AquaPack Robotics Technical Design Report

Marcus Behel*, Gabriel Chenevert†, Elizabeth Gillikin†, William Kelso*, Xingjian Li*,
Christopher Mori*, Alex Pendergast*, Bennett Petzold*, Donald Shrader†,
Tajah Trapier§,Robert Williamson‡

*Electrical and Computer Engineering, †Mechanical and Aerospace Engineering
‡Biomedical Engineering, §Materials Science and Engineering

***Abstract*—AquaPack Robotics at NC State University (formerly known as Underwater Robotics Club at NC State) is returning to RoboSub with SeaWolf VIII in 2023. After successfully qualifying at RoboSub 2022, several overhauls and additions were made to the system. SeaWolf VIII has in a way become new, with overhauls reflecting lessons learned from the previous competition, and additions reflecting the team's ambition to perform more than in 2022. Our priority was to continue iteration of SeaWolf VIII such that it is capable of performing complex tasks reliably, whilst also making it friendlier for new engineering students. SeaWolf VIII aims to attempt a majority of tasks, with emphasis on navigation based tasks. The team sought to overhaul our previous off-the-shelf locomotion controls system with a completely custom control board. In parallel, our full navigation system has been expanded with the addition of full passive SONAR. Sweeping changes to the software architecture were made towards a custom Java-based framework, mirroring the language taught in foundational computer science courses at NC State. This year is the first inclusion of manipulations systems. Dropper and torpedo systems have been installed on the robot, with the aim to attempt manipulations tasks for the first time. Overall, this system has proven to be a stable platform suitable for continuous iteration, enabling development of more complex systems. This year's submission, with all overhauls and additions, is demonstrative of just that.**

Fig. 1: SeaWolf VIII

I. Competition Goals

A. General Strategy: Reliability

The major focus this year has been improving the reliability of our locomotion system. The team has replaced off the shelf flight controller in favor of a control board designed to our eight thruster system to allow highly reliable control of SeaWolf VIII in 6 degrees of freedom. Movement is a vital component of each task and high accuracy is required for navigation in a competition run. Instead of previously used ROS based approaches we have built a new custom software architecture using Java. This new software approach has improved the ability of the team to make gradual improvements to the systems and easily onboard new members to the system.

B. Destination

The competition run starts with successful navigation through the gate. With the reliability of our 6 degrees of freedom locomotion, navigating in a straight line and with style through rotation of yaw 720° is achievable. Using computer vision and image recognition with our front-facing camera should allow us to locate and align with the gate after a coin flip to determine and detect the destination selected for additional points in other tasks.

C. Start Dialing

Utilizing the downward facing camera with computer vision SeaWolf VIII will detect and navigate using the path pointing toward buoy. Image recognition of the symbols will be used to dictate locomotion to tap symbols according to the destination selected.

D. Goa'uld Attack

Seawolf VIII has two spring based torpedo launcher modules mounted forward facing. Navigating to the torpedo task requires navigation through passive sonar. Due to the difficulty in navigation, reliability of operating the launchers, and accuracy of computer vision to align with targets this task is of lower priority during a competition run.

E. Location

Following the completion of buoy SeaWolf VIII will use the downward facing camera and computer vision to detect and navigate using the path facing the bins. Computer vision will be used to center on the bin corresponding to the destination dialed. This task takes priority in our strategy over torpedoes due to reliability of manipulation system testing and ease of navigation with path pointing to its location.

F. Engaging Chevrons

Once all other tasks are completed, to end the run SeaWolf VIII will be using the improved FPGA-based passive sonar system to locate the pinger and the downward facing camera to locate and align to the table in the center of the octagon. Without a grabber manipulation system the only task SeaWolf VIII can accomplish is surfacing inside to end the run.

II. Design Strategy

A. Electrical System

1) Main Electronics Board: The Main Electronics Board (MEB) is the central interface for the entirety of the communications, power, and sensor systems on SeaWolf VIII via a unified I2C bus on which it acts as the controller. All other electrical system boards are connected to the MEB via I2C as followers. Thus, MEB acts as a buffer between the rest of the robot and our on-board computer. This organization enables SeaWolf VIII to effectively manage various sensors situated throughout the main hull of the AUV, in addition to acting as an interface for an LED light bar, the system arming switch, and the system power switch. An important effect of this architecture is that it frees computational power on the on-board computer for CV use.

2) Power System: SeaWolf VIII is powered by two 4S Li-Po batteries each fused at 40A allowing for a total current draw of 80A. This two battery architecture ensures the current capabilities of the battery are well in excess of the vehicle's requirements, allowing for future expansion. Additionally, using two batteries ensures longer run-times of the vehicle.

To address safety concerns with connecting Li-Po batteries in parallel, SeaWolf VIII includes a load balancing circuit using ideal diode controllers to safely connect the two batteries in parallel. This load balancing circuit also provides reverse polarity protection. Additionally, to ensure safe operating conditions, the hulls containing the LiPo batteries each contain a standalone leak detection module to allow early detection of any leak into battery hulls.

After the load balancing circuit, power is provided to the system using two Solid State Relays (SSRs). The first SSR switches power to the entire system. This includes the computer, electrical system, acoustics system, and other peripherals. The second SSR is used to switch power to the thrusters (after the system power SSR; thus thrusters are only powered if the system is powered). The thruster SSR is enabled only if both a software arm signal is generated by the vehicle's computer and if the physical kill-switch is in the "armed" position. Importantly the kill-switch is connected directly in series with the software arm circuitry ensuring that electrical component failures cannot prevent the ability to kill the vehicle with the hardware switch.

Finally, power regulation is required for various components on SeaWolf VIII. These include the Jetson Nano, the acoustics system, and various electrical system boards. Power regulation on SeaWolf VIII uses a distributed architecture where each board has its own regulator. This allows rapid development of boards and integration with minimal disruption to the rest of the system. Each component requiring 5V power is provided a "UBEC" (5V buck regulator) running off battery voltage. This includes the acoustics system, the USB hub, and various electrical system boards. The Jetson Nano uses a more complex regulator architecture due to higher current requirements and increased sensitivity to voltage drops. First, power is regulated to 12V using a SEPIC topology regulator. This regulator can handle battery voltage drops without impacting the output much. Then, the 12V power is regulated down to 5V using a buck topology converter. This dual stage regulator architecture was selected due to the high current required by the computer being a poor choice for a SEPIC topology regulator. However, the SEPIC regulator is more robust given the occasional voltage drops experienced on the batteries due to thruster motion.

B. Locomotion Controls

Vehicle locomotion is handled using our custom control board. The control board is a custom motion controller using an Arm Cortex M4F microcontroller. It acts as a motion co-processor, allowing mission code running on the vehicle's computer to describe motion in various high-level schemes. This co-processor design ensures the computer is spending minimal processing time on motion and ensures control loop stability due to the deterministic nature of timings on the control board.

The control board was developed to control vehicle attitude using a Quaternion-based approach similar to what can be used with a quadcopter [4]. A Quaternion based approach allows numerically stable control of orientation in 3D space [6] without potential for loss of degrees of freedom that accompany Euler angles [8]. Added to this are a PID controller to maintain depth, along with tilt (pitch and roll) compensation to allow description of motion in a partially world-relative 2D plane parallel to the surface. This ensures that slight pitch or roll errors do not result in unexpected motions.

This form of motion description also abstracts the vehicle's nature to mission code, allowing the code to more easily be used on different vehicles or in a simulator.

C. General Mechanical System

The design of SeaWolf VIII is centered around control stability and modularity. To achieve control stability, an octagonal frame shape was chosen. This places the thrusters farther from the center of mass, increasing their lever arm to counteract SeaWolf VIII's comparatively high mass moment of inertia. It also allows for thrusters one through four to be placed in a "strafe" configuration, or at a 45° angle to forwards on Sea Wolf VIII enabling the four thrusters to all contribute to horizontal motion in any direction.

Modularity has been achieved in two ways. First, the frame of SeaWolf VIII features standardized hole patterns to facilitate mounting current and future systems without modification. Second, the frame features large bays on either side of the electronics hull accessible through hinged panels which

provide ample additional space for prototyping systems before their form-factor has been optimized.

D. Software Architecture

1) State Machine: The robot is controlled by an object-oriented state machine pattern in Java. Each mission (steps to complete a task) is represented in a series of states, and combined in a dedicated state machine. The series are then chained together into a full competition state machine, from establishing communications to surfacing Octagon. This pattern improves design by breaking logic into small units that can be individually tested and debugged, then combined to form the full competition logic.

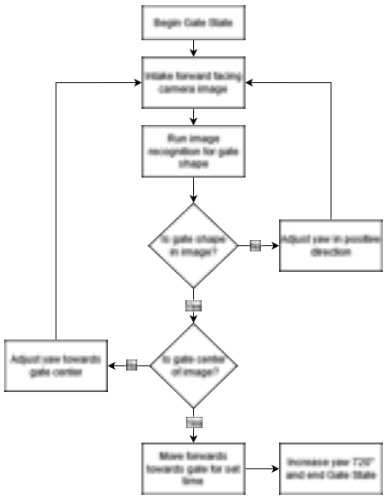
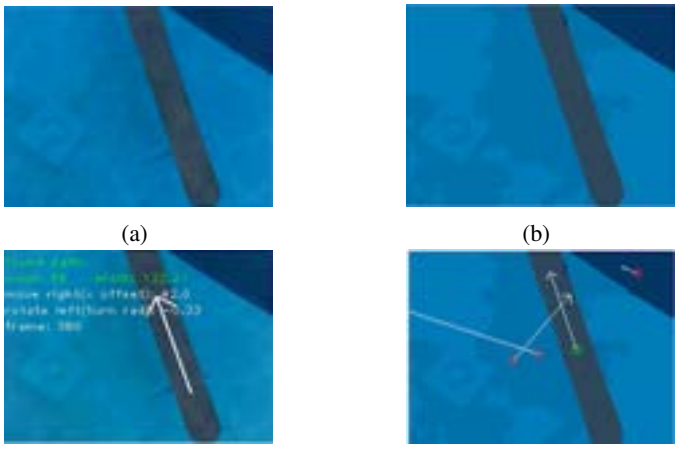


Fig. 2: Gate State Behavior Flowchart

2) Communication: Communications with external systems are handled in a discrete communication manager. It is allocated a static number of threads (to prevent system stalls from explosive thread growth) that handle sending and receiving messages from the control board. All of these serial messages generate an asynchronous task that returns true when an acknowledge is received, allowing a wait for success. This interface is abstract enough that a change of control board (especially to one that can't take commands in the exact order they are sent) would not require any changes to the high-level state machine. Currently it allows for the same communication between a real system and simulation.

The system runs a gstreamer connection to share camera feeds. There is one pipeline offering a UDP stream for live footage during robot testing. Another pipeline records to disk, so the real pool images can be used for vision model testing. The last pipeline is hooked into by the live machine learning code, feeding directly into the trained models.

3) Building and Deploying: The code is managed through Gradle with custom scripts for automatic testing and deploying code to the target platform. This ensures all members have the exact same environment and every step is clearly documented. On the system itself, a SystemD script is used to automatically launch logic during competition. Otherwise a ssh connection is used for manual control.



(a) (b) (c) (d)

Fig. 3: Comparison of Path-finding software in Python (a & c) and Java (b & d)

E. Computer Vision

Classic and machine learning-based approaches were considered when developing the SeaWolf VIII computer vision system. What determined which approach was implemented in a mission was the complexity of the images to be identified. A classic computer vision approach was implemented for tasks with simple polygon shapes (e.g. Path) and a machine learning approach was implemented for tasks with images with complex features (e.g. Buoys).

1) Classic Computer Vision: Edge and line detection was the foundational technique our classic computer vision was designed around. Edge and line detection are both image processing techniques that involve distinguishing outlines and line segments in an image. This design was referenced from software developed for RoboSub 2022 as this was a method that worked well. The algorithm was rewritten in Java for better integration into SeaWolf VIII's Java-based software architecture. The adapted model reduced images to 4 RGB colors by way of a combination of localized and global K-means to effectively segment said image with better consistency. Location and directional information were obtained using Principal Component Analysis (PCA) in tandem with color and size filters to determine each color's mean and covariance.

2) Machine Learning: For images with a complexity of features, neural networks were more suitable than classic methods as they enable the extraction of distinct features in complex images. We used YOLOv5 nano as our model and developed a Unity-ROS simulator to generate photo-realistic images of tasks to train the model. The simulator is capable of generating thousands of auto-labeled, synthetic images in different environments. Deployment of the machine learning model used Java OpenCV and ONNX formats. The entire

model is expected to pre-process the camera or model input and extract the object location, approximating distance using the output for robot decision-making.

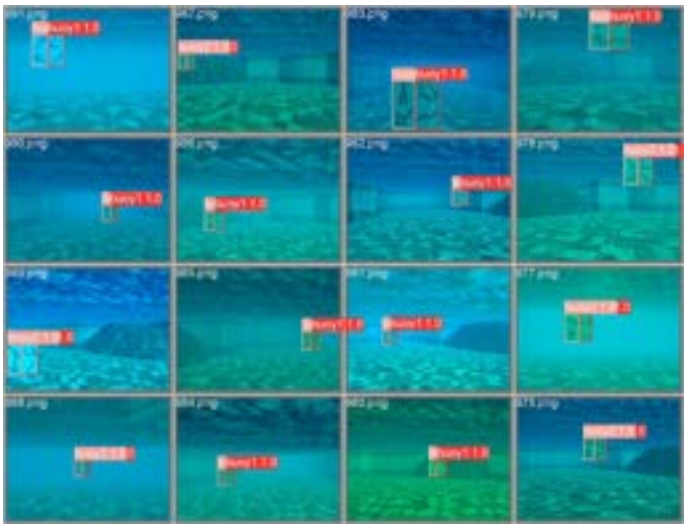


Fig. 4: Identification of Buoys in various simulated environments using machine learning

F. Manipulations Systems

1) Torpedo: An electro-mechanical torpedo launcher system was developed consisting of two identical launcher bodies each housing a watertight servo motor capable of actuating a lever arm holding a spring in compression until release. Several FDM printed projectiles were designed, with the chosen fusiform shape performing most favorably. The design was chosen due its simplicity which minimized the amount of electrical infrastructure needed and enabled iterative prototyping allowing for rapid development using an FDM printer while still meeting task requirements.

Fig. 5: SeaWolf VIII Torpedo Launcher

2) Dropper: The dropper mechanism for SeaWolf VIII consists of three major components: a 5V electromagnet, a magnetic 440C stainless steel ball, and a 3D-printed housing. The electromagnet is seated in the upper portion of the housing and is snugly held in place with a screwed-on lid. The stainless steel ball, or marker, rests in the lower part of the housing. Current continuously runs through the electromagnet to create a magnetic field, which holds the marker secure in the housing. When the time comes to drop the marker, the current is stopped and the marker is allowed to fall straight down out of the bottom of the housing. There are two of these droppers attached to the bottom of SeaWolf VIII.

Fig. 6: SeaWolf VIII Dropper

G. Acoustics

Acoustic navigation via the passive SONAR system required three central components: signal capture and pre-processing, digital signal processing, and signal source estimation. All three components are required to work in tandem for our acoustic system to produce reliable results for the robot to then use for navigation.

1) Signal Source Estimation: The basis of our passive SONAR scheme is signal source estimation using time difference of arrival for hyperbolic position location estimation.

One method to find timer difference between respective pairs of receivers is via a cross correlation of two captured signals[7]. Mathematically, if two signals are correlated, as is expected in this situation, a waveform will be produced that has a maxima in time, showing at what group delay the two signals are most correlated. This extends to the discrete time case, which allows us to do this digitally.

With time difference between two pairs recovered, a hyperbolic equation can be used to estimate the signal source[2]. Ultimately, this gives reasonable azimuth and elevation angles, which the robot can then use to navigate to the pinger.

2) Signal Capture and Pre-processing: The acoustic signal from the pingers occupies 25kHz-40kHz frequencies. This pure tone signal motivates a simple pre-processing system, which accounts for signal attenuation and white noise. Signal capture utilizes phantom powered hydrophones passing into buffer circuitry, isolating the hydrophones from the rest of the system, and biasing circuitry, which removes the need for a negative voltage rail by using a 0dB gain op amp circuit to change "zero" reference to half supply voltage. A 10.4dB gain non-inverting op-amp circuit is then used as a pre-amplifier to ensure a larger capture of the input signal which is fed to four cascaded Chebyshev band pass filters with peak gain of 0dB or small attenuation. Without significant pass band gain in the filtering stage, the pre-amplifier becomes necessary, as small attenuation in the pass band can compromise signal integrity of small signals captured at hydrophones. A digitally controlled amplifier, the LTC6910, is used as a linear post-amplifier. This stage amplifies the filtered analog waveform such that the peak-to-peak voltage occupies the entire 5V range of the analog to digital converters (ADC) which follow. Different

amplification is necessary because path loss is going to change with distances due to the inverse square law.

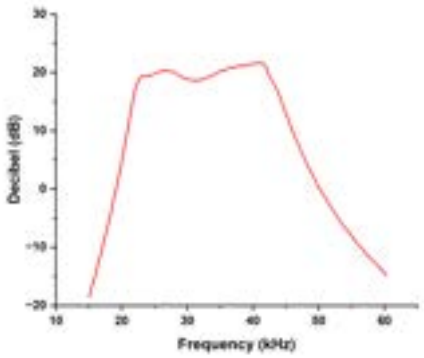


Fig. 7: Frequency Response of Analog Band Pass Filter for Passive SONAR System

3) Digital Signal Processing: Analog pre-processing gives us a reliable and less noisy waveform, but one that is difficult to gain insight on without digital processing. In the digital domain, the data is first sampled and digitized via a 5V, 500KSps, 10-bit ADC. This chip allows us to sample much greater than the nyquist frequency (80KSps for a 40 KHz signal) to avoid aliasing, and at 4.88 mV steps to give less significant quantization noise.

In the digital domain, all processing is offloaded to an Artix-7 FPGA. On board, we employ a basic linear detector of the fast Fourier transform (FFT) of our signal, with sole attention on our target frequency. For example, if searching for a 30KHz ping, the system performs the FFT on one of the incoming signal channels and observes the frequency at the 30KHz step. If the FFT at the target frequency exceeds a threshold, the signal is said to be detected.

Once the signal of interest is deemed present, the previously described cross correlation stage must be implemented to produce pair time differences. A direct implementation of cross correlation requires N^2 computations. Being computationally inefficient, it was determined that an equivalent approach would be using the Fourier Transform equivalent of the process. In discrete time, the cross correlation result can be obtained by taking the Fourier Transform of a signal, and a time reverse version of the other signal, pointwise multiplying the results, and taking the inverse Fourier Transform. This is more computationally efficient, and computationally faster. With the cross correlation obtained, a simple peak detector can be used to find the maxima, yielding the time difference of two channels.

H. Interchangeable Central Platform

1) Electrical Design: Taking a cue from version control systems like Git, the Interchangable Central Platform (ICP) is a system in SeaWolf VIII by which two main hull internals are maintained to allow for a stable and development platform. These two internal platforms, identified as Mother of All Boards (MOAB) and Father of All Boards (FOAB), maintain identical copies of the core electrical system supporting power, locomotion, and autonomy, but may differ at times in what auxiliary systems are present. As mentioned, this creates a situation whereby the electrical team may make continual improvements to the system while not impeding the testing of other teams who require a stable system, such as software. Furthermore, having two copies of the core electrical system on hand facilitates fault resilience in testing or competition situations. Should a non-inherent fault in the electrical system occur during testing, the other platform may simply be swapped into SeaWolf VIII and testing can continue while the fault is assessed.

Fig. 8: SeaWolf VIII Interchangeable Central Platform

The development of this system has been a focus of the electrical team this year, as we found in the past that modifications to the electrical system both impeded on and were impeded by testing of SeaWolf VIII. Previously, if major changes wanted to be made to the system either a large break in the testing schedule would need to be found or a condensed timeline would need to be implemented and thus these sorts of changes were often difficult to make.

2) Mechanical Design: The main features of the mechanical design of the ICP are the the super-structure used to hold the midplane within the hull and the electronics connections to the hull. The ICP super-structure is composed of three rings which suspend an acrylic midplane within Sea Wolf VIII's main hull. The midplane was fabricating using a CNC LASER system to enable cutting custom mounting holes for components directly into the midplane. Removable trusses mounted at the top and bottom of the ICP provide increased rigidity to resist moments which could dislodge components from the midplane. To improve the accessibility of placed components, each truss is individually removable.

The ICP features electrical connections to Fischer panel mount connectors at both ends of its hull. To enable this, the rear connectors are self aligning and connecting. As the ICP is pushed into its hull, these connections lock into place. The front connections are manual to preserve visibility of internal debug lights while testing on land.

III. Testing Strategy

A. General Testing Strategy

Our approach to testing our subsystems required simulation-based testing and in-field testing to validate our systems worked in real environments. Simulation-based testing was

implemented to ensure our design provided expected results and to pinpoint flaws. In-field or pool testing was implemented to validate all systems that were proven operate successfully via simulation. We aimed to have two pool tests per month in order to ensure sufficient testing of our custom control board and software architecture. See Appendix B.I: Test Schedule.

B. Locomotion Controls

The control board was developed in parallel to a new software system this year for SeaWolf VIII. Thus, to ensure initial locomotion testing could be done in parallel to mission code architecture development, a separate "development interface" to the control board was crated. These scripts were developed by the control board developers, not the software team.

On-system testing was done using the control board with the vehicle located in the water. The general strategy was to verify system modeling with base levels of motion, and gradually introduce more complex (more abstracted) forms of motion along with closed-loop control.

Additionally, some testing was done using the control board simulator (see Appendix B for details). This simulator models SeaWolf VIII using either a physical control board or a simulated control board. The use of a physical control board allows testing the real firmware in simulation. This was a critical part of the control board development process, allowing firmware bugs to be corrected before putting the vehicle in the water. The option to use a simulated control board allows testing (primarily by the software team) to occur without having any hardware. Thus locomotion portions mission code can be tested outside of the water, without access to a control board or to the vehicle itself. See Appendix B.II: Locomotion Testing and Validation.

C. Software and Computer Vision

System functionality was validated by automated tests through means of JUnit. A combination of simulation and in-field testing was utilized to evaluate the performance of our software and identify areas of concern. Simulations were used to both train and tune our computer vision algorithms. The simulation used was the Unity-ROS simulator mentioned in "Design Creativity". The ability of this simulator to generate thousands of photorealistic images in various environments in minutes enabled quicker turn-around rates for tuning our software. See Appendix B.III: Machine Learning Training and Results.

D. Manipulation

1) Torpedo: Once assembled, the torpedo system was fired repeatedly in and out of water to ensure independent actuation of each servo and characterize projectile motion in water. This was used to validate the design and identify an optimal projectile shape. See Appendix B.IV: Torpedo Projectile Test.

2) Dropper: The original dropper design was tested twice. Initially, the electromagnet was installed into a 3D-printed housing and hooked up to a DC power supply at 5V. The marker was placed in the housing against the electromagnet and, at the designated time, a current was applied to the electromagnet such that it repelled the marker. This test was both successful and repeatable when the dropper was out of the water. The same test was repeated with the dropper submerged in a bucket of water, but the marker failed to deploy correctly underwater. The dropper design was revised such that the electromagnet would remain on and produce a magnetic field until a designated time or location was reached, at which point the electromagnet would stop receiving current and the marker would drop. This version of the dropper was tested identically to the first iteration, both out of water and in the water. Both tests were successful, allowing a singular dropper to be attached to SeaWolf VIII for pool testing. Pool testing was a success, with the marker staying stable in the housing through a variety of underwater maneuvers and deploying consistently in the pool.

E. Acoustics

All direct testing of the Acoustics system has been through simulation, and mathematical verification of operation. Utilizing Python NumPy, we were able to generate sinusoidal waveforms at our sample frequency with insert discrete additive white Gaussian noise, which is similar to our conditions after sampling. In code, we were able to directly control our time differences, giving us a benchmark for performance of an FFT based solution. In testing, it was found that the direct cross correlation result and the FFT result were identical across a broad range of time differences. This confirmed to us that mathematically, this method was effective at efficiently producing time differences digitally.

Testing the digital backbone required simulation of the Verilog code utilizing in house created test benches, executed in the Vivado simulation environment. Testing attempted to sweep a wide range of input possibilities to give confidence that the system was robust, and could stand up to extremities and randomness that is otherwise unpredictable. Inherently tests are limited to giving insight on how the system reacts to specific circumstances, but a large volume of testing instills the confidence in the system necessary for deployment.

IV. Acknowledgements

The AquaPack Robotics is housed within North Carolina State University's Electrical and Computer Engineering department. We would like to thank the faculty and staff who have supported and continue to support the club. Special thanks to Dr. John Muth and Dr. John-Paul Ore for advising the club, as well as Casey Aquatic Center at Carmichael Gymnasium for providing us a facility for testing. Lastly, we would like to extend our gratitude to our sponsors for providing us financial support and access to their technical products for helping us design and develop the robot. Our 2022-2023 sponsors are Altium, Analog Devices Inc., Blue Origin, Intel, National Havoc Robot League, NC Space Grant, Microsoft Corporation, NCSU Engineer's Council, NCSU Student Government Association, NCSU Engineer Your Ex-

perience. We would also like to give a special thanks to our donor Carl Ryden.

V. References

[1] H. Anand Et al. "OpenUAV Cloud Testbed: a Collaborative Design Studio for Field Robotics". In: *IEEE Transactions on Automation Science and Engineering* (2021). DOI: 10.48550/arXiv.1910.00739.

[2] Elkamchouchi and Abd Elsalam Mofeed. "Direction-Of-Arrival Methods (DOA) and Time Difference of Arrival (TDOA) position Location Technique". In: *Twenty Second National Radio Science Conference*. 2005.

[3] Fjellstad and Fossen. "Quaternion feedback regulation of underwater vehicles". In: *1994 Proceedings of IEEE International Conference on Control and Applications*. 1994, 857–862 vol.2. DOI: 10.1109/CCA.1994.381209.

[4] Emil Fresk and George Nikolakopoulos. "Full Quaternion Based Attitude Control for a Quadrotor". In: 2013.

[5] Gernot Hoffman. *Application of Quaternions*. 2002. URL: http://docs-hoffmann.de/quater12012002.pdf.

[6] Byung-Uk Lee. "Unit Quaternion Representation of Rotation". PhD thesis. Stanford University, 1991.

[7] S. Lawrence Marpler Jr. "Estimating Group Delay and Phase Delay via Discrete-Time "Analytic" Cross-Correlation". In: *IEEE Transactions on Signal Processing* 47.9 (1999), pp. 2604–2607. DOI: 1053587X/99.

[8] R. Pio. "Euler Angle Transformations". In: *IEEE Transactions on Automatic Control* 11.4 (1966), pp. 707–715. DOI: 10.1109/TAC.1966.1098430.

[9] Leandra Vicci. "Quaternions and Rotations in 3-Space: The Algebra and its Geometric Interpretation". In: 2001.

Appendix A: Component Specifications

Component	Vendor	Model/Type	Specs	Qty.	Custom/Purchased	Cost	Year Acquired
Waterproofing							
Main Hull	OnlineMetals	8" Aluminum Tube	0.25 x 25.75 in	1	Purchased	$165.56	2022
Battery Hull	BlueRobotics	4" Watertight Enclosure	100 mm x 200 mm	2	Purchased	$638	2023
Main Hull Endcap	Mecha Inc	–	8" 6061 Aluminum	2	Purchased	$114.94	2022
Camera Enclosure	McMaster-Carr	–	–	2	Purchased & Customized	$33.2	2023
Waterproof Connector Plug	Fischer	S Series	–	22	Purchased	$750	2022
Waterproof Connector Receptacle	Fischer	DEU Series	–	22	Purchased	$750	2022
Electronic/Power System							
Load-Balancing Board (LBB): Ideal diode controllers	Digikey	LTC4359CMS8	150 µA/4V-80V	2	Purchased	$6.43	2021
Load-Balancing Board (LBB): MOSFET	Mouser Electronics	IXTN660N04T4 - I	40V/660A	2	Purchased	$32.17	2021
Main Electronics Board (MEB): Launchpad	Texas Instruments	MSP430G2553	1.8 V-3.6 V	1	Purchased		2022
LiPo Battery	Gens Ace	GEA10K4S10E5	15.2V/10000mAh/100C	2	Purchased	$154.99	2023
UBECs	SoloGood	–	5/3A Brushless Receiver Servo	3	Purchased	$12.99	2023
Manipulators							
Dropper	–	3D Printed	PETG	250g	Custom	$5.75	2023
Electromagnet	Adafruit	5V Electromagnet	5 Kg holding force	2	Purchased	$9.95	2023
Torpedo Launcher	–	3D Printed	PETG	450g	Custom	$11.50	2023
Servo Motor	Zoskay	High Torque Metal Gear Servo	25KG hold force, 6.8 V	2	Purchased	$18.36	2023
Mechanical Systems Board (MSB): Microcontroller	Texas Instruments	MSP430FR2355	16-bit/24MHz	1	Purchased	$12.99	2022
Controls							
Control Board: Microcontroller	Adafruit	ItsyBitsy	512 KB flash, 192 KB RAM32-bit Cortex M4 core	1	Purchased	$14.95	2022
Control Board: IMU	Adafruit	BNO055	9-DOF sensor, ARM Cortex-M0 based processor	8 1	Purchased	$34.95	2022
Thrusters	Blue Robotics	T200	Brushless DC motors	8	Purchased		2022
ESCs	Blue Robotics	Basic	7-26V/30A	8	Purchased	$36	2018
Acoustics							
Hydrophones	Aquarian Audio	H2C	10 Hz-100kHz Range	4	Purchased		2019
Power Distribution	–	Custom PCB Solution Rev. 2	4 Way Distribution, 1 A/Channel, 5 Vdd, 2.5 V AGND		Custom	≈$5	2022
Acoustics Front End	–	Acoustics Single Channel Rev. 1.2	500 KSps, 25kHz-40kHz BPF, 20dB-60dB passband amplification	1	Custom	≈$20	2022
Digital Signal Processing Unit	Digilent	Basys3	Xilinx Artix-7 FPGA, 90 DSP Slices, 1800 Kbits Block RAM	1	Purchased	$165	2022
Software Architecture							
Operating System	Qengineering	Ubuntu 20.04	–	–	Open-Source	$0	2022
Primary Language	OpenJDK	Java 11	–	–	Open-Source	$0	2022
Development Language	Python Foundation	Python 3	–	–	Open-Source	$0	2022
Serial Communication	Fazecast	jSerialComm 2	–	–	Open-Source	$0	2022
Automated Testing	junit-team	Junit 4.13.2	–	–	Open-Source	$0	2022
Deployment	int128	gradle-ssh-plugin 2	–	–	Open-Source	$0	2022
Build Tool	Gradle Inc	Gradle 8.1.1	–	–	Open-Source	$0	2022
Video Server	Gstreamer Team	Gstreamer 1.2.0	–	–	Open-Source	$0	2023
Video Processing	OpenCV	OpenCV 4.6.0	–	–	Open-Source	$0	2022
Vision							
Cameras	ArduCam	IMX219	4K 8MP	2	Purchased	$34.99	2023
OpenCV	Big Vision LLC	4.6.0	–	–	Open-Source	Open-Source	2023
YOLO	Darknet	v5	–	–	Open-Source	Open-Source	2023
Frame							
Perforated Aluminum Side 1	Custom KB Metalworks	–	8 x 10.48 x 0.250 in		Custom	Donated	2018
Perforated Aluminum Side 2	Custom KB Metalworks	–	8 x 10.48 x 0.250 in		Custom	Donated	2018
Hull Cradle	Custom KB Metalworks	–	8 x 10.48 x 0.250 in		Custom	Donated	2018
Main Hull Threaded Rod	McMaster-Carr	–	8 x 10.48 x 0.250 in		Purchased	9	2018
Interchangeable Central Platform							
Acrylic Backplane	McMaster-Carr	–	.25" Cast Acrylic	1	Purchased & Customized	$36	2023
Acrylic Truss	McMaster-Carr	–	.25" Cast Acrylic	4	Purchased & Customized	$14.40	2023
Rings	–	3D Printed	PETG	1250g	Custom	$28.74	2023

Appendix B: Test Plan and Results

I. Test Schedule

We performed seven major types of testing to debug and validate our systems. These tests included our navigation systems, manipulation systems, waterproofing measures, and full system tests. A legend of what our main protocols for simulation and testing is listed in Table I below.

TABLE I: Legend of System and Simulation Test

Test/Simulation	**Application**
Acoustics Simulation and Testing	Consists of testing how well acoustics circuits can detect and process pings in simulation and at pool tests. This helps determine if acoustics system can detect and process pings in an environment similar to the RoboSub competition.
Computer Vision Simulation	Consists of running mission and vision code through a simulated environment consisting of a pool, robot, and tasks at RoboSub. Each test consists of both running simulation and debugging. On many occasions issues found in code was debugged/improved outside of that time frame and simulated again.
Leak Tests	Consists of sealing the robot and vacuum testing its main hull and battery hulls. For each test, a pressure of -25 mm/Hg was held for a specific duration of time. Vacuum was held on the main hull for 40 minutes and each battery hull for 10 minutes. Leak tests were performed before each pool test. In the event that there appears to an issue with air leakage during a leak tests longer tests are held.
Locomotion Simulation	Consists of development and testing of simulation to validate math and orientation of custom control board. Used to identify issues with control board and correct them without in-water testing time.
Manipulation Systems Test	Consists on testing each mechanical system via its trigger to determine if system works.
Pool Tests	Consists of testing SeaWolf VIII in pool. Tests conducted at pool tests include locomotion, acoustics, computer vision, mission code, manipulation systems, and full system tests.
System Dry Run	Will occur before each pool test and after major changes to the electrical system have been made. Longer sessions would consist of doing checks on all of the electrical subsystems to ensure expected output was occurring. Thrusters are also run to ensure proper communication and determine if re-calibration is necessary. Typically 30 minutes each.

Table II presents the number of hours spent simulating and testing various systems and operations of SeaWolf VIII. This table includes planned hours of simulation and tests as well. Details of what each test entails is listed above in Table I.

TABLE II: Hours Spent Simulating and Testing Systems

	To-Date	**Planned**
Acoustics Simulation and Testing	20	20
Computer Vision Simulation and Testing	60	10
Leak Tests	24	3
Locomotion Simulation	50	2
Manipulation Systems Test	5.66	3
Pool Tests	68.75	21
System Dry Run	14	1.5
Total	242.41	60.5

Locomotion testing and validation was solely prioritized during the fall semester as completion of missions are not possible without proper orientation calculations and reliable communication between the control board, MEB (Main Electronics Board), and the computer. Locomotion validation was continued in the spring semester but testing of software architecture and communications was prioritized as well. Footage of props was collected for the purpose of testing simulated computer vision algorithms on real test cases. Summer pool tests prioritizes testing of missions and competition runs. Table III lists all completed and planned pool tests for the 2022-2023 academic year.

TABLE III: Completed and Planned Pool Test Dates

Semester	Fall	Spring	Summer
Dates	October 9th, 2022 October 15th, 2022 October 29th, 2022 November 12th, 2022	January 14th, 2023 January 29th, 2023 February 11th, 2023 March 31st, 2023 April 7th, 2023 April 9th, 2023 April 16th, 2023	June 10th, 2023 June 17th, 2023* July 8th, 2023* July 15th, 2023*
Hours Tested	23.75	38	28

Planned Pool Test Dates are denoted by ”*”.

II. Locomotion Testing and Validation

Proper testing of the control board is critical to mission success. However, this is a complex task requiring significant amounts of testing time during development. Due to limited in-water testing time and to ensure sufficient testing time for mission code, communications, and mechanical systems, a significant portion of control board testing occurred in simulation.

The simulator was developed using an open source 3D game engine, Godot. The selection of this tool for simulation was solely motivated by prior familiarity with this game engine. Godot includes a 3D rendering and physics engine, allowing simulator development time to focus on vehicle modeling and control board math validation.

The simulator not only models SeaWolf VIII, but simulates a control board as well. This allows testing and validation of mathematical methods in a high-level language where math libraries are already provided (by the game engine). Additionally, it allows mission code unit tests to run under simulation without access to any control board hardware or sensors.

Fig. 1: Control Board Simulator

However, a simulated control board is only capable of validating the approach to the problem, not the actual device. The largest risk with simulation testing is that the real control board's firmware has an implementation error. Even if the math is correct, it can be implemented improperly or other firmware bugs can prevent proper operation of the device. To address this, the simulator was expanded to allow use of a physical control board to control the simulated vehicle. In this operating mode, the simulator provides simulated sensor data to the control board, and receives motor speeds from the control board. Thus, the control board firmware itself can be tested and debugged under simulation.

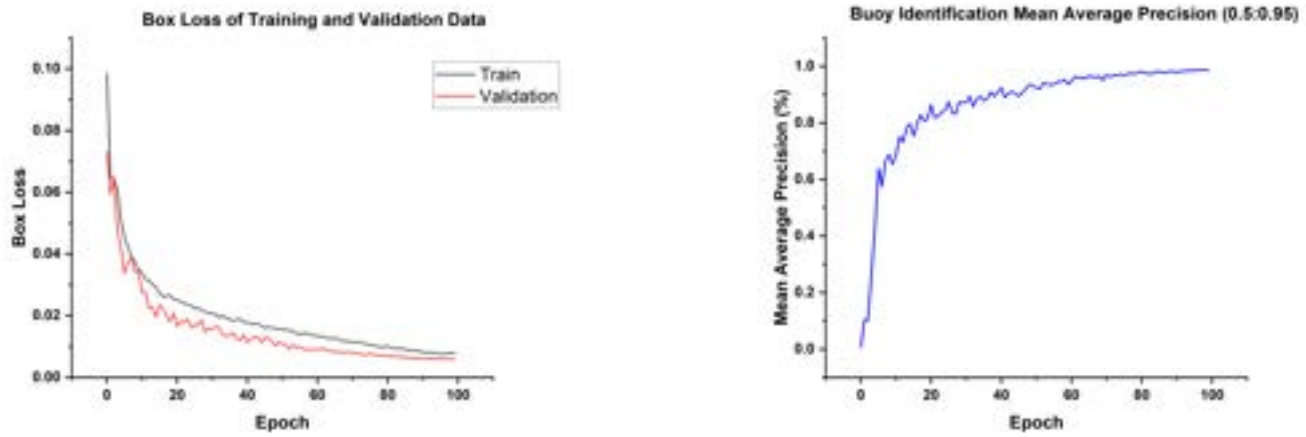
III. Machine Learning Training and Results

Training and validation of machine learning algorithms is vital to their performance in detection and identification of targets. This requires an efficient and reliable environment for us to train, test, and validate our algorithms. These needs require a simulator with capabilities to support simulation of robotic systems and a powerful game engine to generate photorealistic images and environments to increase the model accuracy. As such we chose a Unity-ROS simulator as it provides sufficient support for simulating robotic systems in virtual environments and is a game engine that can produce hyper-realistic simulations.

(a) (b)

Fig. 2: Simulation of Buoys in pool environment (a) and detection of Buoys in simulated pool environment (b)

The simulation consists of images ranging from blue to blue-green for simulating various water and visibility conditions. The generation of the environment also randomizes the position and rotation of the camera and pool which challenges the algorithm to become accustomed to undesirable conditions. The training/testing/validating split of datasets used is 80-10-10.



(a) (b)

Fig. 3: Box Loss of ML algorithm in training and validation data over an epoch of 100 (a) and mean average prediction of Buoys over epoch of 100 (b)

A considerable risk in developing machine learning models is the datasets generated oftentimes is biased to the simulated environments which results in negative performance of the algorithm in real-world conditions. The drop in performance can be associated with a simulations inability to simulate all conditions possible in the real-world which affects the neural networks ability process its input in real-world environments. To mitigate these issues, the simulator can be easily adjusted using a color slider to fit additional water conditions previously not possible. Additional assets can be added to the environment to increase variety of datasets, resulting in a more robust model. Finally, one of the best methods to mitigate a model's bias to simulated environments is incorporating real data in its training, testing, and validating datasets to improve the model's accuracy.

IV. Torpedo Projectile Test Results

Primary testing of torpedo projectile shape was through in-field testing in air and water. The test procedure included firing the assembled torpedo system and video-capture of the range of the torpedo projectile as measured by a 36 in ruler. A visual of a torpedo projectile test in water can be observed in Fig. 1.

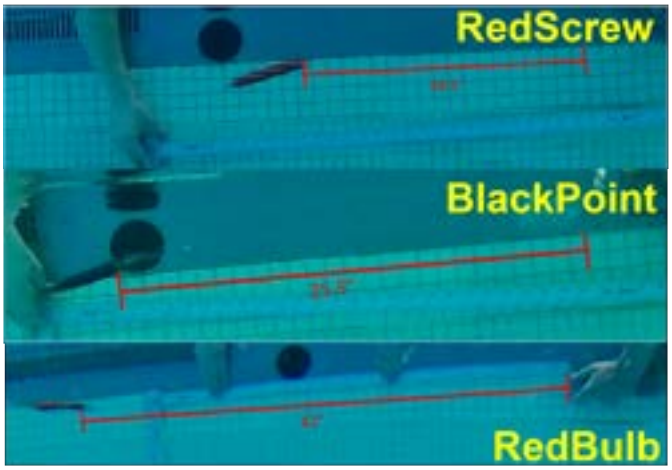


Fig. 4: Torpedo Projectile Testing

Table IV below summarizes torpedo projectile test results. Forward distance was defined as the distance the back end of the torpedo traveled away from the end of the launcher before sufficient momentum loss resulting in vertical deviation occurred. Side-to-side deviation is listed as 'minimal' if no qualitative deviation was observed.

TABLE IV: Torpedo Projectile Data

Projectile	**Avg. Forward Distance**	**Max Forward Distance**	**Side Deviation (first 30cm)**	**Side Deviation (Total)**	**Pass/Fail**
BlackPoint	23”	25.5”	Minimal	1-2”	Pass
RedScrew	17.5”	18.5”	Minimal	Minimal	Pass
RedBulb	36”	42”	Minimal	Minimal	Pass

All torpedo models met the desired minimum 12” (30cm) straight-line travel requirement with the RedBulb outperforming the other designs.