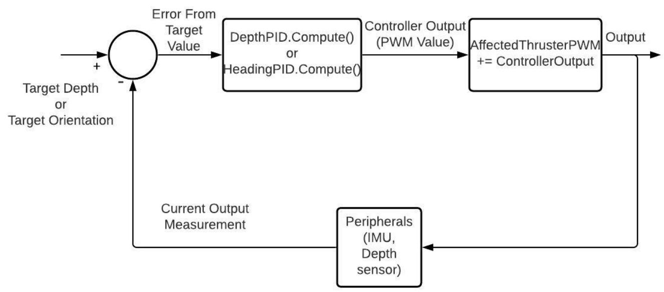

Fig. 9. PID Feedback Control Loop

provide acceleration, gyroscopic and magnetic heading data while the MS5837 is used to provide pressure data. The BNO055 is a standalone IMU while the MS5837 is housed as the Bar30 as provided by BlueRobotics. Both the MS5837 and the BNO055 interface with the microcontroller via I2C communication. The IMU is able to internally calculate Euler angles based on its current orientation, which we use to orient the sub in the pool Currently, we are only looking at the two dimensional X-Y plane for the Euler angles, and the depth sensor provides the Z offset.

4) RFID Inputs: Our team worked on a wireless method to send signals to the submarine while in untethered. We added an RFID sensor which communicates with our Mission Control software through UART. This RFID sensor was specifically chosen to for its low frequency of 125kHz, allowing it to communicate even through water. The RFID sensor actively reads RFID tags that come within 5cm. If the RFID tag's identification digit matches the stored values within our Mission Control, it triggers a programmable action. This method is used to trigger the shut-down of the microcontroller or the start of various mission scripts. The RFID sensor provides a unique and flexible way of communicating with our submarine when untethered.

III. Experimental Results

A. Embedded Systems Minibot

We created a small robotic car for testing and debugging system software early in the year. The car operates similarly to how the sub operates in terms of heading and horizontal movement. This minibot is used whenever software needs to be tested. Using this platform, we can develop and test new drivers for potential peripherals we want to add later on. Doing so has optimized time spent in pool tests by decreasing debugging time and enabling us to focus more on fine-tuning sub-operations. Although the car has limitations compared to the sub, such as lack of z-axis motion and exposure to friction, it has helped us prepare for the pool tests by enabling us to formulate more detailed plans ahead of time. Some of the stuff tested include heading as well as depth, which was simulated with a potentiometer.

1) Torpedo Servo Testing: One of ways the Minibot helped test software was with developing the servo driver used for the torpedo launcher. The embedded software responsibility for the torpedo launcher is controlling a servo that allows it to launch. The Minibot allowed us to debug the servo control in an isolated environment, allowing easier and more time efficient debugging since we weren't working with the entire sub system.

B. Field Programmable Gate Array (FPGA) Development

One of the challenges that we face with growing our embedded system is the amount of available pins on our current microcontroller. Therefore, to combat this and add more functionality to our system, we began experimenting with the DE0-Nano - Altera Cyclone IV FPGA starter board. With this board, we have implemented (using Verilog) a general-purpose PWM module and are currently developing a means of communicating with the board via UART. The PWM module is configured to accept a maximum period and width of 50MHz. The UART module is configured to have a baud rate of 115200 bits/second and uses 1 stop bit. The communication protocol designed thus far is made up of 5 bytes: 1 byte holding a command to indicate what the FPGA must do, and 4 bytes that contains data that it needs to execute the command.

C. Mechanical Torpedo Device

The 2020 mechanical team experimented with adding a torpedo launcher to the sub, with the goal of being able to shoot a torpedo through the competition opening to score points. Initially, the discussion revolved around the propulsion system that would be used to fire the torpedo, and it was decided that the best option was a spring-loaded launcher. Materials, actuation, and the possibility of x-y axis movement independent of the submarine was also discussed at length before an initial design was created.

After the development of the initial design, a crude prototype was created using PVC pipe and spring. This small hand-actuated prototype was tested. Observations during this test were considered along with the complexity advantages associated with 3D printing, and PLA plastic was the chosen as the material for the launcher exterior. The launcher in its current configuration is made of two separate printed parts joined by bolts, and the release mechanism is motor powered with the torpedo locked in a revolver type cylinder.

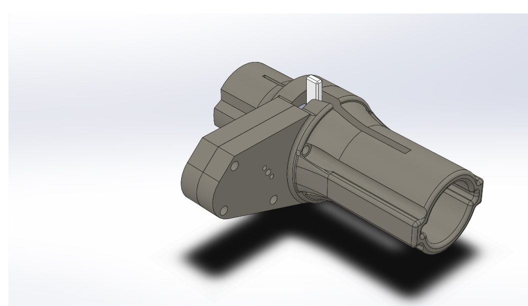

Fig. 10. Torpedo Launcher 3D Design

D. Hydrophones

Though slowed by Covid-19, development has continued on adding hydrophones. We built our own hydrophones in order to limit cost. The hydrophone was made by putting a piezoelectric in a PVC pipe bushing. A aux cable was then soldered onto the mic element and pulled out the smaller end of the PVC pipe bushing. A liquid tight cord grip was put on to the aux cable and screwed into the other side of the pipe bushing. A foam bumper was then put onto the side of the mic element that captures sound. Finally, a two part epoxy was used to fill in the pipe bushing in order to protect all of the electronics. The hydrophone was tested by putting it into a sink with a waterproof Bluetooth speaker playing a constant tone. An oscilloscope was then used in order to see the signal and confirm that the hydrophone was working. A lm386 integrated circuit was used in order to amplify the signal coming from the hydrophone. The circuit was built in order to get a theoretical gain of 200V/V in accordance to the data sheet. This system was then tested in the pool where a gain of 500V/V was observed. This test was done using a dog whistle in order to test at higher frequencies and to keep the cost of testing down. Before COVID-19 restrictions, our aim was to build a system that involved a push-pull amplifier and an arbitrary waveform generator using an cheap high frequency speaker in order have more control over the frequencies at which we could test. To isolate the pinging frequency, the MAX 268 integrated circuit bandpass filter was used for experimentation. The filter has an adjustable centering frequency and bandwidth making it adjustable to a wide range of frequencies. The integrated circuit will be powered by the TM4C microcontroller, sending the filtered data to a general purpose input/output (GPIO) pin. More experimentation will be needed to explore its usability.

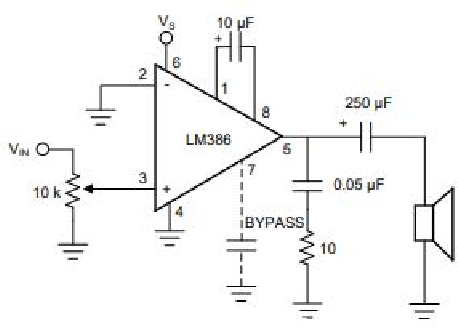

Fig. 11. Amplification Circuit

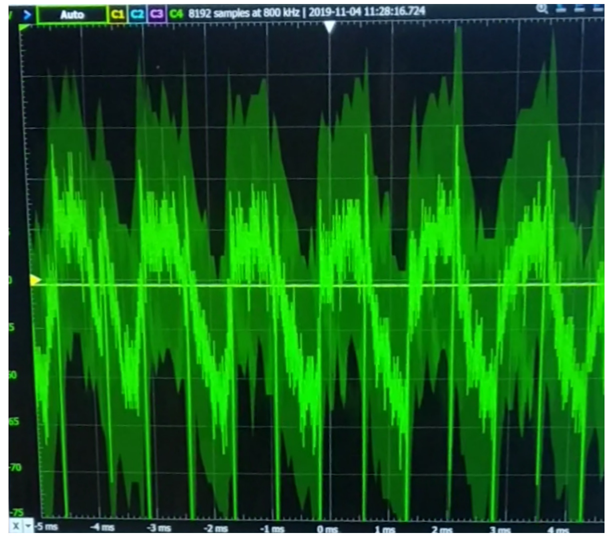

Fig. 12. Initial Hydrophone Test Results

creation, plugin creation, continuous integration via a Docker workflow, and developed a workspace using catkin.

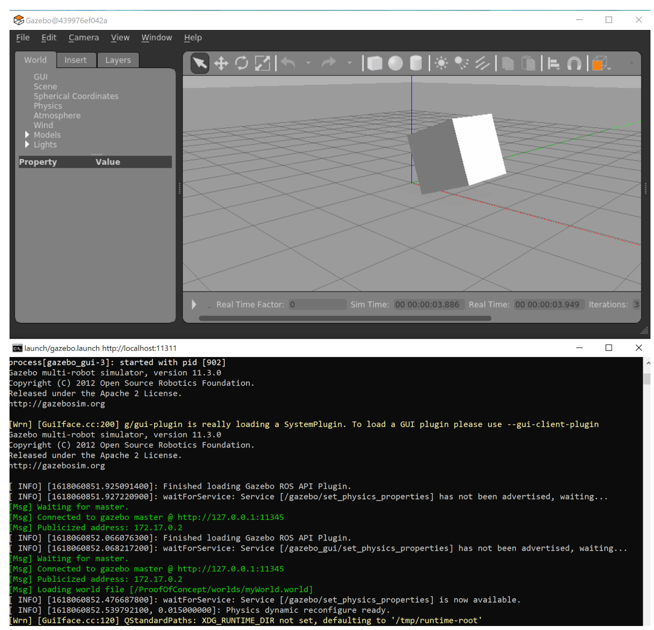

Fig. 13. Cube Model being pushed in a simulated environment

E. Navigation and Control Model

The main project for the Machine Learning team has been the Navigation and Control Model. We have been building a simulated environment in which to train and test a model to control the sub. This year's work consisted mainly of creating this environment using Gazebo, ROS, and Docker. This was made as a proof of concept in order to master the tools used, and to prove that it worked, we created some simple plugins to move a cube around the simulation space. We tested model

Using the environment we built this year, we will be developing and training a model to control the simulated submarine using Gazebo/ROS. This model will likely be based on a reinforcement learning model, but we have further investigation and testing to do to determine the best approach for training and development. The goal is for this model to be able to replace and improve upon aspects of the current Mission Computer control system.

ACKNOWLEDGMENTS

First, we would like to extend a huge thank you to Fischer Connectors and Novelis Aluminum, whose sponsorship and generosity have allowed our team to innovate and design freely. We would also like to thank Dr. Timothy Fitzgerald for being our club advisor, hearing our crazy ideas, and always pointing us in the right direction. And to Dr. Aaron Crandall, who has agreed to be our advisor for the upcoming year, who we look forward to working with. Thank you to Janean Schmidt who helped format our website this year. And the biggest thank you of all to the students of this team, alumni that paved our way, and the current students whose energy and passion to know more and do more drives this team. We are excited to see what they will do with the club. Lastly, thanks to the students of the other RoboSub teams for your knowledge, camaraderie, and freely given help and advice.

IV. REFERENCES

br3ttb, “br3ttb/Arduino-PID-Library,” GitHub. [Online]. Available: https://github.com/br3ttb/Arduino-PID-Library

Texas Instruments Incorporated, “TivaWare™ Peripheral Driver Library,” Texas Instruments, March-2013. [Online]. Available: https://www.ti.com/lit/ug/spmu298e/spmu298e.pdf. [Accessed: 31-Jul-2020].

K. Joy, J. Hamilton, M. Jewell, and I. Babb, “COSEE TEK University of Connecticut Simple Hydrophone Design ,” Northeast Underwater Research, Technology & Education Center, 07-Jul-2017. [Online]. Available: https://www.nurtec.uconn.edu/wp-content/uploads/sites/287/2016/08/COSEE-TEK-Simple-Hydrophone-Material-List-Fabrication-Instructions-V4.2-7-7-2016.pdf. [Accessed: 31-Jul-2020].

Texas Instruments, “LM386 Low Voltage Audio Power Amplifier,” Texas Instruments, May-2004. [Online]. Available: https://www.ti.com/lit/ds/symlink/lm386.pdf

APPENDIX A: COMPONENT LIST

TABLE I

Component	Vendor	Model/Type	Specs	Cost (if new)	Status
Buoyancy Control	N/A				
Frame	8020	1x1 in. T slot	20 feet, cut to length	not new	
Waterproof housing	Polycase	YQ-100806	10x8x8 inches	not new	
Waterproof Connectors	Fischer Connectors	UR 01	8 contacts	donated	
Thrusters	Blue Robotics	T200		not new	
Motor Control	Texas Instruments	TM4C123GH6PM		not new	
	Intel	Cyclone IV (EP4CE22F17C6N)		$149.95	In Development
High Level Control	Nvidia	Jetson			
Actuators	N/A				
Propellers	Blue Robotics	Propellor Set		$5	
Battery	Turnigy	4S1P 14.8V 20C Hardcase Pack		not new	
Converter	N/A				
Regulator	N/A				
CPU	Nvidia	Jetson		$400	
Internal Comm Network					
External Comm Innterface					
Programming Language 1	Java				
Programming Language 2	C++				
Programming Language 3	Python				
Programming Language 4	Verilog				In Development
Compass	Adafruit	BNO055		$34.95	
Inertial Measurment Unit	Adafruit	BNO055		$34.95	
Camera	ELP	ELP-USB500W02M-L36	3.6mm fixed lens	not new	
Hydrophones	self-made			$27.55 x 4	
Algorithms: Autonomy	Self-made				In development
Algorithms: Simulation	self-made/open source				In development
Open source software	Open-Source				In Use
Team Size	25				
HW/SW Expertise Ratio	11:14				
Testing Time: In Water	15 hours				

Fig. 14. Service Event 2021

APPENDIX B: OUTREACH ACTIVITIES

A. Community Involvement

Being involved with Gonzaga's surrounding community was difficult this year with the pandemic, but we still found ways to help. We were able to pair up with the local organization, Meals in the Margins, to put on a socially distanced care package event. Our club members assembled over 200 care packages containing toiletries, masks, hand sanitizer, and snacks to be distributed to Spokane's houseless population. We also prepared information cards about mental health and housing resources in Spokane, to be distributed to the members of our community.

B. Educational Outreach

We had the chance to participate in an engineering event for prospective students at Gonzaga. This virtual event for high school students around the nation allowed us to answer questions and showcase our club. One of the cooler features we enabled at this event was an opportunity to control our Minibot through Zoom's remote control feature. Prospective students were able to drive the Minibot using their keyboard while watching it move in a classroom. Even though some students may not attend Gonzaga, it was very special for our team to be able to inspire future generations on what their college experience could look like.