RoboBoat 2025: Technical Design Report

Name 1, Name 2, Name 3, Name 4, Name 5, Name 6
Military Technical College, EGYPT

***Abstract*—MW is debuting the BOAT NAME autonomous surface vehicle, for RoboBoat 2026. After our last season appearance, we had a challenge to build a more reliable, modular, and well-integrated platform, we plan to attempt every task at competition. While our team has prepared for every task, we prioritized the challenges focusing on navigation to ensure a vehicle with reliable maneuverability and navigation with a strong emphasis on the body and the communication before expanding our focus onto other tasks. Our strategy focuses on adaptability and reliability to create a well-functioning system that works as a base for further development and improvements. Following through on our commitment to show new promise this year's season, we improved system integration and the testing process has been thorough in multiple environments to ensure a reliable performance.**

I. Competition Strategy

This season our goal is to present a vehicle that provides mechanical resilience and well-integrated system of electrical and software. With that, our goal is to attempt every task at competition; however, we prioritized building detection model that is well refined Guided by this strategic view, we concentrated on detection tasks (Tasks 1, 2 ,3, and 4) to improve our existing vision model. While (Tasks 5, 6) will be assigned at a lower priority, this decision was taken upon studying our capabilities and hardware requirements. As we prioritized achieving consistent performance before focusing on the additional complex systems.

A. Environment

The strong waves and winds of Nathan Bendrson Park during our participation in 2025 season proved to be a challenge for our [BOAT_NAME] so this season we expect to face the same circumstances but we are confident after the major material improvements and structural redesigns to be able to withstand such circumstances. *INSERT PIC OF BOAT STRUGLLING*

B. Task completion

1) Task (1, 2, 3, 4) :

The BOAT_NAME can identify the locations of red and green buoy gates maintaining safe distance from them to ensure avoiding them. Buoy colors and numbers are detected and location and time stamps will be recorded.

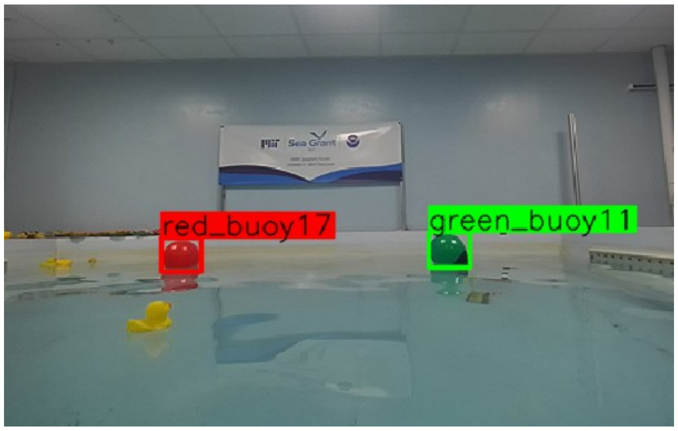


Fig. 1: red and green buoys detected by our fine-tuned YOLOv4 model. [REPLACE PIC]
Which helps with task 1 and 2 giving us reliable detection and maneuverability. For tasks 3 our YOLO model has been trained and refined to detect the beacon color to determine the direction of rotation with that we can also record the color of the beacon. Lastly, task 4 successful execution required two dedicated mechanical system implementation. The design racquetball shooter uses a flywheel mechanism, which consists of two high-speed counter-rotating wheels that shoots the ball [INSERT SOLID WORK PIC OF THE WHEELS]. For the yellow vessels we have installed a water pump under the boat to shoot water from it's muzzle once the yellow vessel is detected,to assure the high precision aim in this task we used the ZED camera to calculate the distance between the boat and for us to be able to calculate the trajectory of projectile and the vessels so the mechanism only works when the vessels are detected and in range which guarantee

high accuracy.

1) Task (5, 6) :

We chose these tasks to be lower on our priority list because the hardware needed and the more complex requirements that they need nonetheless we are aiming to try all tasks. Task 5 for the docking this year the task approach has changed due to the LED beacons that tells you the availability of each dock, through training our model to detect the beacon color it will record the color of the beacon through a specific ROS2 node that will also receive and compare the numbers indicators which another code will subscribe to this node once the lowest number and the correct beacon color are detected the docking code starts. When the [boat name] start, the docking procedures with small adjustments to thruster values guided by the ZED camera measurement of the distance away from the dock to ensure correct docking without any collisions. For task 6 we faced a lot of problems picking up the required frequencies and tuning the model to recognize and operate after the certain frequency is picked up…..

II. Design Strategy

After our last season, we recognized the need to place an enormous focus on system testing. To ensure a flawless run through all subsystems, we designed a list of requirements for our vehicle, prioritizing a lightweight and portability to make testing process flow smoother. Modeling our previous boat to meet such design requirements proved to be quite difficult between the remaking of the hulls and restructuring the power distribution system. After a lot of effort and testing redesigning the sub-systems to better focus our goals.

A. Hulls

We opted to use the Catamarans due to it proving to be overall better in the context of the RoboBoat competition over other hull designs for their high stability and low drag characteristics [3]. The design of the hulls was made to be symmetric for better maneuverability during the docking and navigation tasks. To increase meta-centric height and therefore stability, the batteries were placed within the hulls and protected by plastic containers [4].

The hulls went through two major iterations. The first pair of hulls weighed 36 lbs, exceeding design specifications by 105%. To reduce weight, we shortened the hulls from 1.8 m to 1.2 m, reduced fiberglass layers, and employed vacuum bagging technology, decreasing total weight to 12 lbs.

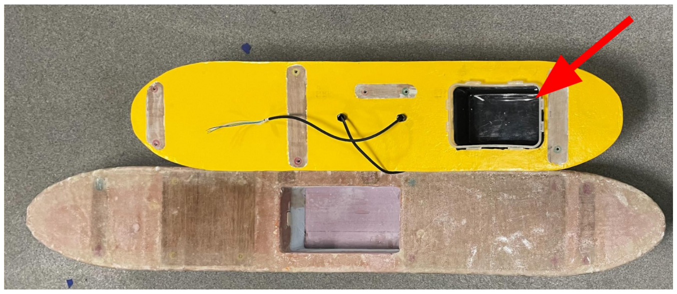


Fig. 2: Top to bottom: new hulls and old hulls. Note the cavity in each hull for holding battery boxes.

B. Propulsion

Using two T200 thrusters in order to make system simple, and it gives us all required movements that all what we need, and powering it by powerful lipo-batteries to take its full thrust power

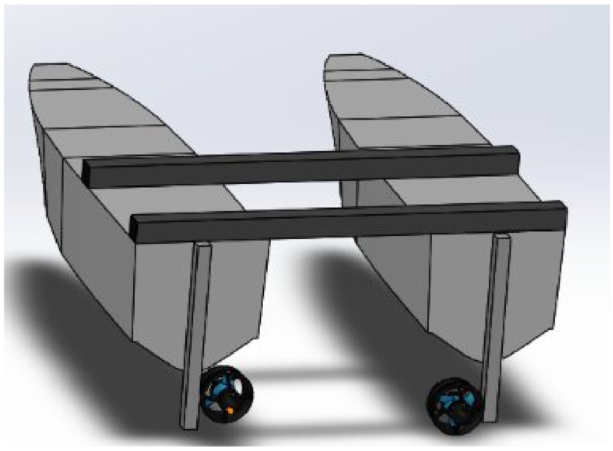


Fig. 3: CAD of thruster vectored configuration. Each T200 thruster can be manually rotated and mounted to the hulls at 22.5° increments. Red arrows indicate direction of thrust on the water.[REPLACE PIC]

C. Electrical box

We used a transparent acrylic box to house all our ASV components due to its lightweight nature, ease of modification and fabrication, and cost-effectiveness. Acrylic was chosen to protect the internal components from water splashes caused by high thrust, wind, and small waves. Based on our experience from last year, exposure to splashing water may have caused issues with our components, which this design helps prevent. The smooth surface of the acrylic enclosure reduces surface friction drag, improving hydrodynamic efficiency and reducing power consumption. Additionally, acrylic helps prevent overheating of the components by allowing better heat dissipation. The box is closed from the top using a hinged cover, which provides secure opening and closing while allowing easy access for maintenance, adjustments, and troubleshooting. The dimensions of the box are **60 × 25 × 15 cm** with a thickness of **5 mm**, providing sufficient internal space, strength, and rigidity while resisting minor impacts during competition.

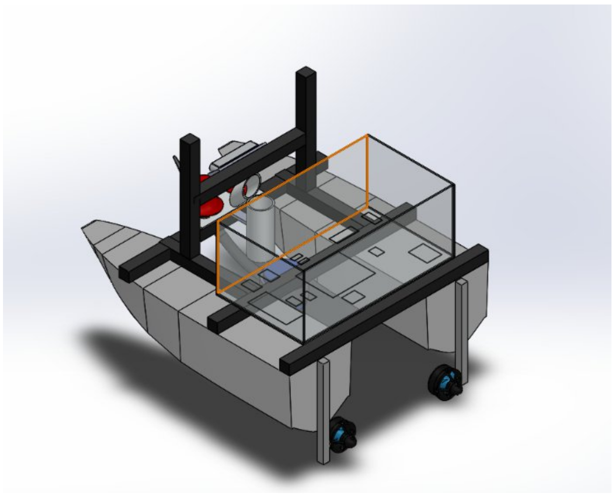


Fig. 4: E box. (1) short base (2) elevated peg boards (3) removable faceplates (4) H gasket and (5) center pole. Lid not pictured.

The LiDAR and ZED are mounted on an aluminum sensor mast to increase range of vision. Vibration-damping mounts, adjustable levels, and multiple points of contact securing the sensor mast to the deck and aluminum beams ensure that the sensors are level and experience minimal vibrations.

D. Mechanisms (Ball Launcher, Turret, Water Cannon)

The mechanism for delivering balls and water to the vessels around the course consists of three subsystems: the turret, the ball launcher, and the water cannon. By integrating these subsystems into a single assembly, we create a reliable method for aiming at targets.

1) Turret: The objective this year was to create a lightweight, precise turret that could withstand both the weight of the ball launcher and water

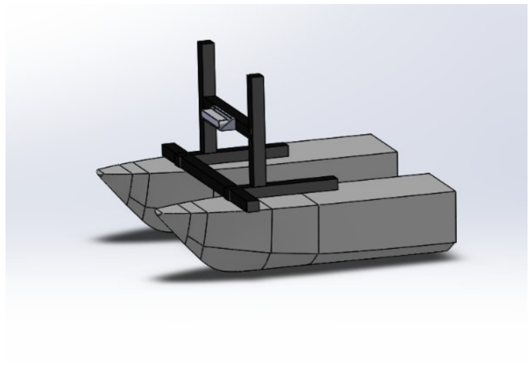


Fig. 6

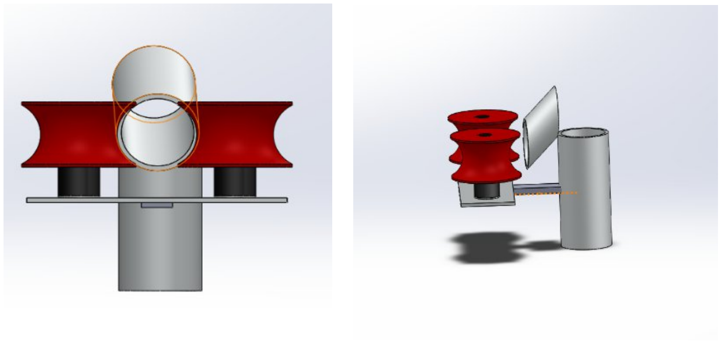


Fig. 6: Full Mechanism assembly. (1) Gear turret

cannon. Initially, a belt-driven turn-table riding on ball bearings was used for rotation. However, testing showed that the bearings would get stuck, so the bearings and belt were removed, and acrylic gears were used to drive the rotation.

2) Ball Launcher: We have designed our ball launcher by assembling two T200 thrusters horizontally and attaching rollers to them to grip and accelerate the balls between the rollers. We benefit from the T200 thrusters' high torque and speed to launch the balls easily. We placed cylindrical tubes to guide the balls into and between the rollers; the inner diameter of the tubes is fit to the ball size, ensuring smooth delivery and reducing friction. Since we are using the same type of T200 thrusters as those employed in our ASV's propulsion system, this simplifies the control system, allowing the use of shared motor controllers and software algorithms.

3) Water Pump: For the water delivery we are using a water pump with a flow rate of 2000 gallons per hour, it provides reliable and efficient water transfer, while being easy to integrate into the ASV. We designed nozzle to allow the water to be directed upward without the need to manually adjust its angle.

E. Electrical System

From previous years participation we reached some sort of confidence in our electrical system because it was able to prove functionality and reliability yet this season we decided to upgrade the power system due to our new demands from upgraded or newly added hardware such as the 5G communication. Our approach to this was that we decided to use two Libo batteries one is connected to the Nivida jetson tx2, the other is connected to the thruster and the other system components.

1) System Overview: For the system overview, we are using two LiPo batteries: a 6S battery providing 22.2 V and a 12 V battery for the thrusters. We are also using a buck converter to step down the voltage to 5 V for auxiliary components. The 22.2 V from the 6S battery is used to power the Jetson TX2. To protect the Jetson from overvoltage and sudden power interruptions, we have placed a UPS between the battery and the Jetson, ensuring stable and safe operation of the onboard computer. The E-stop provides us safely stopping the ASV system by cutting off all the power o the system.

2) Battery Management System (BMS): The BMS monitors battery health and performs battery shutoff. It protects against undervoltage, overcur- rent, and cell imbalance. We chose to use a 200 A NFET transistor on the BMS to implement battery shutoff and E-Stop, preventing the need for heavy and power-hungry contactors. The transistor gate is pulled down, so the system will fail safe in the case of control system failure. The board was tested up to 70 A of load current, which is well in excess of our expected maximum operating current of 40 A.

3) E-Stop Implementation: The previous, WiFi based E-Stop experienced problems with reliability and range. Instead, we chose to use LoRa on the 915 MHz ISM band to communicate E-Stop and manual boat control for its long range and high noise immunity. E-Stop can be triggered by three conditions: 1) the E-Stop button on the boat is pressed, 2) the E-Stop button on the shore-side transmitter is pressed, or 3) the E-Stop loses connection with the shore-side transmitter for over one second. When any of these conditions occur, a signal is sent to both BMS boards on the 4S batteries, which then cut off the output power.

F. Software System Architecture

This competition cycle, the autonomy team prioritized modularity and reproducibility in our code, enabling rapid development and iteration. To achieve this, we organized our code base, `all_seaing_vehicle`, into several Robot Operating System (ROS) 2 packages, covering navigation, perception, controls, and task-specific functionality. This modular approach allows us to continually reuse and refine core features such as navigation and perception across competition cycles, while easily swapping out task-specific code. Additionally, we decided to transition away from another middleware, Mission Oriented Operating Suite (MOOS-IvP), due to two main factors: 1) the steep learning curve of its C++ implementation and features, and 2) the added complexity of the bridge between MOOS and ROS, which unnecessarily complicated the overall system architecture.

In addition to utilizing ROS nodes for continuous communication between processes through a publisher-subscriber pattern, we implemented a hierarchy of ROS actions to manage task execu- tion and vehicle commands (Fig. 7). This request-response model allows for continuous feedback and the ability to cancel or abort tasks as needed. The action servers are organized in three layers, with each layer sending requests to deeper layers:

- Task manager: A ROS node responsible for deciding which task the vehicle is currently performing based on the current state.
- Layer 1 – Task execution: Action servers responsible for responding to high-level requests to execute RoboBoat tasks such as Follow the Path and Docking.
- Layer 2 – High-level commands: Commands requiring additional calculations before requesting Layer 3 commands. Examples include navigating to a target point while avoiding obstacles and completing the process of aiming at a target and shooting.
- Layer 3 – Core commands: Low-level commands for fundamental control of the boat such as waypoint following, station keeping, moving the turret, and firing water/racquetballs.

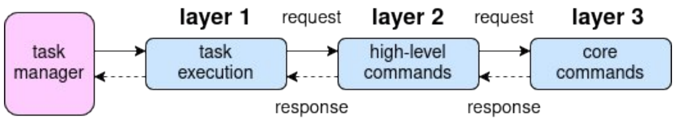


Fig. 7: all_seaing_vehicle's ROS 2 action server hierarchy with the task manager node responsible for sending requests to layer 1.

Underneath Layer 3, we have a network of publishers and subscribers for handling state estimation, map generation, object detection, and hardware interfacing. See Appendix G for more details.

III. Testing Strategy

From past experience, we knew that it was important to not have software testing bottle-necked by hardware development, and that frequent testing was crucial to developing a reliable system. Thus, we devised a plan to test our overall system both in simulation and physically on our sister test boat Minerva, which has a similar sensor and propulsion system to Fish 'N Ships, while the new vessel was under construction.

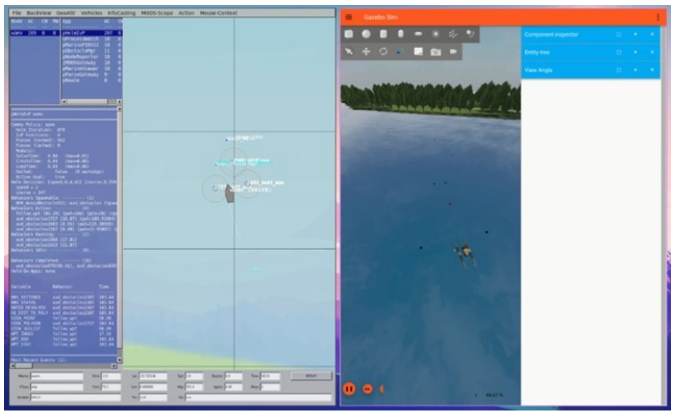


Fig. 8: Gazebo simulation testing.

To facilitate frequent testing of our autonomy stack, we rely on the Virtual RobotX (VRX) Gazebo simulator provided by the Open Source Robotics Foundation. In addition to the default

VRX worlds, we developed custom Simulation Description Format (SDF) files and Python scripts to model RoboBoat tasks such as Navigation Channel, Follow the Path, and Docking. While the simulation provides an idealized environment, it enables rapid prototyping and code validation without interfering with the mechanical team's hardware design process.

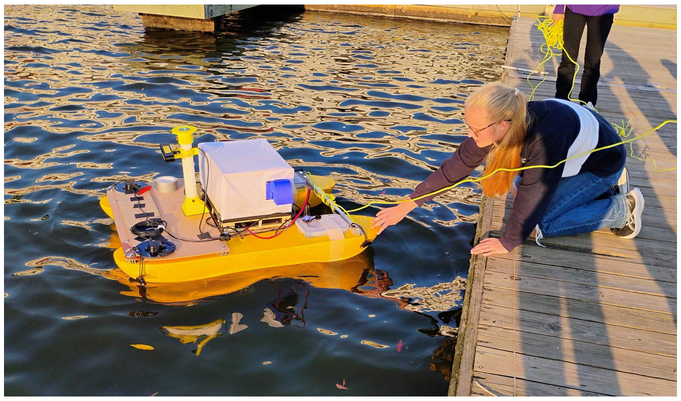


Fig. 9: Physical testing on the Charles River.

We conducted various physical tests to gain a more accurate representation of the competition environment and evaluate components such as sensors and communication systems that often perform differently than in simulation. To ensure consistent progress, we established a timeline to test the boat physically at least once every two weeks. This schedule provided time to iterate on unreliable systems while keeping us accountable to deadlines. For each test, we outlined specific objectives and goals, documenting the results to better plan for future milestones. We began with small-scale tests focused on isolated components (see Appendix B and C) and gradually progressed to validating the entire system through the completion of full tasks (see Appendix A).

We conducted small-scale tests at the MIT Sea Grant test tank to verify sensor and thruster functionality. Additionally, we performed system integration tests to ensure the mechanical, electrical, and software systems were working together as intended. These focused tests allowed us to validate design changes efficiently, avoiding the need for full-scale tests, which are more time-consuming and require additional planning.

Large-scale indoor physical tests were conducted at the MIT Z-Center swimming pool. Due to the lack of a reliable GPS signal indoors, we used Nav2's Adaptive Monte Carlo Localization algorithm [5], [6] on a pre-built map using Cartographer [7] for accurate localization of the vehicle. Although GPS systems could not be tested in this environment, we focused on isolating and testing tasks such as Follow the Path, Docking, and the Speed Challenge. These indoor tests proved especially valuable during the harsh winter months when outdoor testing was not feasible.

Finally, full-scale tests were conducted at the Charles River, where we extensively tested GPS, WiFi, and LoRa systems to ensure accurate robot localization and reliable communication between the shoreside and the boat. Beyond isolated tests, we evaluated the autonomy stack's ability to complete a sequence of tasks, simulating the competition environment. Outdoor testing also allowed us to collect training and testing data under various weather conditions, further improving the robustness of our system.

IV. Acknowledgments

The MW team's achievements and the successful upgrade of our Autonomous Surface Vehicle (ASV) would not have been possible without the motivation and generous financial support of the Military Technical College. We would like to extend our sincere appreciation to Brigadier General Hossam Ragheeb and Captain Adel Mohab for their invaluable advice, guidance, and continuous support throughout the upgrade.

We are also deeply grateful to the Marine Engineering Department for providing specialized tools, technical expertise, and essential knowledge that greatly contributed to our development process. In addition, we acknowledge the Military Technical College for facilitating access to various locations and diverse environments that enabled comprehensive testing and validation of the ASV.

Finally, we would like to thank all the cadets for their valuable feedback, and practical suggestions, which played an important role in refining and improving our system.

REFERENCES

[1] K. Nonami, F.Kendoul, S.Suzuki, W. Wang, D. Nakazawa (2010) Autonomous Flying Robots for Unmanned Aerial Vehicles and Micro Aerial Vehicles.
[2] Team Handbook - RoboBoat 2026
[3] Ardupilot, "Mission Planner," Ardupilot, 2023.https://ardupilot.org/planner
[4] “ROS Documentation”, OSRF. Available: http://wiki.ros.org/Documentation.

[5] L. Joseph, Mastering ROS for Robotics Programming, 2nd ed. Birmingham, UK: Packt Publishing, 2018.

[6] GitHub, “YOLOv4-Tiny · GitHub topics,” GitHub. [Online]. Available: https://github.com/topics/yolov4-tiny

[7] mavlink, “MAVROS: MAVLink to ROS gateway,” GitHub. [Online]. Available: https://githubgithub.com/mavlink/mavros.

[8] R. Szeliski, Computer Vision: Algorithms and Applications. New York, NY, USA: Springer, 2010.

The Task Manager node supervises mission execution by activating individual task nodes and continuously monitoring a global recall signal. Upon reception of the recall signal, the Task Manager immediately aborts all running tasks and commands the Pixhawk flight controller to execute a Return-to-Launch maneuver, ensuring system safety.

[1] G. Jocher, A. Chaurasia, J. Qiu, "Ultralytics YOLOv8," 2023. [Online]. Available: https://github.com/ultralytics/ultralytics.

[2] P. Couser, "An Investigation Into the Performance of High-Speed Catamarans in Calm Water and Waves," Mar. 1996.

[3] J. Me´gel and J. Kliava, "Metacenter and ship stability," American Journal of Physics, vol. 78, no. 7, pp. 738–747, Jul. 2010, doi: 10.1119/1.3285975.

[4] S. Macenski, F. Martin, R. White, J. Clavero, "The Marathon 2: A Navigation System," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.

[5] S. Macenski, T. Moore, DV Lu, A. Merzlyakov, M. Ferguson, "From the desks of ROS maintainers: A survey of modern & capable mobile robotics algorithms in the robot operating system 2" Robotics and Autonomous Systems, 2023.

[6] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-Time Loop Closure in 2D LiDAR SLAM," Robotics and Automation (ICRA), 2016.

Appendix A
Test Plan and Results

Our team had physical, simulation, and bench tests to allow independent development of subsystems. To do so, the workload is broken down into three main sub-teams: mechanical, electrical, and autonomy. From there, we have even smaller project teams working on designing, prototyping, and testing subsystems of Fish 'N Ships. Every week, we have system integration meetings where we check in on each sub-team's progress and make adjustments to our testing plan as needed.

Subsystem: every subsystem has a design timeline as follows: 1-2 weeks for design/research, 1 week for design review and material lead time, 1-2 weeks for prototyping, 1 week for integration, and reiteration as needed. Please see detailed examples of our subsystem testing in Appendices B and C.

Full-system: after independent subsystems are tested, we integrate them onto the main vessel and deploy the vessel to ensure that the components are behaving as expected. Our full system tests follow the Testing section of the Gantt chart timeline in Appendix H, and more details are included in the Simulation Tests and Physical Tests subsections below.

A. Simulation Tests

Simulation testings were done through the VRX Gazebo simulator as mentioned in the Testing Strategy section of the paper. For these tests, we needed a Linux based computer to run the simulations. Thankfully, we were able to obtain a dual booted Toughbook laptop borrowed from the MIT Sea Grant lab in order for all the autonomy members to run the simulation. There are far fewer safety factors/risks to account for since we run these simulations in the laboratory space. The Autonomy section of the Gantt chart (see Appendix H) describes the simulation testing schedule, where each system is developed over the course of two to three months. The objective of these tests were to validate that the code is working as intended, and the simulations were run continuously throughout the development process to verify that the subsystems were working.

B. Physical Tests

Physical tests allowed us to verify our design, ensure progress, and integrate independent components. Since our team built a new vehicle and redesigned all of the hardware, most of the base platform was developed independently. Once the base platform was integrated, we had biweekly full system tests that followed the schedule of the Testing section of the Gantt chart in Appendix H. To simulate the competition environment, we tested outdoors at the MIT Boathouse, where a dock was available by the Charles River for easy deployment of Fish 'N Ships. However, other groups on campus also utilized this space and it was only available on the weekdays, so we often had to work around conflicting schedules. To replicate the competition environment for practice, we built task props such as the dock and the delivery vessels. We also reused Polyform buoys from previous years for object detection testing. There were significantly more risks when it came to physical testing because we needed to make sure there were enough helpers for carrying the boat and that we were following safety measures in working with the river. To ensure the safety of our members, we coordinated tests with the boathouse manager to ensure the weather was appropriate and that safety protocols at the docks were followed. When the weather posed a severe risk for hypothermia and frostbite, we moved our tests indoors to the pools at the MIT Z-center to ensure the safety of our members. Below, we will describe the various stages of our full system testing and the corresponding objectives and results.

Early Stage — Completion of Fish 'N Ships

Objectives: We started testing our new vessel as soon as it was completed. During this stage, we mainly focused on achieving teleoperation functionality with the new vehicle and adjusting the vessel to ensure a desired center of mass and thruster positions.

Results: We began testing our new vessel immediately after its completion, focusing initially on achieving teleoperation functionality and adjusting the vehicle to optimize the center of mass and thruster positions.

During this phase, we encountered several challenges:

- The customized electronics box was difficult to access, complicating debugging efforts.
- The sensor mount was unstable, hindering the effective use of the camera and LiDAR.
- Water accumulated in the cavities between the battery box and the hulls.

To remedy these issues, we iterated on the design of the EE box and sensor mount, and we waterproofed the cavities to prevent water pooling in future tests.

Mid-stage — Fish ’N Ships with New Features

Objectives: After the initial round of tests, Fish ‘N Ships had new features integrated in terms of the electronics and the updated mechanical parts. Our objective for these tests was to validate that the new electronics function as expected, the updated mechanical components met our requirements, and the GPS and waypoint following portions of autonomy worked.

Results: We were able to successfully set up the new GPS system after a few tries, and most of the electronics behaved as expected. The newly designed electronics box was a lot more accessible. However, like most of our tests, unexpected issues arose:

- One of the tests happened on a windy day, and our vehicle was a lot more soaked than usual. We realized that we needed to better waterproof our connectors into the EE box.
- The sensor mount was still not as stable as we expected.

Late-stage — Competition Preparation

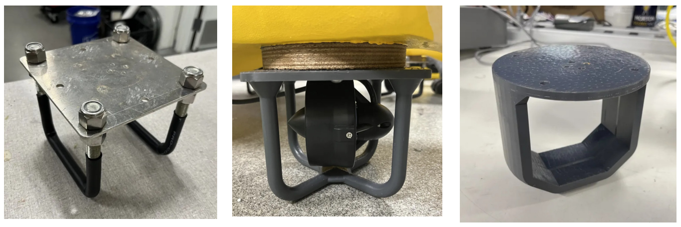
Objectives: Our goal was to simulate the competition course to practice the tasks, test the perception system of our autonomy stack, and integrate the latest iteration of the sensor mount.

Results: We were able to record ROS bags for the autonomy team to test the perception system on. Some unexpected problems were:

- With more nodes running for our software functions, our onboard computer unexpectedly crashed.
- Our batteries drained a lot faster than expected even though the current the electronics system was drawing seemed to be reasonable.

Appendix B
Thruster Cages

The initial cages weighed 1.9 lbs per cage and together contributed 10% of the ASV’s overall weight. To reduce weight, two alternative designs were proposed; the first design weighs 0.6 lbs per cage and the second weighs 1.04 lbs per cage, both were printed in Formlabs Tough 2K resin. Onshape FEA showed the first design deflected by approximately 0.0001 in, under loads equivalent to the boat being dropped from 1 m. Physical tests also supported this, as each cage could support over 150 lbs. Given these results, the first design is optimal considering its proven functionality and lower weight. The redesign decreased cage weight by 68.4%; the cages now contribute only 4% of overall vessel weight.



Initial Design (1.9 lb each) | Design 1 (0.6 lb each) | Design 2 (0.9 lb each)

Fig. 10: Different iterations of the Thruster Cages.

Appendix C
EE Box Lid Design

To calculate the maximum load on the walls of the EE box lid, we calculated drag force on the box in 5 m/s gusts:

$$F_d = \frac{1}{2} \cdot c_d \cdot A \cdot \rho \cdot v_{\text{rel}}^2$$

where

F_d = drag force,

c_d = drag coefficient ≈ 1.05,

A = ⊥ Area of Lid = 0.14 m^2 along long face, ρ = density of air at STP = 1.2 kg/m^3,

v_{rel} = relative velocity of flow ≈ 7 m/s.

$$F_d = \frac{1}{2} \cdot 1.05 \cdot 0.14\,\text{m}^2 \cdot 1.2\,\frac{\text{kg}}{\text{m}^3} \cdot 49\,\frac{\text{m}^2}{\text{s}^2}$$
$$= 4.3\,\text{N}$$

The cross members make analytical methods difficult so we used Onshape's FEA tool:

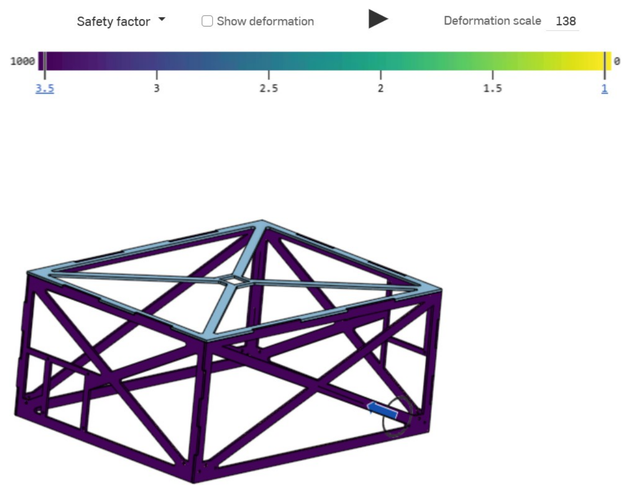


Fig. 11: FEA of EE box lid under 5 m/s gusts.

With a safety factor >3.5, the lid is well within the specification.

Appendix D
Ball Launcher Calculations

To simplify aiming, the launcher was designed to hit the 0.584 m tall target from a maximum distance of 3 m without adjusting launch angle. By measuring the time to travel 1.82 m over 5 trials, we found that the muzzle velocity of a launched ball is 7.15 m/s. Various launch angles were simulated to determine how many targets would be hit within a set distance.

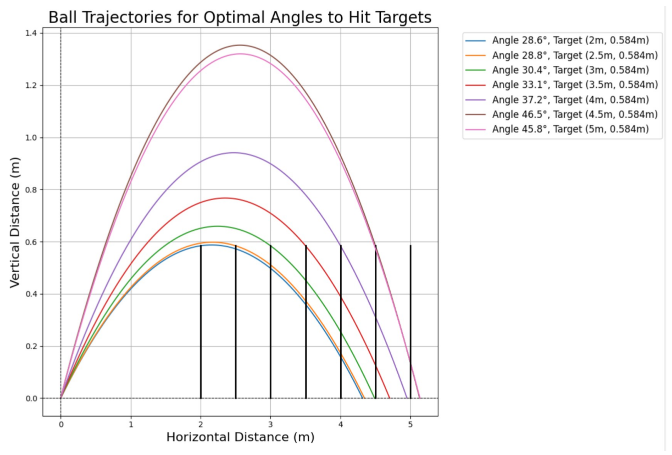


Fig. 12: Different ball trajectories depending on angle. Air resistance neglected. Black lines represent target height.

ensuring that any launch between 0 and 3 m will not overshoot the 0.584 m tall target. As long as the boat shoots within this range and in the correct x-y direction, it will hit the target.

Appendix E
Water Delivery Calculations

Our pump maintains a volumetric flow rate of $273.4 \cdot 10^{-6}$ m^3/s. To determine the appropriate nozzle size to reach s_{min} = 3 m, we utilized the formula for projectile motion:

$$s = \frac{v^2 \sin(2\theta)}{g},$$

s = horizontal range,

v = fluid flow $\frac{4Q}{\pi d^2}$,

g = gravitational acceleration (9.81 m/s^2), θ = initial angle (45° for maximum range).

Substituting $v = \frac{4Q}{\pi d^2}$ into the range equation gives:

$$s = \frac{16Q^2}{\pi^2 d^4 g}.$$

The optimal angle is 28.6°. This angle exceeds the 3 m requirement and peaks at the top of the target,

Using this formula and the given flow rate, and assuming that the pressure gradient along the tubing is negligible, we have found that a nozzle outlet diameter of $d = 1/4$" ($6.4 \cdot 10^{-3}$ m) will reach
$d =$
7.36 m.

Appendix F
Electrical System Diagram

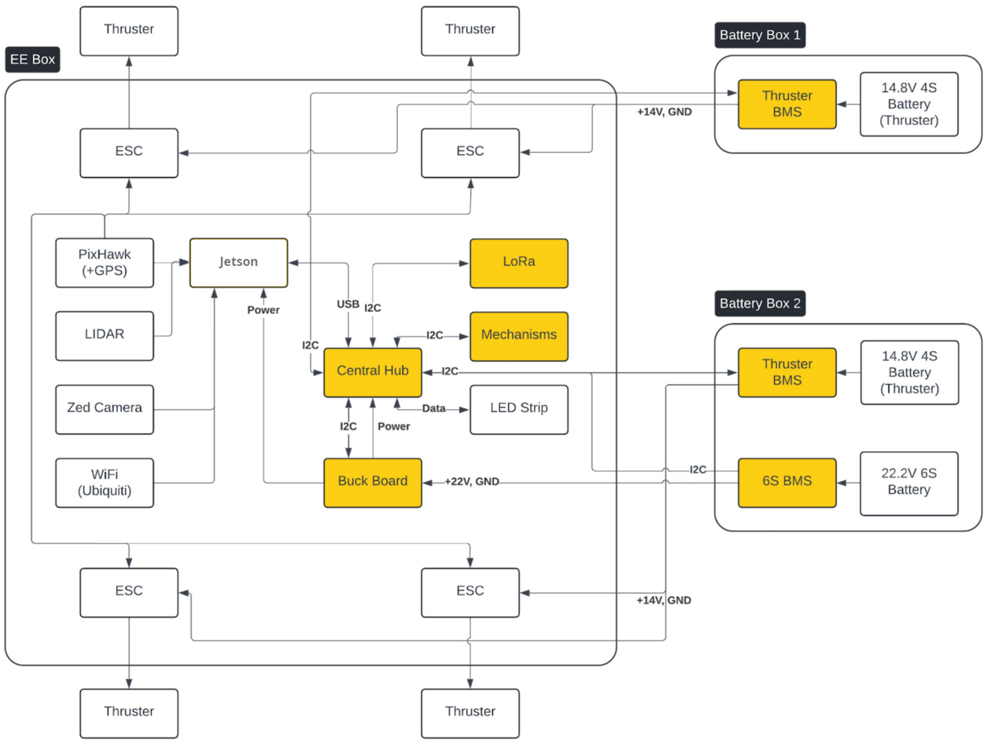


Fig. 13: Electrical system overview. Custom boards are indicated in yellow.

Appendix G
Software System Architecture

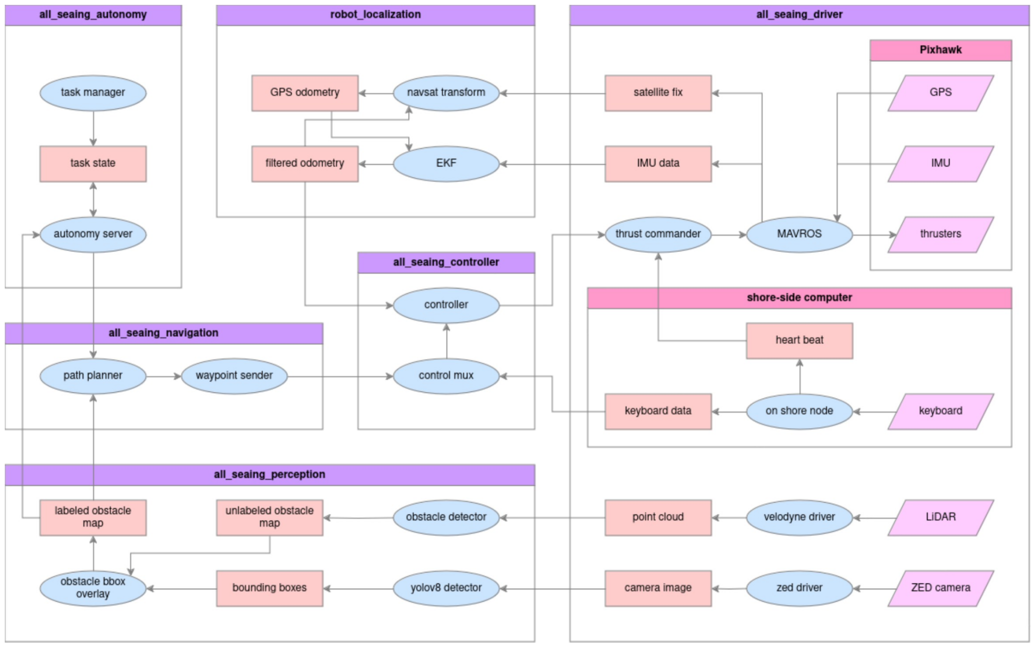


Fig. 14: Software System Architecture.

Appendix H
Arcturus Team Gantt Chart

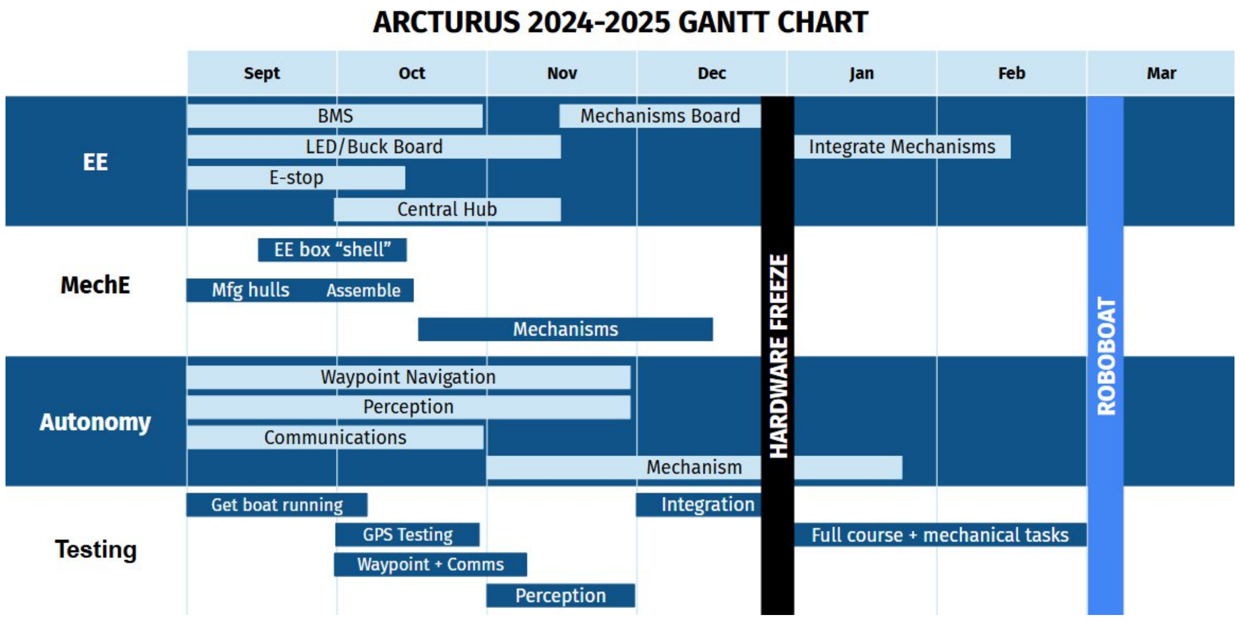


Fig. 15: The team Gantt chart consists of subsystem projects, organized by the respective sub team. The long bars that span horizontally indicate the relative timeframe for the design, prototype, and test of the subsystem.