# Troy High School NJROTC RoboSub Technical Design Report

Anuj Patel, Rohan Patel, Sahana Anand, Timothy Elnitiarta,Phoenix Pitaknarongphorn, Taiyu Chen, and the rest of the Troy High School RoboSub Team

***Abstract*— The Troy High School NJROTC RoboSub team's Autonomous Underwater Vehicle (AUV), Neptune, was designed to compete in the 2021 RoboSub competition. Our first-year team of 11 high school students designed Neptune using off-the-shelf components and developed software using Python and Java. Designing Neptune allowed our team to learn how to use OpenCV, a proportional–integral–derivative (PID) controller, hydrophones, and power distribution boards (PDB). Our use of a single AUV was primarily driven by time and resource constraints and allowed our team to focus on Neptune's design. We used virtual collaboration tools including Zoom, GitHub, and LucidChart to work as a team while respecting COVID-19 guidelines.**

## I. Competition Strategy

The competition course this year, 23 (+1) Skidoo, is a continuation of last year's competition. This course consists of 5 components:

(i) Choose Your Side (Gate)
(ii) Make the Grade (Buoy)
(iii) Collecting (Bins)
(iv)Survive the Shootout (Torpedoes)
(v) Cash or Smash (Octagon)

As a new team competing for the first year, our overall approach to this competition was to tackle the most important components of the course first and then move onto tasks that were simple in order to maximize the number of points that we could score while still budgeting enough time to perfect each task that we focused on.

### *A. Number of AUVs*

As per rule 10.3.1 of the RoboSub 2021 Mission and Rules, each team is allowed to enter up to two vehicles. [1] While this option was considered by our team as it would allow for increased time in the pool and task specialization, we ultimately decided against it due to the increased cost and complexity. Our team determined that investing our time and resources into one submarine would not only allow us to reduce complexity in our setup but also allocate funds towards more advanced sensors that would enable us to complete tasks with greater precision. Once Neptune is fully operational, our team will reevaluate this decision.

### *B. Task Prioritization*

As a first-year team, we recognized that giving each task an equal amount of time would result in an AUV that would not have the capability to perform any tasks adequately. Therefore, during the planning process, we prioritized the tasks in the following order: Choose Your Side (Gate), Make the Grade (Buoy), Cash or Smash (Octagon), Collecting (Bins), Survive the Shootout (Torpedoes).

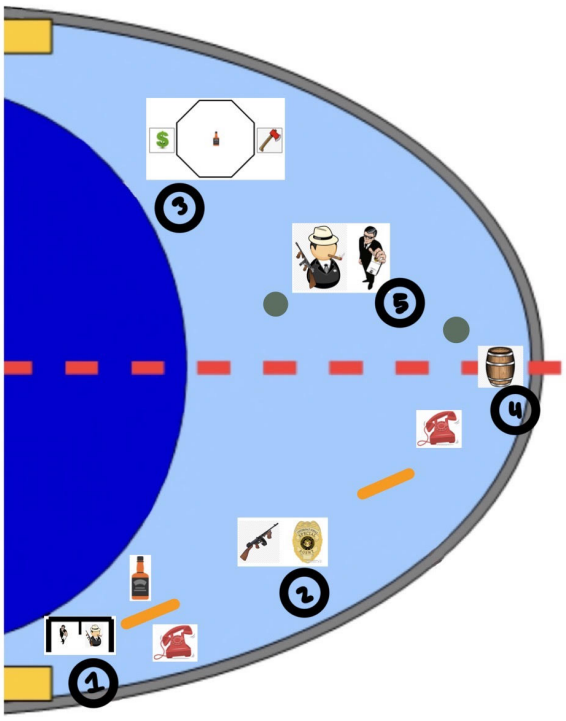


Fig. 1. The 2021 course diagram with task prioritization labeled

This order was selected based on requirements (navigating through the gate is a required task and thus was our top priority), point values, and ease. The prioritization can be seen in the submarine's design and software.

## II. Design Creativity

### *A. Vision*

Neptune currently uses 2 cameras and 2 sonars in addition to an onboard computer in order to detect various objects that are significant to the course. Object detection was the main focus of Neptune as it can be applied to all of the competition tasks and is essential to the navigation of the submarine. As such, our software sub-team dedicated most of its time to the development of the vision algorithm.

(i) Cameras

Neptune utilizes 2 mounted lowlight cameras with continuous video capture as its primary vision input. Both cameras have been placed on 180° tilt systems for full coverage of Neptune's environment. The front-facing camera has horizontal motion and the rear-facing camera has vertical motion. This setup was chosen for its cost-efficiency, reliability, and simplicity.

A previous setup included an extra non-adjustable camera mounted on the bottom of Neptune, however, it provided minimal additional coverage and strained the onboard computer so our team decided to remove it. Another previous setup involved mounting 2 cameras to the front of Neptune for high accuracy distance detection via a stereoscopic vision algorithm. While this setup did work well, we found that sonars were not only more reliable but also easier to apply to a wide variety of objects.

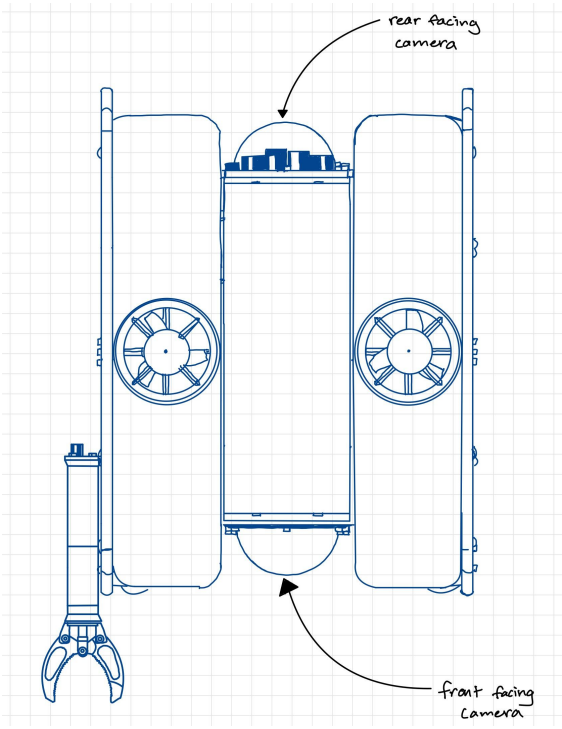


Fig. 2. Camera placement of Neptune

(ii) Sonars

Neptune utilizes 2 ping sonars for precise object location and obstacle avoidance. The sonars are positioned to the immediate left and right of the forward facing camera.

This setup was chosen as Neptune always reorients itself such that the front camera will be facing the object in focus. As such, forward-facing sonars will be able to face the same orientation as the camera. Two sonars were mounted so that Neptune could determine the real distance between itself and the object (See Figure 12) Our team decided to use BlueRobotics's Ping Sonar Altimeter and Echosounder as it is more cost-effective than other alternatives while still providing adequate reliability and functionality.

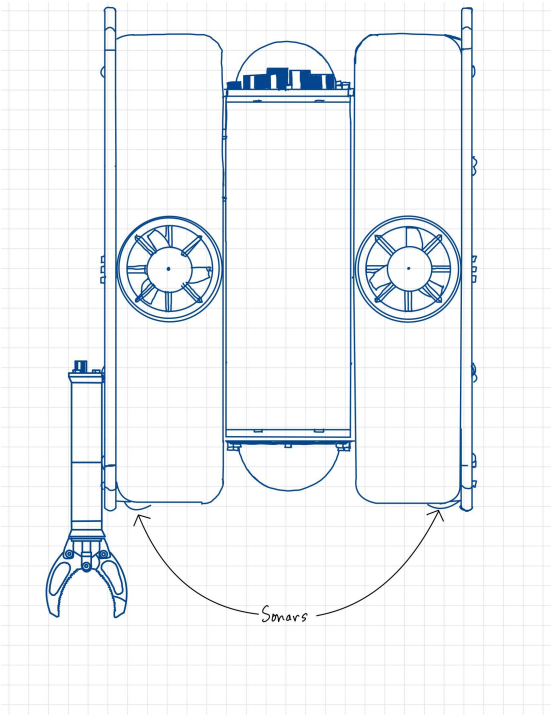


Fig. 3. Sonar placement of Neptune

(iii) Computer

Neptune's onboard computer is an NVIDIA Jetson Nano. It runs most of Neptune's software including the object detection algorithm. Our team chose to use an NVIDIA Jetson Nano over a Raspberry Pi due to more I/O ports and a better processor.

(iv) Software

a. Video Processing: Our video processing algorithm includes extracting frames of the video feed to process and color-correcting each frame. Neptune utilizes the BlueRobotics Low-Light HD USB Camera which is calibrated for underwater low-light conditions therefore, the lighting and color accuracy is adequate without any additional correction. However, while testing, we noticed that the video footage had a significant green tint that interfered with the object detection. To combat this, we developed a color-correction algorithm that implements Sea-Thru, an algorithm designed by Derya Akkaynak that recovers the original colors of objects from underwater images to allow for consistent grayscale object detection. [2]

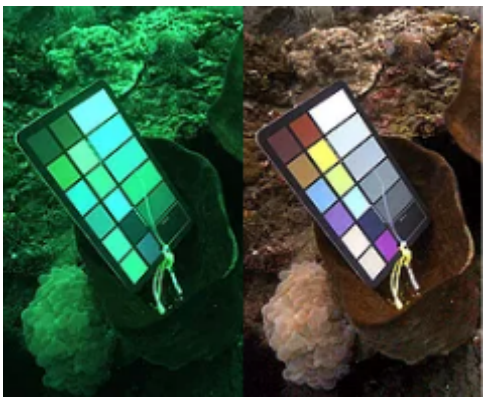

Fig. 4. Non-color corrected image on the left and image with Sea-Thru algorithm applied on the right

b. Blob Detection: After a frame is extracted from the video feed and

color-corrected, Neptune uses a blob detection algorithm to identify any objects in the pool. The blob detection algorithm identifies objects by recognizing when properties such as color or brightness are different from an object's immediate surroundings and is controlled by 4 parameters: thresholding, grouping, merging, and center & radius calculation. The blob detection algorithm outputs a grayscale image to make the object detection more precise. The blobs have properties including area, threshold, circularity, inertia, and convexity.

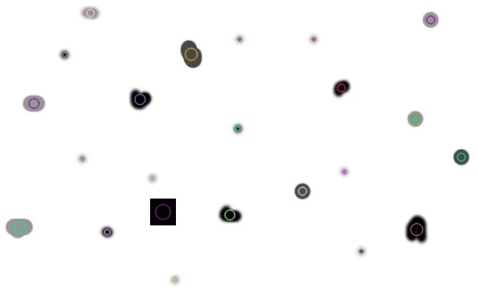

Fig. 5. Blob detection output

c. Object Detection: After the blob detection algorithm is run, Neptune uses the properties listed above (area, threshold, circularity, inertia, and convexity) to attempt to match the objects to course elements.

The steps taken by Neptune after an object is detected are split into 3 stages:

1. Initial detection - Neptune moves around the area of the pool it is in until it recognizes an object significant to the current task.
2. Repositioning - Once the submarine finds the object it is searching for, it repositions itself so that the object is in the frame of the forward-facing camera with no tilt (right in front of Neptune)
3. Distance detection - The forward-facing sonar is activated and calculates the distance between itself and the object in focus.

*B. Hydrophones*

Our team placed hydrophones on our AUV to help us detect the pings used to direct us to the next task. Pingers are present for the last two challenges to guide the AUV. We used two teledyne hydrophones which are omni-directional. This means sound coming from various directions will be detected and recorded with equal sensitivity. Our hydrophones measure the changes in pressure of the surrounding water when the ping is sent and convert the sound waves into electrical energy.

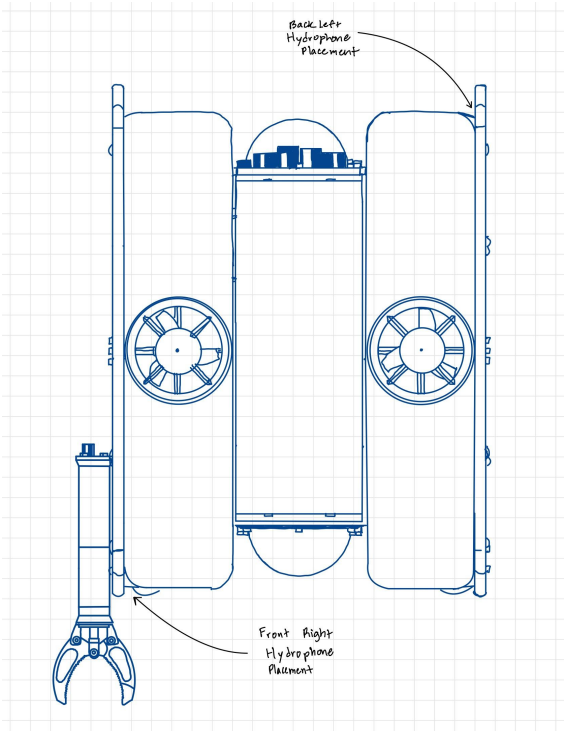


Fig. 6. Hydrophone placement of Neptune

After collecting this data, we use the MUSIC algorithm to identify the location

and angle of the pinger. [3] There are many advantages to using the MUSIC algorithm that make it better suited to detect and identify the location and the direction of arrival of the pings than other algorithms we might have considered. These advantages include its ability to measure numerous signals at the same time, its precision, and our larger understanding of this algorithm in comparison to other algorithms. The way this algorithm works is that from the data collected by the two teledyne hydrophones we have on our AUV, the correlation matrix is calculated, and the MUSIC spectrum calculations allow for the estimation of the largest peaks leading to the estimated angles.

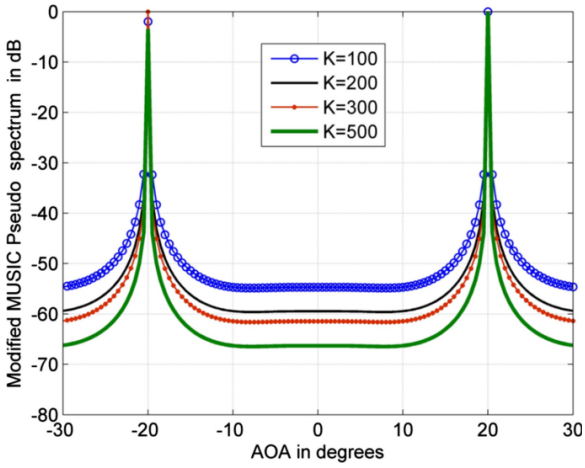


Fig. 7. MUSIC algorithm output

The hydrophones are placed at opposite ends of Neptune's frame, with one mounted to the front right, and the other to the back left of the submarine. A previous setup included placing both hydrophones near the front of the frame. This provided for easier coding, however it did not make full use of both hydrophones so we decided on our current placement. Having the hydrophones spaced out allows us to collect more data from a larger space in the pool, thus making Neptune more efficient in the tasks that utilize pingers.

*C. Overall Design*

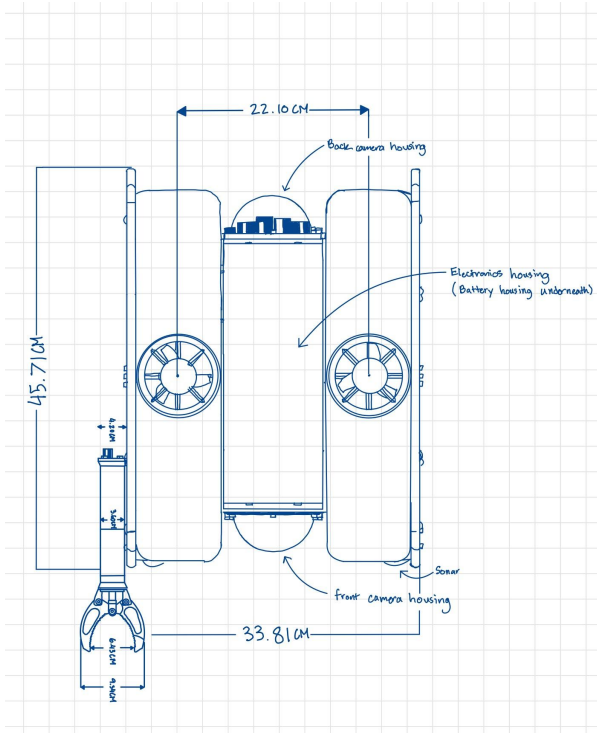


Fig. 8. Top view of Neptune

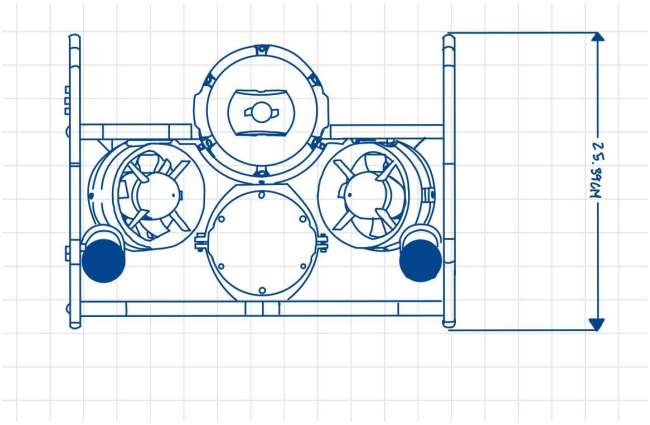


Fig. 9. Front View of Neptune

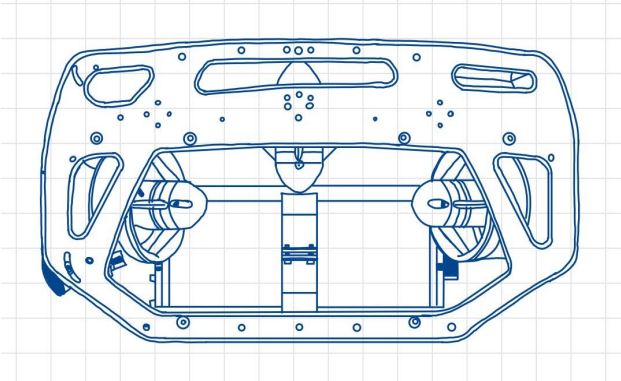

Fig. 10. Left-side view of Neptune

## III. Experimental Results

### *A. Cameras*

The first aspect of Neptune that needed to be tested was the vision algorithm. This was arguably the most integral aspect of Neptune if it were to succeed in the competition. We needed to test if it was able to correctly identify objects and images based on the database we provided it. Neptune's vision algorithm utilizes a framework that allows for us to switch between databases in a single line of code. Thus, we are able to use different databases instantaneously to identify numerous images and objects.

Once we have completed the vision algorithm, we began the testing phase. We initially manually inputted images into the algorithm. Some of these images include an underwater shot of a pool to test as a baseline, and home-made orange markers and other pictures similar to the one used for the competition to see if the vision algorithm could successfully identify it. Once the algorithm could correctly identify all of the images, the same process was repeated for the cameras on the AUV.

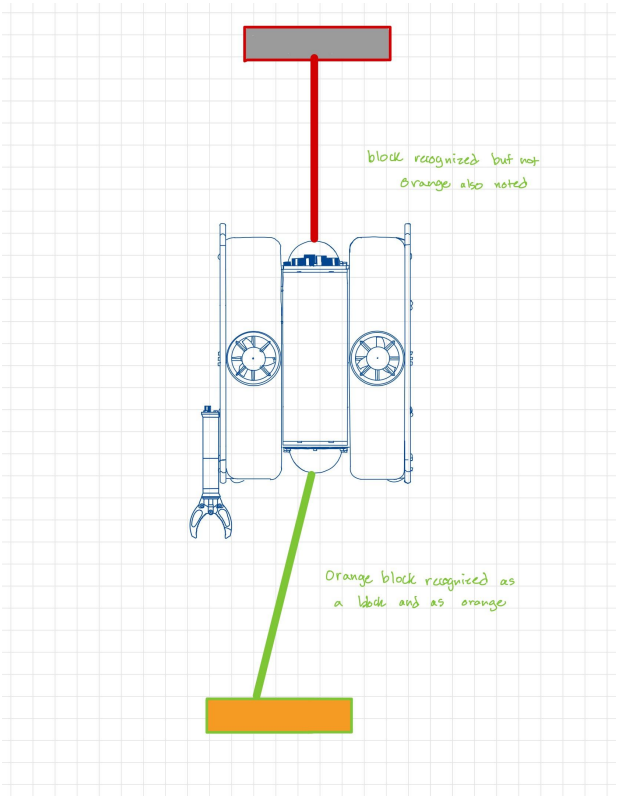


Fig. 11.Neptune's object detection algorithm depicted in a diagram

### *B. Sonars*

Furthermore, needed to test the accuracy of the sonars and how to utilize both of the forward sonars to map a front view of what Neptune should perceive in order to complete the challenge. We tested the sonars initially by submerging them underwater in a pool while moving a board of wood in front of it. The first issue we ran into was the fact that the frequency sent was being distorted by the water and did not provide a definitive reading. To fix this issue, we increased the frequency it operated at to 115 kHz and it started to provide us with proper readings. We used the data we received to develop a viewer for a singular sonar that showed how confident a reading was and how far it was away from the sonar.

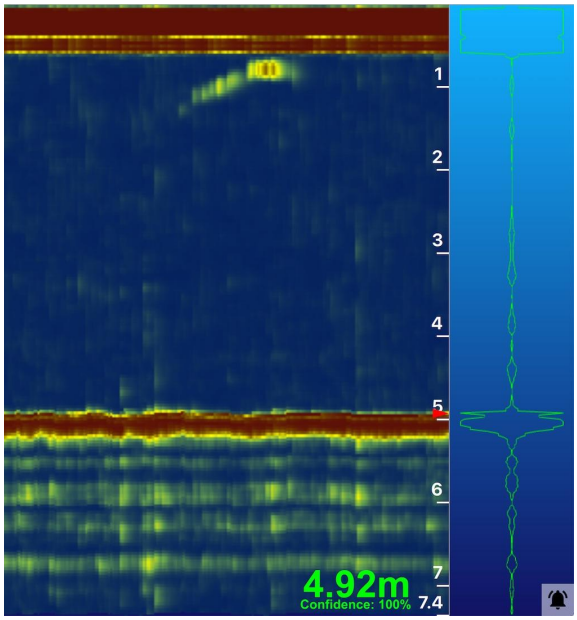


Fig. 12. Sonar readings while being held at a constant distance from surface

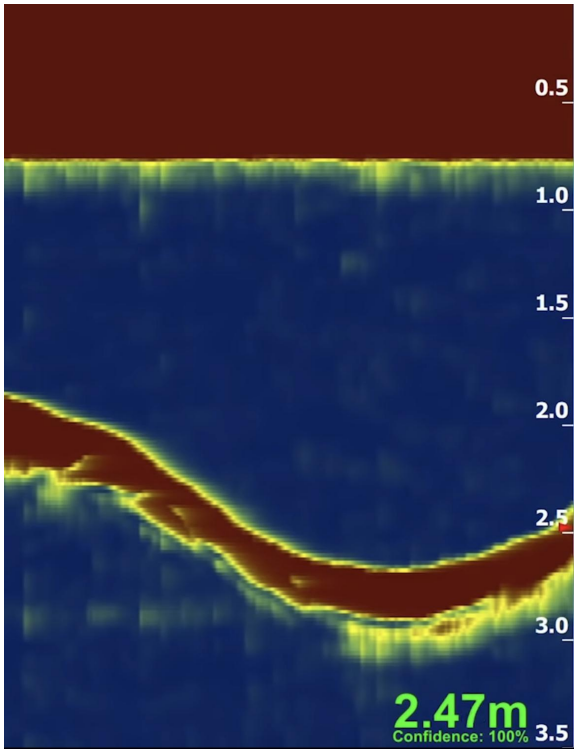


Fig. 13. Sonar readings while being held at variable distances from surface

However, when trying to use the input from both of the sonars to map the area in front to the AUV, there was a great level of inaccuracy as if an object was on the right half of Neptune, the left sonar would output a greater distance reading, with the same happening in the mirrored case. To solve this problem, we utilized the smaller measurement as long as the sonar was to the left of the left sonar and the right of the right sonar. (See diagram below) If it was in the middle of the two sonars, i.e. directly in front of the AUV, we used trigonometric functions to dictate the true distance the object is from Neptune.

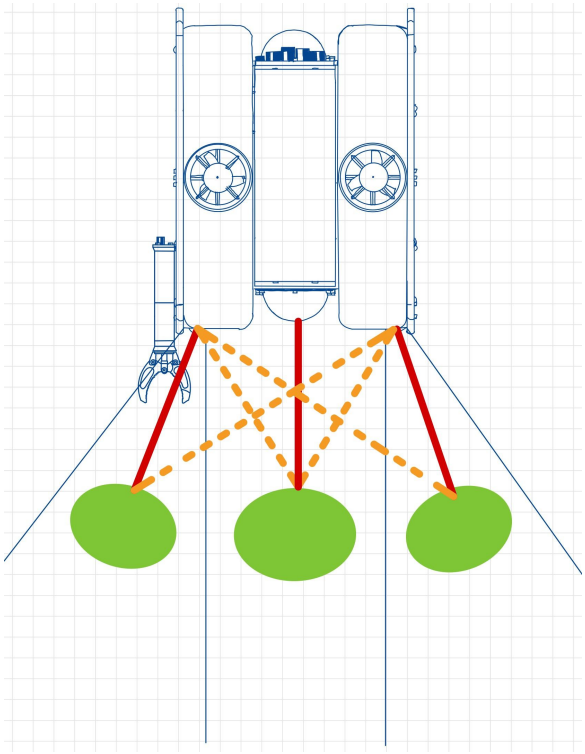

Fig. 14. Diagram of Neptune's use of two sonars

## IV. Acknowledgements

Our team could not have functioned without the support of our generous sponsors. We would like to thank the following organizations for sponsoring our team: Troy High School NJROTC Booster Club, Raytheon Technologies, Armed Forces Communications and Electronics Association, Navy League of the United States Inland Empire Council, and the Navy League of the United States STEM Institute.


Fig. 15. Troy High School NJROTC Booster Club


Fig. 16. Raytheon Technologies


Fig. 17. Armed Forces Communications and Electronics Association


Fig. 18. Navy League of the United States Inland Empire Council & Navy League of the United States STEM Institute.

## V. References

[1] “Resources.” *RoboSub*, robosub.org/resources/.

[2] Akkaynak, Derya. “Sea-Thru.” *Derya Akkaynak*, www.deryaakkaynak.com/sea-thru.

[3] Zhou, H. & Gu, X. & Jiang, X.. (2006). MUSIC estimation algorithm based on vector-hydrophone array. 30. 565-568.

# Appendix A: Component Specifications

| **Component** | **Vendor** | **Model/Type** | **Specs** | **Qty** | **Total Cost** |
|---|---|---|---|---|---|
| Frame | BlueRobotics | BlueROV Frame Only | - | 1 | $339 |
| Waterproof Housing | BlueRobotics | Watertight Enclosure for ROV/AUV (4″ Series) | - | 1 | $162.90 |
| Thrusters | BlueRobotics | T200 Thruster w/ ESC | - | 6 | $194 |
| Motor Control | Ardupilot | ArduSub | Software | - | $0.00 |
| Battery | Turnigy | Turnigy Graphene Panther 5000mAh 4S 75C | - | 1 | $81.74 |
| CPU | Nvidia | Jetson Nano | GPU and 4 GB of RAM | 1 | $98.95 |
| Vision | BlueRobotics | Low-Light HD USB Camera | - | 2 | $89.00 |
| Pinger Localization | BlueRobotics | Ping Sonar Altimeter and Echosounder | BLUART USB to TTL Serial and RS485 Adapter | 2 | $279.00 |
| Hydrophone | Teledyne Marine | Teledyne TC 4013 | - | 2 | $2,625.00 |
| Manipulator (Subsea Gripper) | BlueRobotics | Newton Subsea Gripper | - | 1 | $439.19 |
| Algorithms: vision | - | - | Sea-Thru, Blob Detection, Object Detection | - | - |
| Algorithms: localization and mapping | - | - | MUSIC algorithm | - | - |
| Open source software | - | - | OpenCV | - | - |
| Team size (number of people) | - | - | 12 persons | - | - |
| Expertise ratio (hardware vs. software) | - | - | 5 mech to 7 software | - | - |
| Testing time: simulation | - | - | - | - | - |
| Test time: in-water | - | - | 7 hours | - | - |
| Programming languages | - | - | Java, Python | - | - |

## Appendix B: Outreach Activities

### *A. Outreach*

In the 2020 - 2021 school year, SPEAR created a 26 week programming course for middle schoolers with varying coding experience. With over 60 students from multiple states, this program taught middle schoolers the basics of coding in Java and Python through lectures, projects, and quizzes. Concepts covered in this class included an introduction to Object Oriented Programing, syntax, iteration, recursion, booleans, and arrays. After becoming familiar with basic programming concepts and learning how to navigate IDEs, each student successfully completed a "final project" that demonstrated skills they learned throughout the year.

### *B. Partnership with Techtacular*

We partnered with a local organization, Techtacular, to further our reach in spreading computer science expertise to young scholars. Techtacular hosts free events around the local community teaching STEM-related topics impacting over 2400 rising technology stars. Along with Techtacular, we hosted events both virtually and in person. The in-person events took place late 2019 before the COVID pandemic took hold of the world. In these events we taught elementary school aged children how to create their own games using their own imagination. We fostered the process of taking an idea and making it into a reality. The students used resources such as Scratch and App Inventor to create their own apps/games. Additionally, we were able to give them an insight into robotics by having them code Spheros to solve a maze. In another one of our classes, we gave the students an introduction to 3D design and had them design something on their own that we could print for them. All of these kids come from different backgrounds and were able to learn these skills free of cost.

Fig. 19. Technology class hosted in partnership with Techtacular in late 2019.