RoboSub 2026 Technical Design Report

Amador Valley High School (AVBotz)

Aadi Dahake (President); Mechanical: Kedarnath Arunkumar (VP), Noah Bowen, Devesh Senthilraja, Rachel Schmidt, Pratham Mehta, Rian Shah, Shrihan Gogineni, Arush Gajula; Electrical: Frank Li (VP), Alex Vovenko, Allen Ding, Gaurav Gupta, Vedaant Sharma, Aditya Nalabotu, Edward Chen, Sumedh Venkata, Ryan Li, Arnav Garg, Kriday Ashok, Leo Zhang; Software: Kethan Vegunta (VP), Erick Liu, Aarush Tahiliani, Varun Gundamarju, Ankur Sharma, Aarav Juneja, Joshua Kim; Business: Nyle Farhan Ul Haq (VP), Ishan Ramakrishnan, Rishik Miriyala, Neal Goswami, Neil Verma, Arush Shah, Ruhan Datla, Badrinath Arunkumar, Tanvi Jani.

***Abstract*—For RoboSub 2026, AVBotz is proud to introduce Marlin V3, featuring significant improvements across its mechanical, electrical, and software subsystems. Building upon a foundation of tried-and-tested components from Marlin V2, our latest iteration integrates several new additions aimed at improving efficiency and reliability. Key hardware advancements include an upgraded grabber and intake, enhanced electrical connectivity, and optimized thruster communication. Furthermore, we have significantly streamlined our simulator, computer vision pipeline, and autonomy systems, culminating in our most accurate and capable vehicle to date.**

I. Competition Strategy

Our competition strategy for RoboSub 2026 involves focusing more heavily on earlier tasks (Gate, Slalom, and Bin), as these are essential to the overall run and will garner most points for the amount of time spent on them. Subsequent tasks (Torpedo and Octagon) will be primarily validated in the simulator, though physical testing will still be conducted to some level for these tasks.

A. Gate

Similar to last year, our approach to the Gate task involves training ML models to detect the correct opening, the distance to the gate, and the angle toward the gate. Using this information, our sub moves through the *Gate* while adjusting its angle and depth to ensure that it passes half a meter below the *Gate*. Immediately after passing through, the sub spins 720 degrees to earn style points.

B. Slalom

The *Slalom* task introduced in the 2025 competition called for different set of requirements from the prior buoy challenge. To improve our execution of this task from last year, we have focused on tuning and improving the control system of the AUV for improved precision. In response, we redeployed our machine learning–driven vision pipeline—originally trained for buoy detection—to identify and track vertical pipe pairs, enabling dynamic estimation of gate centerlines. Upon detection, the AUV initiates a controlled slalom behavior, executing lateral transitions while maintaining sub-crossbar depth to satisfy bonus scoring criteria. This maneuver is governed by a real-time feedback system integrating camera data, inertial measurement data, and depth sensing, ensuring continuous heading correction and fine-grained positional control.

Fig. 1: Marlin V3 Render.

C. Bin

Because there is a path marker from the buoy to the bins, we can reuse our vision and navigation software for this task, limiting complexity. We follow the direction of the path marker until we find the bins, scanning the floor while moving forward to save time. As there is no lid on the bin to pick up in this year's competition, we no longer need to spend time accurately aligning our grabber with the bin lid. Instead, we directly use our ML models to center our sub with the correct side of the bin, offset our sub so that our marker dropper is above the bin, and then drop the markers with our new one-hole dropper. Based on our simulation testing, our simple approach is both effective and time-efficient.

D. Torpedo

The torpedo task was heavily modified under new vision techniques, while the mechanical team's torpedo shooter remained relatively the same. Reusing our image processing techniques to calculate the orientation of the torpedo board so that we could align perpendicular to it, the software team found it difficult to precisely align to the two smallest octagon holes, because our ML models failed to differentiate between the smaller and the larger holes. As a result, we decided to isolate the red color of the holes, create a bounding box around each hole, and find the smallest bounding boxes. Through our simulation testing, this proved an effective way to fire our torpedoes through the smaller holes, maximizing our points.

E. Octagon

The octagon task was again the most difficult to complete. Mechanical first 3D-printed a grabber with four fingers but modified it to five fingers (a thumb-like contraption) after testing to grab the tube worm. On the software side, substantial simulation testing was required to detect the PVC pipes, orient with them, and pick each prop up. Another challenge software faced was dropping each prop in a different bin to maximize points, which the team is still currently working to address. With many more pool tests to come, we plan to continue testing this task.

II. DESIGN STRATEGY

This year's design effort focused on three areas: a mechanical refresh of our manipulators and electrical rack, a ground-up rebuild of our microcontroller and power-distribution stack, and a software-side upgrades in the simulator, perception, mission logic, and control pipelines. The following subsections detail each.

A. Mechanical Subsystem

1) Torpedo Launcher: The torpedo launcher system uses 3D-printed torpedoes coated in a hydrophobic layer, loaded into a custom 3D-printed enclosure. The mechanism consists of a printed chamber, locking lever, compression springs, and a servo-actuated release system. The torpedoes are secured in place by the lever while the springs remain compressed. When activated, the servo releases the lever, allowing the springs to propel the torpedoes forward. Integrated guide grooves within the enclosure align with the torpedo fins to maintain a straight and stable launch trajectory.

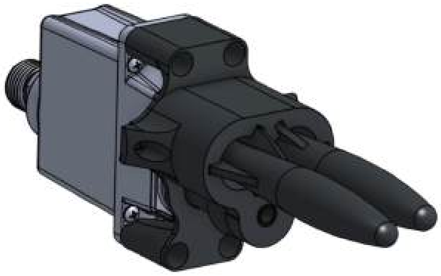

Fig. 2: Updated torpedo design with locking lever.

2) Marker Dropper: This year, several major design improvements were implemented to enhance the stability, reliability, and overall performance of the marker dropper system. A key upgrade was the addition of a servo hub and servo mounting plate, which redistribute the previously concentrated point load on the servo shaft across the entire hub assembly. By spreading the load more evenly, the system experiences reduced mechanical stress and improved structural support during operation. These changes help minimize wear on critical components, increasing the durability of the mechanism while also improving long-term accuracy, consistency, and reliability throughout repeated testing.

3) New Grabber: Throughout the year, improving the reliability of the grabber system when collecting a wide range of autonomous task props has been a major focus for the team. This season, development centered around an interlacing-finger grabber design aimed at increasing grip consistency and overall collection accuracy. Several 3D-printed variations of the finger geometry were designed, manufactured, and tested to refine the interaction between the grabber and the props. The interlacing configuration proved especially effective for this year's rounded props, providing a more stable and controlled grasp while maintaining a simple and efficient mechanical design.

4) New Rack: Following several issues encountered during last year's competition, the team identified major shortcomings in the design of the electrical rack, particularly the difficulty and time required to remove the rack for repairs and troubleshooting. To address these problems, this year's system was redesigned with a modular architecture to simplify maintenance and allow for faster component replacement and electrical access. In addition, linear slides were integrated into the mounting system, enabling the rack to be smoothly extended and removed without requiring extensive disassembly. These improvements significantly reduced maintenance time, increased accessibility to critical electronics, and improved the overall reliability and serviceability of the system during testing and competition environments.

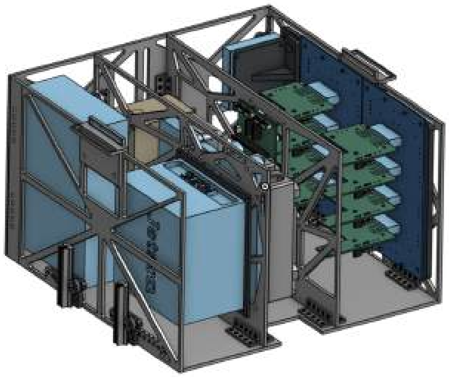

Fig. 3: Rack design: modular with slides, easily put together and taken apart.

B. Electrical Subsystem

1) Microcontroller Unit: This year, we switched our primary low-level controller from an Arduino Mega 2560 to a Raspberry Pi Pico alongside a CAN board. We developed a new custom hat that sits atop our Raspberry Pi Pico (MCU), which manages all of our submarine's communication protocols—CAN (Controller Area Network), UART (Universal Asynchronous Receiver-Transmitter), PWM (Pulse Width Modulation), and SPI (Serial Peripheral Interface)—to interface with the thrusters, servos, DVL, and more. The hat also integrates our kill switch circuit and our AHRS into a single PCB. The Raspberry Pi microcontroller allows us to integrate both PWM and CAN transmission better than the Arduino Mega allowed. It is also lightweight and easier to replace in case of unexpected failure in critical situations. Lastly, it is more cost-efficient and space-optimized, taking up far less room than our Arduino.

2) Sensor Stack: For our sensor stack this year, we made innovations to better integrate it within our new sub design.

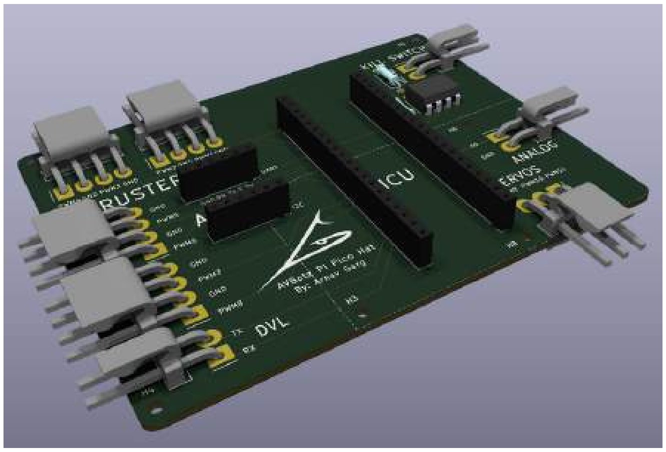

Fig. 4: Custom Raspberry Pi Pico baseboard integrating communication protocols.

Previously, it contained only temperature sensors, battery voltage monitors, and leak detection, all relayed to the STM Black Pill. However, in preparation for our new sub's face-seal design, we included an internal pressure sensor in the sensor stack to monitor the seal's integrity and pressure. This sensor informs us of our O-rings' condition, streamlining maintenance and troubleshooting when testing the sub underwater.

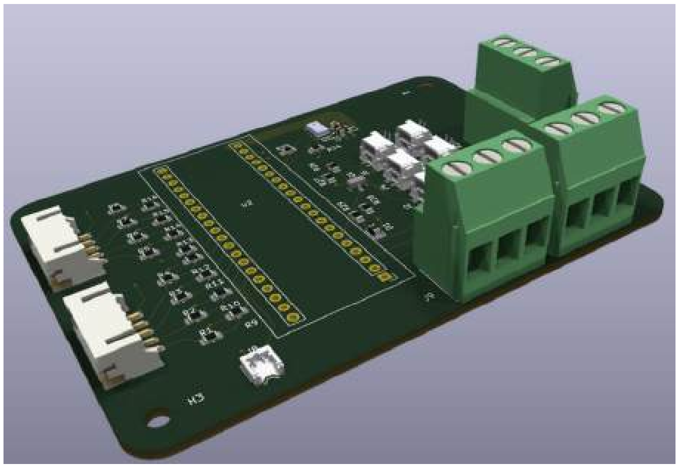

Fig. 5: Sensor stack built to monitor internal pressure, temperature, and voltage.

3) ESC Backplane: The ESC Backplane consists of two sets of four daughterboards mounted on a motherboard. Each daughterboard is attached to a Flipsky ESC, allowing for the use of either a PWM or CAN bus–based design. Four daughterboards are attached to the motherboard by a single hybrid power-signal connector, ensuring secure connectivity. Each sends 16 V to the ESC while relaying PWM signals, serving as a power distribution board between the battery and the thrusters. This year, we replaced our large ESC relay with several smaller relays, each mounted on the board in series with an individual ESC. These relays function as a hardware kill switch by disconnecting power to the ESCs during emergency stop conditions.

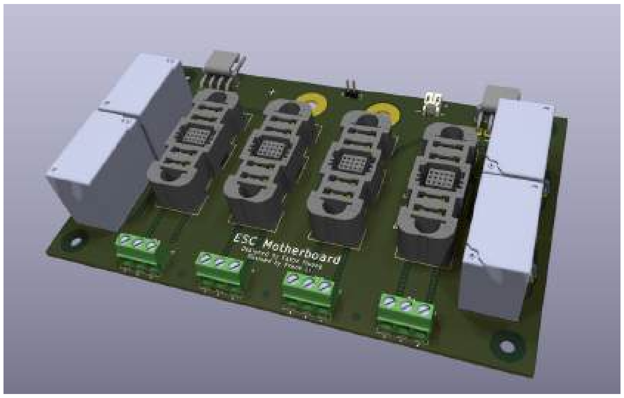

Fig. 6: Custom Flipsky ESC backplane motherboard integrated with both PWM and CAN bus support.

4) Jetson: This year, we transitioned to the NVIDIA Jetson AGX Orin, which significantly enhances our computational capabilities and provides the processing power required for more sophisticated algorithms and complex Computer Vision (CV) models. Additionally, thanks to the Jetson AGX Orin's 10 Gb Ethernet ports, we are able to incorporate a PoE Ethernet switch that connects all of our cameras and many of our sensors in one place. This setup streamlines connections by centralizing data from all cameras and sensors, enabling efficient information flow to the Jetson for faster and more reliable data processing—critical for real-time decision-making and improving the submarine's overall performance.

C. Software Subsystem

This past year, the software architecture was restructured to improve reliability, maintainability, and simulation fidelity. Key advancements include migrating the high-level system to the new *Thalassic* codebase (running Ubuntu 24.04), developing the *Pelagic* firmware for the STM32-based sensor stack, transitioning to the Stonefish simulation library, and improving the controls and perception algorithms.

1) New Simulation: To mitigate the impact of sporadic physical pool access, the team migrated the simulation environment from Gazebo Classic [1] to Stonefish [2], [3], a purpose-built maritime physics engine (Fig. 7). Stonefish provides high-fidelity modeling of hydrodynamic forces, such as buoyancy and drag, and generates realistic sensor data for the Doppler Velocity Log (DVL), IMU, cameras, and other sensors. Furthermore, an automated segmentation pipeline was developed using Stonefish's segmentation camera. By correlating per-pixel object identifiers with simulator metadata, the pipeline automatically generates pixel-accurate YOLOv11 annotations, allowing for synthetic data to be generated and used as part of the training corpus (Fig. 8).

2) Embedded Systems: Zephyr RTOS and micro-ROS: Upgrading from the Arduino ATmega2560 used in previous years, the embedded stack now utilizes a Raspberry Pi Pico and a WeAct Black Pill V3.0 (STM32). The increased computational power and memory of these microcontrollers enabled the adoption of the Zephyr Real-Time Operating System (RTOS) [4], improving hardware abstraction and code portability. Addi-

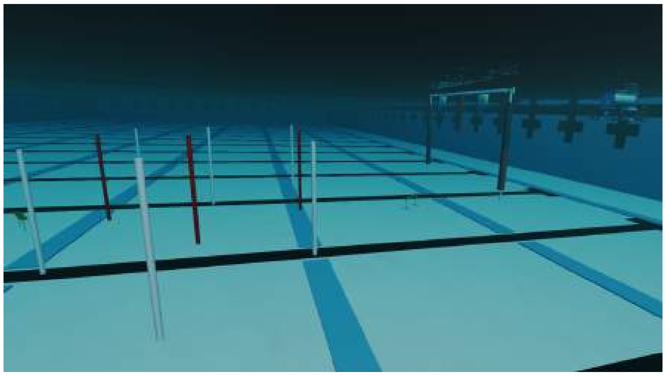

Fig. 7: Stonefish simulation pool environment with competition elements (Pathmarker, Gate, Marlin V3).

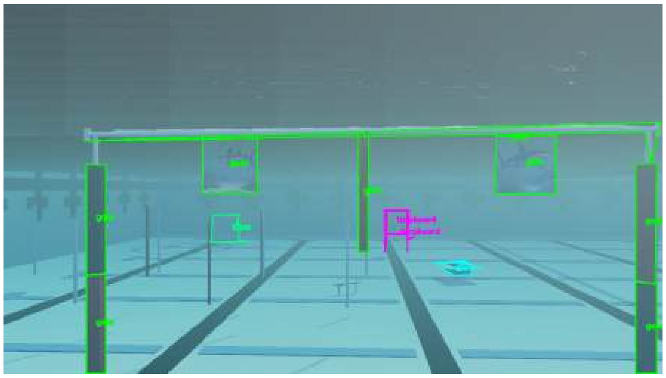

Fig. 8: Automatic segmentation and generation of synthetic data in simulator.

tionally, this performance overhead allowed the team to replace the previously used serial communication protocol with micro-ROS, ensuring more robust communication between the high-level compute and the low-level embedded stack.

3) Improved Computer Vision Pipeline: The perception pipeline integrates DeepSeeColor [5], YOLOv11 [6], and the Scale-Invariant Feature Transform (SIFT) [7] to ensure robust target identification.

As a preprocessing step, DeepSeeColor, a model-based underwater color correction algorithm, mitigates water-column attenuation and backscatter, significantly improving the reliability of downstream vision tasks (Figs. 9, 10). YOLOv11, trained on a combination of synthetic and real data, is then utilized to generate regions of interest (ROI) bounding boxes around mission elements (Fig. 11). Finally, SIFT is applied within these bounding boxes to extract robust, scale- and rotation-invariant keypoints, enabling the precise spatial localization required for complex tasks such as the torpedo board.

4) Upgraded Control Algorithm: The vehicle's control algorithm utilizes a cascaded Proportional-Integral-Derivative (PID) algorithm integrated with a quadratic programming (QP) thrust allocator [8] [9]. Transitioning from a static allocation matrix to a dynamic QP solver optimized power distribution and introduced fault tolerance against thruster degradation. While alternative control methodologies such as Model Predictive Path Integral (MPPI) and Linear Quadratic Regulator (LQR) were considered, the benefits they afforded were deemed not meaningful enough to justify the additional processing power and developmental complexity required to implement them.

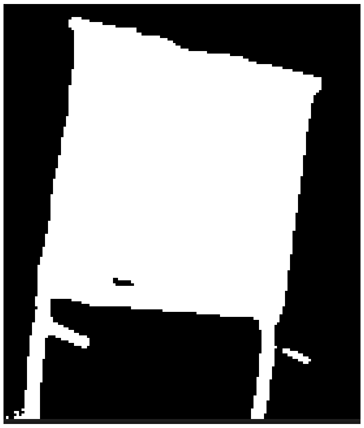

Fig. 9: Identifying the torpedo board with DeepSeeColor.

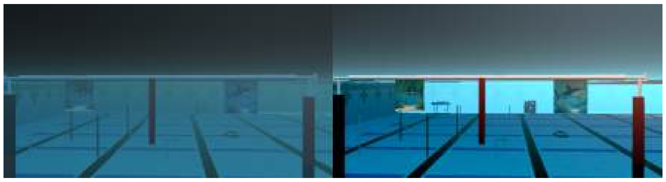

Fig. 10: DeepSeeColor used to remove underwater attenuation and color shift.

5) Mission Control with BehaviorTrees.CPP: This year we migrated from custom C++ logic for managing our mission logic and control to using BehaviorTree.CPP [10] in order to have a more scalable codebase with modularized behaviors, enhanced debugging with recordable state transitions, and increased developmental efficiency utilizing the GUI. GROOT 2, the UI framework used to visualize the behavior tree, let's us see the outcome of every node at a glance. This replaces tedious logging with immediate visual feedback, enabling us to develop much more rapidly. Another major benefit has been the clearer separation of concerns between mission logic and connectors. By strictly separating these two concerns, complex tasks that previously dealt with thousand line files are now either a quick GUI adjustment or a focused changed in a specific node which automatically propogates to all tasks across the entire codebase.

6) Telemetry and Visualization via Foxglove: Foxglove Studio [11] serves as the primary ground control station and telemetry visualizer for the software stack. The software team uses the platform to control the vehicle, monitor sensor data, and ensure proper operation. Furthermore, recorded ROS bags are visualized using Foxglove's playback capabilities for analysis of pool tests.

III. TESTING STRATEGY

Our testing strategy is divided across the three subteams, each adopting an approach tailored to its development cycle. Mechanical testing emphasizes iterative prototyping; electrical testing emphasizes modular validation of individual PCBs be-

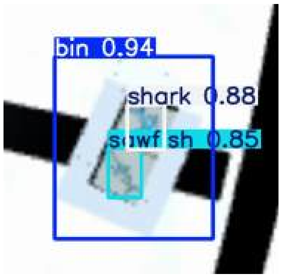

Fig. 11: YOLOv11 bounding boxes on *Bins* task.

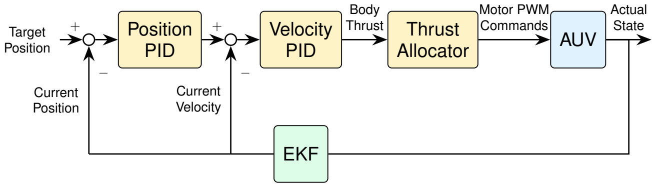

Fig. 12: Control system block diagram.

fore full integration; and software testing leverages simulation as the primary regression environment, with pool tests reserved for final validation. Appendices C through F provide detailed test plans and observations.

A. Mechanical

For our goals this year, the mechanical team worked to reduce, upgrade, and innovate on the current submarine, focusing on developing more task-based components and upgrading multiple existing components. Our strategy was to break down each task into smaller steps. The first step was design and research, where we researched, brainstormed, and designed multiple solutions. The second was prototyping, and the third—often the most challenging—was testing those designs and iterating as necessary. The team knew it was crucial to allocate as much time and precision as possible during the testing stage to explore every possibility and maximize the probability of success. The mechanical team followed this procedure when developing the grabber, torpedo shooter, and ball dropper.

B. Electrical

This year, our electrical team aimed to create an efficiently repairable design in order to maximize pool testing time. To do so, the team focused on improving the modularity of our design. Using the open-source KiCad electronic design software, the team created a series of custom PCBs to streamline different levels of our sensor and communication systems. By using several modular PCBs as the infrastructure for our entire electrical rack, we were able to quickly and efficiently test individual parts of the design. In particular, our ESC backplane allows us to quickly identify failures and replace parts efficiently, streamlining our testing process. Additional Electrical testing procedures can be found in Appendix D and E.

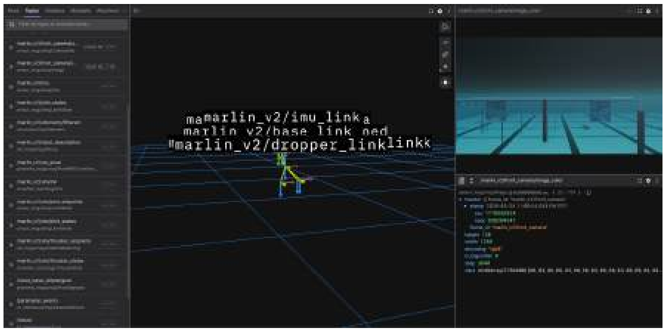

Fig. 13: Foxglove studio interfacing with simulation.

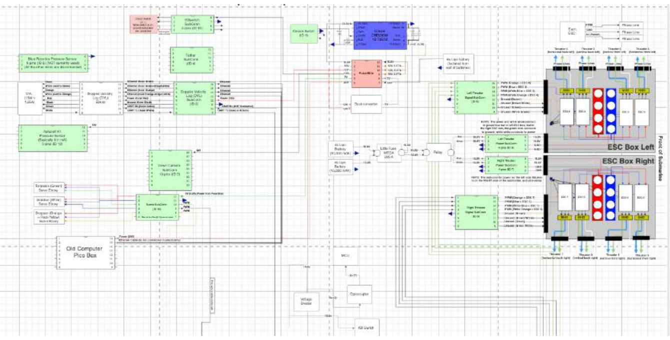

Fig. 14: Power and signal distribution chart.

C. Software

Simulation testing (fig. 15) has been the most effective method to validate software improvements. The software team models all competition props in Blender and integrates them into the Stonefish simulation environment, enabling comprehensive testing of all mission code, particularly the perception system. By verifying core logic in simulation beforehand, we are able to utilize our physical pool tests far more effectively, debugging integration issues and testing the vision system.

Fig. 15: Simulating the *Gate* Task.

D. Lessons Learned

Our most significant takeaway is that communication, across subdivisions and between leadership and general members, is essential to ensure goals and project timelines are met. Furthermore, we recognize that despite rigorous preparation, unforeseen technical challenges are inevitable. When faced with these setbacks, we strive to maintain composure and to methodically troubleshoot the issue.

ACKNOWLEDGMENTS

None of our work this year would be possible without a community of support. We thank our advisor, Bree Barnett Dreyfuss, for advising us on technical and organizational issues. We would also like to thank the Pleasanton Partnerships in Education Foundation, The Rotary Club of Pleasanton, Costco, and the Parent Teacher Student Association for their financial support; LDO Motors for generously sponsoring a Voron 3D printer; Rohde & Schwarz for generously providing an RTH1002 oscilloscope; PNI Sensor Corporation for kindly providing a user-friendly, magnetically self-calibrating Nav-iGuider AHRS; Amador Valley High School for providing us with pool facilities; and our parents for their endless patience and emotional support throughout this season. A full list of sponsors is provided in Appendix A.

REFERENCES

[1] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3. IEEE, 2004, pp. 2149–2154.

[2] P. Cieślak, “Stonefish: An advanced open-source simulation tool designed for marine robotics, with a ros interface,” in *OCEANS 2019 - Marseille*, jun 2019.

[3] M. Grimaldi, P. Cieslak, E. Ochoa, V. Bharti, H. Rajani, I. Carlucho, M. Koskinopoulou, Y. R. Petillot, and N. Gracias, “Stonefish: Supporting machine learning research in marine robotics,” in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, may 2025. [Online]. Available: https://arxiv.org/abs/2502.11887

[4] Linux Foundation, “Zephyr RTOS project,” 2026, accessed: June 10, 2026. [Online]. Available: https://www.zephyrproject.org/

[5] S. Jamieson, J. P. How, and Y. Girdhar, “DeepSeeColor: Realtime adaptive color correction for autonomous underwater vehicles via deep learning methods,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. London, United Kingdom: IEEE, 2023, pp. 3095–3101.

[6] G. Jocher and J. Qiu, “Ultralytics yolo11,” 2024. [Online]. Available: https://github.com/ultralytics/ultralytics

[7] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision (IJCV)*, vol. 60, no. 2, pp. 91–110, 2004.

[8] M. Zhou, E. C. Gezer, *et al.*, “Towards modular and accessible AUV systems,” *arXiv preprint arXiv:2309.16000*.

[9] URI Ocean Robotics, “ROS-MVP Vehicle Controller (mvp_control),” https://github.com/uri-ocean-robotics/mvp_control, 2024, accessed: June 10, 2026.

[10] D. Faconti, “Behaviortree.cpp,” 2024.

[11] Foxglove Technologies Inc., “Foxglove: Visualization and debugging for robotics,” 2021, accessed: June 10, 2026. [Online]. Available: https://foxglove.dev/

APPENDIX A
SPONSORS

We are incredibly grateful to all of our sponsors and supporters for helping make our team and our projects possible. Your generosity, encouragement, and belief in our mission allow us to continue designing, building, and testing our autonomous underwater vehicle while competing against collegiate teams from around the world at RoboSub 2026.

From funding materials and equipment to supporting travel, outreach, and innovation, every contribution has had a meaningful impact on our team. None of this would be possible without your support, and we sincerely thank you for investing in the next generation of engineers, innovators, and problem solvers.

A. Platinum Sponsors

Tanius

Datron

Water Linked

David Haubert

KLA

B. Gold Sponsors

LDO

Sim Scale

C. Silver Sponsors

Rotary Club

D. Bronze Sponsors

PPIE

Costco

APPENDIX B
ELECTRICAL DESIGN AND VALIDATION

To validate our power distribution under load, we performed sustained load-tests on each subsystem. Key metrics captured were current draw, voltage drop at the load, and steady-state temperature of the power components. Table I shows representative results.

TABLE I: Load Test Results.

Component	**Current**	**Voltage**	**Temp.**
Subsystem A	9.1 A	12 V	46 °C
Subsystem B	7.5 A	12 V	39 °C

Based on these load tests, we concluded that the power channels reliably support the expected step and sustained load profiles of our compute and camera components, giving us confidence in the rack's ability to safely power high-value components during competition.

APPENDIX C
ELECTRICAL INTEGRATION TEST PLAN

Once individual PCBs passed bench validation, full-stack integration was performed against the checklist in Table II. This checklist ensures every interface—power, thrusters, sensor I/O, and actuation control—is verified before in-water deployment.

TABLE II: Integration phase testing feature checklist.

Category	**Checklist Items**
Power	• Able to output stable 12V and 7V rails • Able to load balance across batteries with Voltage difference under 4V • Able to provide stable power to all computing systems eg. Jetson Orin • Able to run all 8 thrusters and all 3 servo actuators simultaneously • Able to run all devices excluding thrusters with one 4s battery
ESCs and Thrusters	• Kill switch is able to kill/unkill thrusters in under 1 second • All thrusters remain operational after cycling the kill switch • Kill switch is unable to be turned in before the AUV is alive (cincon switch is on) • ESCs are able to initiate properly on startup with 5 beeps
Sensors	• Raspberry Pi is able to monitor external pressure and heading data • Stm32 blackpill mounted on sensor stack is able to monitor temperature, internal pressure, and battery voltage • Jetson Orin is able to recieve DVL velocity/heading data as well as stream the camera feed through a tether

APPENDIX D
MECHANICAL INTEGRATION TEST PLAN

After fabrication and subsystem assembly, the mechanical systems underwent integration testing according to the checklist in Table III. These procedures verify structural integrity, watertightness, actuator reliability, and subsystem alignment before full in-water operation.

TABLE III: Mechanical integration testing feature checklist.

Category	**Checklist Items**
Hull and Sealing	• Main electronics enclosure remains watertight during vacuum and pressure testing • All O-rings properly compress without visible deformation • Bulkhead penetrations remain sealed under extended testing • Internal pressure remains stable
Frame and Mounting	• All mounted subsystems remain secured under vibration testing • Thruster mounts maintain alignment under full thrust operation • Center of mass remains within targeted stability margins
Actuation Systems	• Marker dropper actuates consistently across repeated cycles • Grabber is able to fully open and close under load • Torpedo launcher reliably deploys projectiles without jamming • Servo linkages return to default position after operation
Pool Readiness	• Vehicle achieves near-neutral buoyancy in water • Vehicle maintains stable orientation without excessive roll or pitch

APPENDIX E
SOFTWARE INTEGRATION TEST PLAN

To ensure reliability and autonomy before physical pool testing, the software stack undergoes rigorous validation using the Stonefish simulator environment. The software subsystems are evaluated against the checklist in Table IV to verify perception accuracy, control stability, and mission logic execution.

TABLE IV: Software integration testing feature checklist.

Category	**Checklist Items**
Perception & Vision	• *DeepSeeColor* effectively neutralizes simulated/real water-column attenuation and backscatter. • *YOLOv11* accurately generates Regions of Interest (ROI) for all mission elements (Gate, Bins, Torpedo, etc.) with $\geq$85% confidence. • *OpenCV* post-processing code correctly generates orientation/other information of all mission elements.
Control & State Estimation	• Extended Kalman Filter (EKF) maintains accurate state estimation by correctly fusing DVL, AHRS, and depth sensor data. • Control Algorithm reliably achieves target position and velocity without excessive overshoot or oscillation.
Mission Logic	• *BehaviorTree.CPP* correctly navigates fallback states and transitions when mission elements are lost from view. • Submarine autonomously executes Gate, Slalom, and Bin sequences consecutively within Stonefish without requiring manual override.
Embedded & Comms	• *micro-ROS* maintains stable, drop-free communication between the high-level Jetson AGX Orin and low-level microcontrollers (Pico/STM32).
Telemetry	• *Foxglove Studio* successfully receives live camera feeds, 3D poses, and diagnostic logs with minimal latency.

APPENDIX F
POOL TESTS AND STRATEGY ADJUSTMENTS

Our in-water testing is conducted in the Amador Valley High School pool. Before each test, the team aligns on testing goals for the day, performs standard hardware checks, and logs ambient conditions (lighting, water temperature) for later reproducibility.

Fig. 16: Pool testing environment.

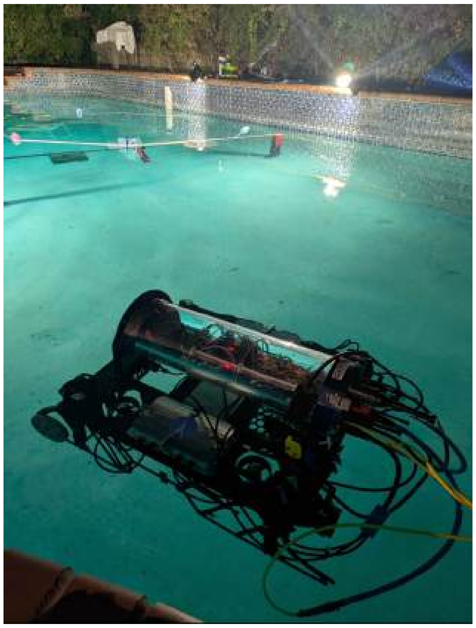

Fig. 17: Pool testing environment.

Beyond validating individual components, pool tests have repeatedly revealed blind spots in our development plan that only emerge in full in-water deployment. Findings from this year's testing campaign are being compiled and will inform our pre-competition adjustments.

APPENDIX G
OUTREACH ACTIVITIES

AVBotz, for the past 27 years, has upheld the core tenet of encouraging and facilitating the innovation and growth of skills in students. We make an emphasis on giving opportunities by offering new experiences through our volunteering and community service.

A. ACE Coding/ACE Code Day

AVBotz is grateful for the opportunity to give back to the community through ACE Coding and ACE Code Day. We promote coding and engineering at younger ages by leading weekly after-school lessons at Harvest Park, Hart, and Pleasanton Middle School in partnership with ACE Coding. Throughout each school year, our members teach Java and Python to more than 100 middle school students through a 20-week curriculum.

Fig. 18: AVBotz Software members teaching students during ACE Code Day

In addition to weekly lessons, AVBotz members also lead workshops at ACE Code Day, a six-hour event where students are introduced to topics such as CAD using OnShape, electrical engineering, and computer vision with OpenCV. These events give younger students hands-on exposure to engineering concepts and encourage them to explore STEM pathways.

The impact of ACE Coding and ACE Code Day has been extremely visible, with many of our own members first becoming interested in coding and engineering through these outreach opportunities. AVBotz is proud to help inspire the next generation of innovators within our community.

B. Donlon Elementary Engineering Night

Donlon Elementary's annual Engineering Night has been an incredible opportunity to inspire the next generation. AVBotz has attended Engineering Night for the past 4 years and has showcased Marlin V3. We make sure to emphasize the spirit of innovation and future growth. We believe that early exposure to robotics will be instrumental in furthering opportunities for children in the community and Donlon Elementary Engineering Night acts as the bridge for young students to grow their skills and become innovators themselves.

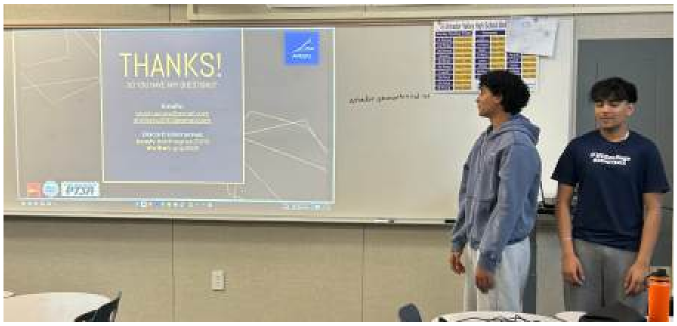

Fig. 19: AVBotz Mech members teaching students during ACE Code Day

Fig. 20: Donlon Elementary Engineering Night

C. Club Fair

AVBotz is dedicated to offering opportunities to other high schoolers as well, especially as high school being a critical period of learning. A key way AVBotz accomplishes this is through club fairs where AVBotz members explain what is done in the club. AVBotz attends all three club fairs throughout the year and makes a constant effort to introduce as many people as possible to the club. Furthermore, attendance at the club fair allows AVBotz students to contribute to school spirit and inspire others.

D. Livermore Innovation Fair

AVBotz is devoted to spreading knowledge, collaborating with local organizations, and engaging with the community through STEM outreach. At the Livermore Innovation Fair, we showcase RoboSub and explain the design and inner workings of our autonomous underwater vehicle, Marlin V3, to students, families, and visitors from all over the area.

What makes the Livermore Innovation Fair especially meaningful is its focus on innovation, education, and community collaboration. The event brings together many unique organizations that contribute to society through science, engineering, and technology. In alignment with the spirit of the event, AVBotz highlights the potential of autonomous vehicles and demonstrates how robotics can be used to solve real-world problems.

The fair also gives our team valuable opportunities to exchange ideas with other innovators and gain insights that help us improve Marlin V3 and refine our engineering processes. AVBotz is honored to participate alongside dozens of interactive scientific displays that inspire curiosity and creativity within the community.

Fig. 21: The Livermore Innovation Fair

APPENDIX H
ARCHITECTURE BLOCK DIAGRAMS

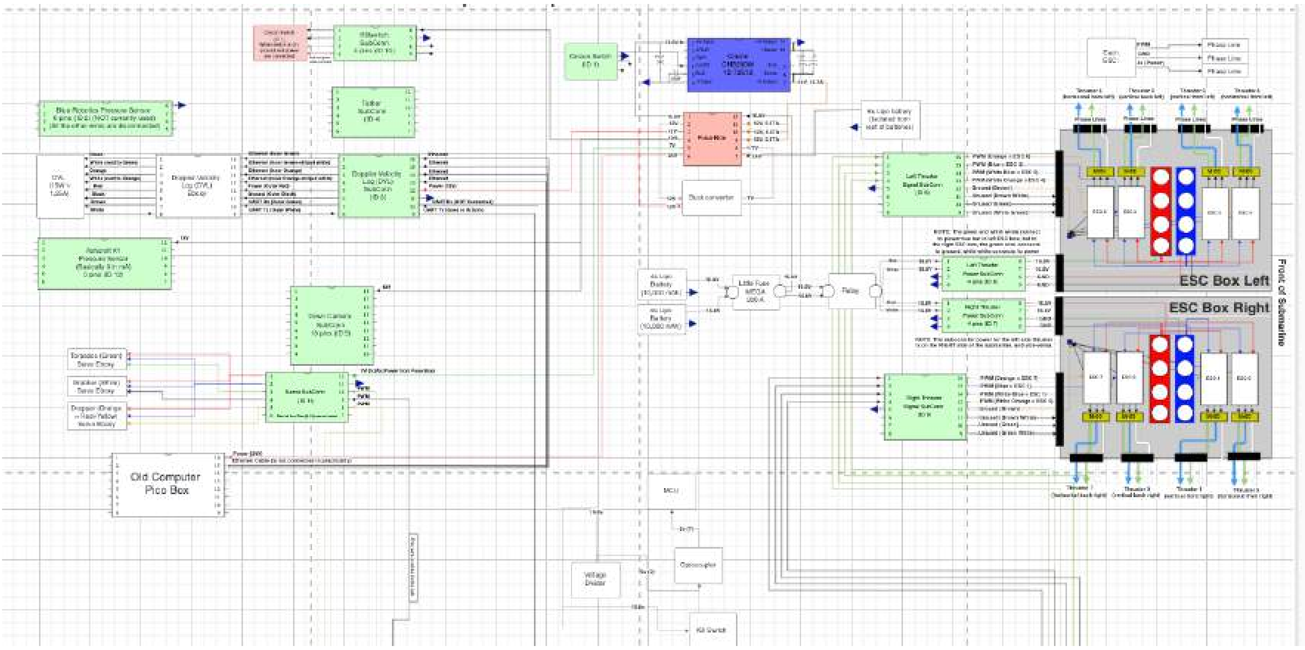

Fig. 22: Electrical architecture overview.

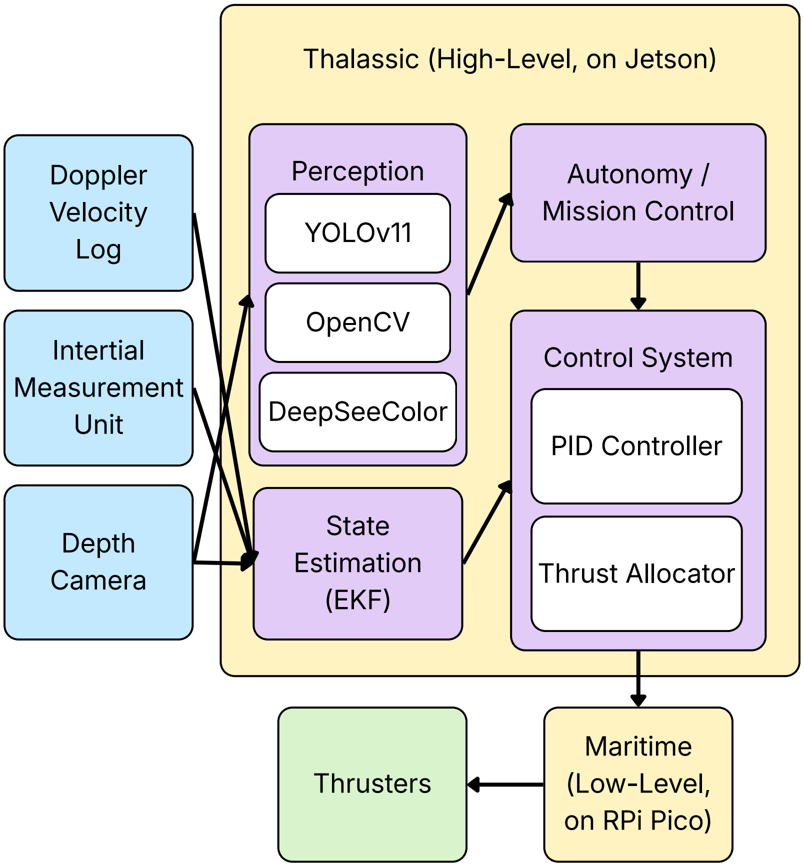

Fig. 23: Software architecture overview.

APPENDIX I
3D MODELS OF ELECTRICAL BOARDS

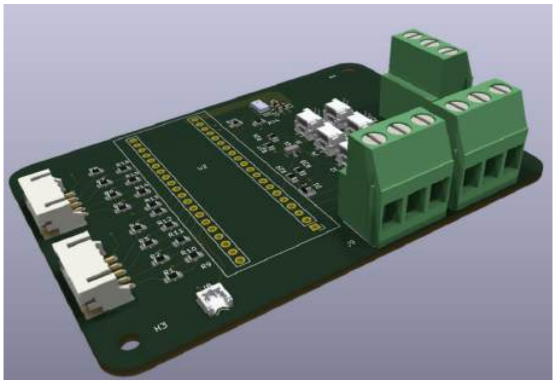

Fig. 24: 3D Render of Sensor Stack Board

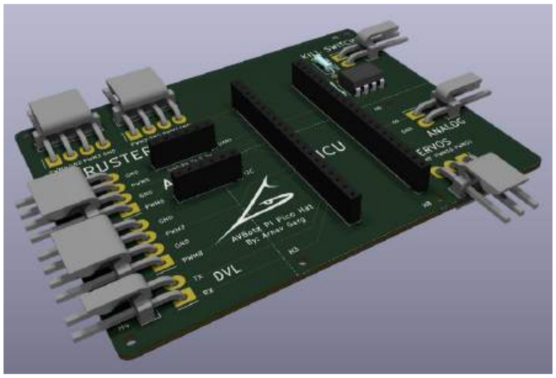

Fig. 25: 3D Render of Raspberry Pi Baseboard

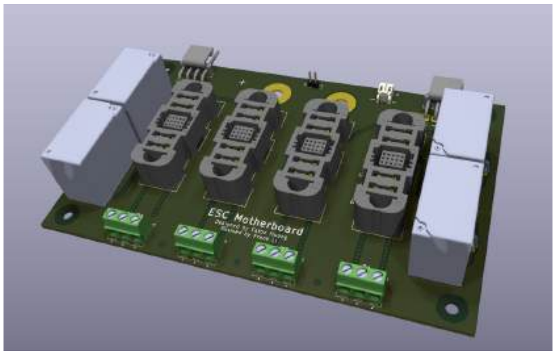

Fig. 26: 3D Render of ESC Motherboard

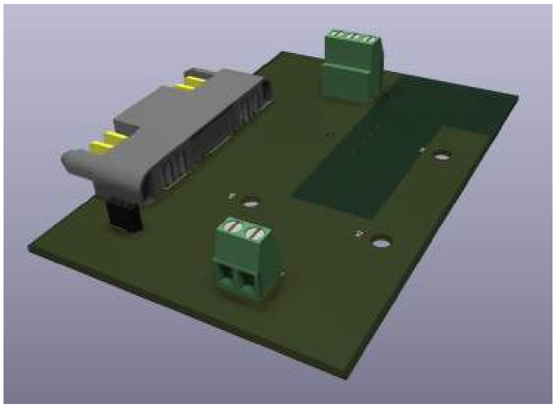

Fig. 27: 3D Render of ESC Daughterboard

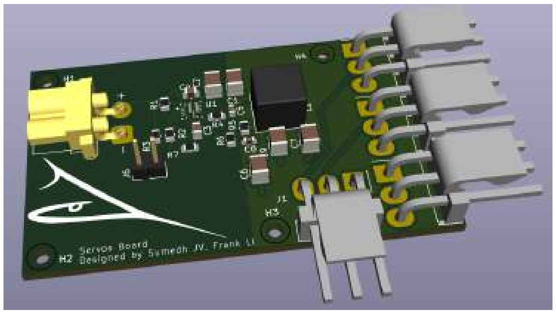

Fig. 28: 3D Render of Actuator Power Distribution Board

APPENDIX J
COMPONENT SPECIFICATIONS

Component	**Vendor**	**Model / Type**	**Specifications**	**Custom / Purchased**	**Cost**	**Year of Purchase**
Frame	Custom	Aluminum 6061-T6	90.50 cm × 63.50 cm × 33.34 cm	Custom	Sponsored	2023
Main Waterproof Enclosure	In-House	Acrylic hull sealed with two rubber O-rings	Diameter: 24 cm	Custom	$200	2016
Waterproof Connectors	SubConn	Circular Series SubConns	Micro-Circular Series, Power Series	Purchased	$1500	2015
Thrusters	Blue Robotics	T200 Thrusters	113 mm length	Purchased	$200 ea.	2022
Electronic Speed Controllers	Blue Robotics	Basic ESC	17.1 × 32 × 3.3 mm; 7–26 V; PWM	Purchased	$38 ea.	2023
Microcontroller (Motor Control)	Raspberry Pi	Pico	2 MB Flash, 264 KB SRAM	Purchased	$6	2025
Microcontroller (Sensor Stack)	WeAct	Black Pill V3.0	512 KB Flash, 96 KB SRAM	Purchased	$10	2025
Batteries	ZEEE Power	4S	10000 mAh, 14.8 V	Purchased	$195	2022
DC–DC Converter	Cincon	CHB200W1-2-72S12	200 W, 16 V to 12 V	Purchased	$185	2023
Computer	NVIDIA	Jetson AGX Orin Dev Kit	414 × 311 × 182 mm; 275 TOPS; 2048-core GPU; 12-core CPU	Purchased	Sponsored	2022
Internal Comm. Interface	—	Ethernet, UART, and CAN	—	—	—	—
External Comm. Interface	—	Ethernet	1 GB/s	Purchased	Included with SubConn	2015
Doppler Velocity Log (DVL)	Water Linked	A50	5 cm–50 m altitude; 600 m depth rated; Ethernet & Serial; 1 MHz	Purchased	Sponsored	2022
AHRS	PNI Sensor	NaviGuider	Heading accuracy: 2° rms; UART	Purchased	Sponsored	2023
Pressure Sensor	Blue Robotics	Bar-30	±2.9 psi; I^{2}C; 2.5–5.5 V	Purchased	$85	2023
Front Camera	Luxonis	OAK-D Pro Fixed Focus	60 fps; 4056×3040; 12 MP	Purchased	$120	2025
Down Camera	FLIR	BFS-U3-13Y3C-C	1280×1024; 1.3 MP; 170 fps; CMOS	Purchased	Sponsored	2025
Down Camera Lens	Theia	SY125M	1.3 mm; 5 MP; H: 125°, V: 119°	Purchased	Sponsored	2015
Algorithms: Vision	—	—	OpenCV, YOLO11, DepthAnythingV2, DeepSeeColor	Open Source	—	—
Algorithms: Localization / Mapping	—	—	robot_localization, Extended Kalman Filter (EKF)	Open Source	—	—
Software: Autonomy	—	—	BT.CPP, C++ ROS2 Nodes	Custom	—	—
Software: Frameworks	—	—	ROS2 Jazzy, Stonefish, Zephyr RTOS, micro-ROS	Open Source	—	—
Team Size	38 members					
Expertize Ratio (HW : SW)	3:1 + 10 Business					
Testing Time: Simulation	125 hours					
Testing Time: In-Water	90 hours					