ZeroneTech Kılıç ROBOBOAT 2026 Technical Design Paper

Alanya Alaaddin Keykubat University

[1] Kenan Ahmet Türkdoğan, [2] Mustafa Görgülü, [3] Buse Demirci, [4] İlayda Aydın, [5] Aliihsan Kerem Kahya, [6] Kaya Furkan İpekci, [7] Ozan Eğe, [8] Seher Emir, [9] Hasan Demirci, [10] Fatma Sıla Kaplan, [11] Yahya Murat Turgut, [12] Yavuz Aydın, [13] Murat Uysal, [14] Muhammet Mermer, [15] Abdurahman Karakurt, [16] Yağız Ata Ulutaş

Abstract

This report presents the system level design and competition driven development strategy of Kılıç, an autonomous surface vehicle (ASV) developed by the ZeroneTech team for the RoboBoat 2026 competition. The team adopted a reliability-first autonomy strategy, aiming to maximize repeatable scoring performance by establishing robust baseline capabilities before introducing higher complexity behaviors.

The competition strategy prioritizes core RoboBoat tasks that validate fundamental autonomy, including Navigation Channel traversal, perception based navigation, and obstacle detection and avoidance. These tasks were deliberately selected as early milestones to verify propulsion stability, perception reliability, localization consistency, and mission continuity prior to attempting advanced interaction based challenges. This prioritization directly shaped architectural decisions and system level trade-offs.

High-level autonomy functions perception, localization, navigation, and mission execution are executed on an NVIDIA Jetson Orin NXUbuntu 22.04 [1] and ROS 2 [2], while safety-critical low-level control and hardware fail-safe mechanisms are handled by a Pixhawk Cube Orange flight controller [3]. To address known maritime perception failure modes such as glare, reflections, and sparse visual features, Kılıç employs a multi modal perception architecture that fuses stereo vision from a ZED X camera [4] with geometric sensing from a Unitree L2 4D LIDAR [5].

The vehicle utilizes a trimaran hull with differential thrust propulsion to enhance stability and maneuverability while minimizing mechanical complexity. Mission execution is governed by deterministic behavior trees [6] to ensure transparent and testable decision logic. Validation follows a simulation-first, test-driven workflow using the VRX framework [7] and Gazebo [8], combined with staged subsystem and field testing. This paper demonstrates how competition strategy, architectural design, and validation methodology are coherently integrated to produce a robust and competition-ready autonomous system.

Acknowledgements

The ZeroneTech team gratefully acknowledges Alanya Alaaddin Keykubat University for providing laboratory facilities, fabrication resources, and institutional support throughout the development of the Kılıç autonomous surface vehicle. The team thanks faculty members and mentors for their technical guidance and constructive feedback during system design, integration, and testing phases.

We also acknowledge the contributions of all ZeroneTech team members involved in mechanical fabrication, electrical integration, software development, and field testing. The participation of high school students within the team is additionally recognized as part of the project's educational outreach and mentoring objectives.

1. Competition Strategy

1.1. Overall Strategy

The competition strategy for RoboBoat 2026 is structured around a task-oriented and risk-aware development philosophy, with the primary objective of maximizing achievable points while preserving system reliability under competition constraints. Rather than attempting to address all tasks simultaneously, the team analyzed the RoboBoat mission set to identify foundational autonomy capabilities that must operate reliably before higher-complexity behaviors can be safely introduced. Fundamental tasks such as Navigation Channel traversal, basic autonomous navigation, and obstacle detection were intentionally prioritized as early validation milestones. Successful completion of these tasks confirms the stability of propulsion control, perception pipelines, localization estimates, power distribution, and software execution under real-world conditions. Tasks requiring additional mechanical actuation or tighter interaction tolerances are treated as modular extensions, ensuring that partial failures in advanced behaviors do not propagate into system-wide instability.

Development followed an iterative and modular workflow, with clearly defined subsystem boundaries and short integration cycles. This approach enables rapid debugging, controlled system growth, and early identification of failure modes. Extensive simulation-based validation is performed prior to hardware deployment, allowing the team to conserve limited testing resources while reducing operational risk during on-water trials.

1.2. Task Strategy Summary

To balance scoring potential with system reliability, competition tasks are organized into progressive development phases based on dependency and required system maturity:

1.2.1. Phase 1 – Mandatory Navigation:
Validates propulsion interfaces, control stability, perception I/O, localization consistency, power integrity, and software robustness through autonomous gate traversal.

1.2.2. Phase 2 – Perception-Based Navigation:
Focuses on obstacle avoidance and speed-related tasks, emphasizing real-time sensor fusion and local planning without introducing additional mechanical complexity.

1.2.3. Phase 3 – Object Interaction and Delivery:
Developed independently due to added actuation and stabilization requirements, preventing coupling with baseline navigation behaviors.

1.2.4. Phase 4 – Complex Interaction Tasks:
Includes docking and sound-based missions, implemented as modular autonomy behaviors to ensure that failure of a complex task does not compromise core system operation. This phased strategy enables controlled progression from mandatory scoring tasks to advanced challenges while maintaining predictable system behavior throughout competition runs.

1.3. Trade-Off Analysis

Key architectural decisions were evaluated under RoboBoat-specific constraints, including outdoor lighting variability, dynamic marine motion, and strict safety requirements. Based on early testing and prior field experience, the team deliberately selected stereo vision over active depth cameras, as depth accuracy degraded significantly under direct sunlight. A 4D LIDAR was chosen over 2D alternatives to provide full spatial awareness and improved obstacle geometry validation.

Rolling-window local mapping was preferred over global SLAM approaches due to unreliable loop closure in water environments with sparse and dynamic features. Deterministic Behavior Trees were selected over learning-based decision systems to ensure predictable, debuggable, and safety-compliant autonomy.

A lightweight LLM-based cognitive module is employed strictly as a diagnostic and recovery advisory tool, selecting from a predefined, safety-validated recovery set without directly generating control commands [9].

2. Design Strategy

2.1. System Architecture

Kılıç adopts a layered system architecture that separates high-level autonomy processing from low-level safety-critical control. The NVIDIA Jetson Orin NX executes perception, sensor fusion, localization, navigation, and mission-level decision making using Ubuntu 22.04 and ROS 2. Low-level actuation, state estimation, and fail-safe mechanisms are handled by a Pixhawk Cube Orange flight controller. This separation ensures that, in the event of high-level software degradation, hardware-level safety behaviors remain deterministic and responsive.

2.2. Navigation and Mission Execution

Due to the non-holonomic nature of the differential thrust platform, navigation is implemented using the ROS 2 Nav2 stack [10]. Global planning is performed with the Smac Hybrid-A Planner [11], producing kinematically feasible paths aligned with the vehicle's turning constraints. Local trajectory optimization is handled by Model Predictive Path Integral (MPPI) control [12], which samples and evaluates candidate trajectories under the vehicle model, providing robustness against environmental disturbances such as wind and wave-induced drift. Mission execution is governed by deterministic behavior trees implemented using BehaviorTree.CPP, enabling transparent task sequencing and explicit transition conditions.

This decision logic is visualized and debugged via Groot2 [13], ensuring testable and maintainable autonomy.

A constrained cognitive support module may assist only in predefined ambiguity cases, after which execution authority is immediately returned to the behavior tree, preserving deterministic control flow.

2.3. Mechanical, Electrical, and Perception Design

A trimaran hull configuration is employed to enhance lateral stability and reduce roll-induced degradation of perception and control accuracy. As shown in Figure 1 differential thrust propulsion minimizes mechanical complexity while enabling precise maneuvering.

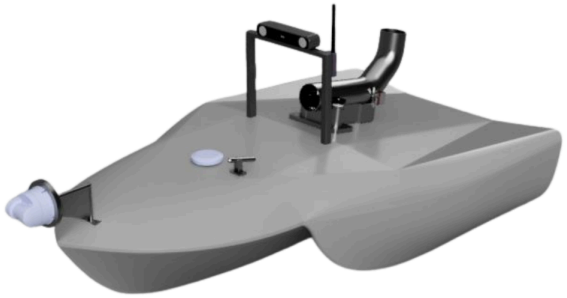


Figure 1

As shown in Figure 3, the electrical architecture utilizes dual 6S Li-Po batteries feeding a centralized power distribution board (PDB) with regulated voltage rails and a physical emergency stop mechanism, ensuring rapid and deterministic power isolation. Perception is implemented as a multi-modal fusion system combining ZED X stereo vision and Unitree L2 4D LIDAR. YOLOv11-based object detection [14] provides semantic classification, while depth projection and LiDAR cross-validation ensure accurate 3D localization and robust obstacle geometry for navigation. A specialized dataset was created to enhance detection reliability.
In the initial stages, over 2,000 virtual images were generated in environments such as Unreal Engine, and in subsequent phases,

over 5,000 images were collected from surface tests and labeled to fine-tune the YOLOv11 model. We also utilized datasets published as open source on Roboflow. This process specifically targeted objects unique to RoboBoat, such as buoys and docking piers, under varying lighting conditions.

2.3.1. Ball Launching and Water Blast System

Figure 2

The ball launching system for the Skeeball task is a fixed-angle mechanism that launches racquetballs using two DC motor–driven rotating disks, with balls fed from a vertical tube via a servo-controlled mechanism. As shown a Figure 2, the Water Blast system uses a 12V high-pressure pump with a fixed-mounted adjustable nozzle, achieving a range increase from 1.5 ft at 0.7 GPM to 13 ft after nozzle integration, operating at up to 80 PSI. Both systems are triggered by the autonomy software when target alignment falls within predefined tolerances, with no active aiming mechanism; targeting is achieved by aligning the vessel using a high-precision MPPI controller.

2.3.2. Acoustic Localization

To address the horn-based signaling tasks defined in the competition handbook, a waterproof, high-fidelity digital microphone is mounted on the vessel's superstructure. The captured audio stream is processed onboard the Jetson Orin NX using Fast Fourier Transform (FFT) algorithms. This allows the system to detect and classify specific target frequencies (600Hz, 800Hz, 1000Hz) in real-time to trigger the corresponding docking or mission sequences.

2.4. Communication and Safety Systems

The communication architecture is designed to prioritize link reliability and safety over high-bandwidth data streaming.

Telemetry & Command: A long-range XBee Pro (900 MHz / 2.4 GHz) telemetry link is utilized to transmit critical vehicle status (Heartbeat, Voltage, Mode) and receive high-level commands (Start/Stop) via the MAVLink protocol. This low-bandwidth approach ensures robust control even in RF-congested environments.

Remote Safety Shutdown: Complying with competition safety rules, a completely independent LoRa-based wireless Emergency Stop (E-Stop) system is implemented. This system physically cuts power to the propulsion motors via a relay upon triggering, regardless of the software state.

3. Testing Strategy

Testing follows a simulation-first approach to ensure reliable autonomy within limited development time. Early validation is performed using Software-In-The-Loop (SITL) testing with the VRX framework and Gazebo Garden. To accelerate development, the standard WAM-V model was used instead of importing the custom Trimaran mesh, allowing the team to avoid complex mesh configuration and focus on validating the ROS 2 software stack.

Although hull geometries differ, this approach effectively verified perception and navigation logic prior to real-world deployment. Simulation results are used as a gating criterion for hardware deployment; autonomy behaviors are transitioned to on-water testing only after demonstrating stable convergence and recovery behavior in simulation.

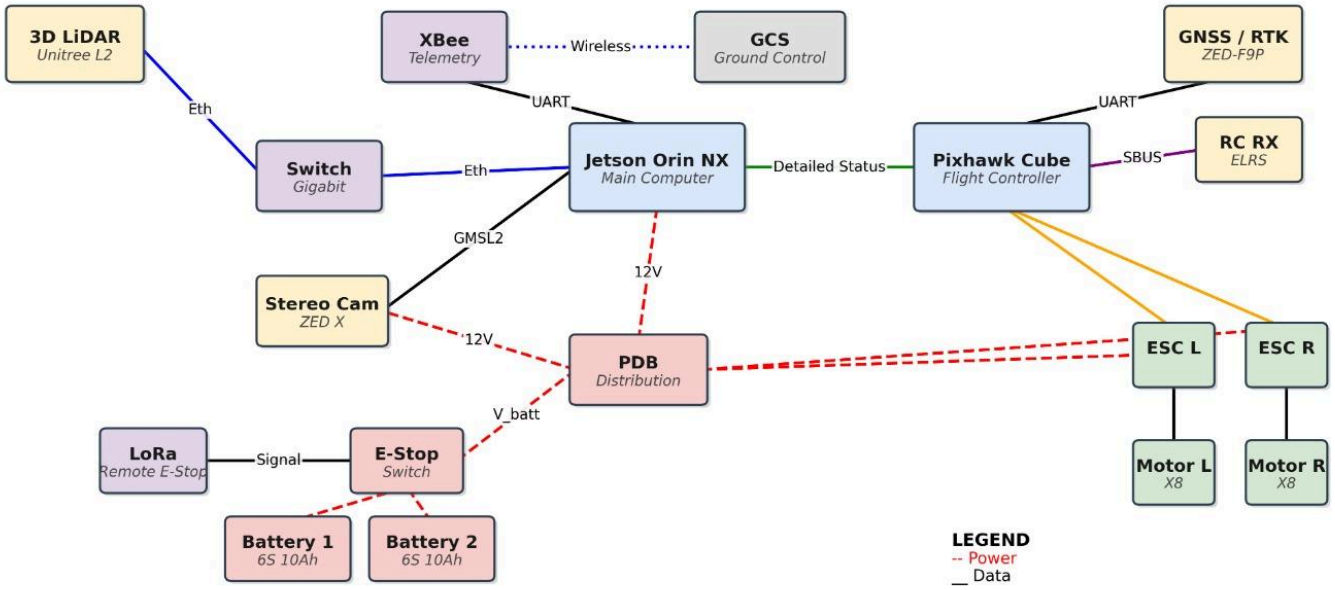


Figure 3 : Hardware System Architecture

Subsystem testing progresses from bench-level validation of power, propulsion, and sensors to staged integration testing within the ROS 2 framework. Controlled field tests focus on sim-to-real calibration, controller tuning, and verification of autonomous behavior under real lighting, wind, and wave conditions. Insights from each testing stage directly inform parameter refinement and mission design, prioritizing repeatability over late-stage feature expansion.

3.1. Preliminary Experimental Results

Initial component-level and integrated validation tests have been conducted to verify performance against design requirements. Key performance metrics, including propulsion benchmarking, battery endurance, and simulation success rates, are summarized in Appendix.

4. Conclusion

Kılıç was developed through a deliberate engineering process that tightly aligns competition strategy, system architecture, and validation methodology around a reliability-first autonomy philosophy. Rather than maximizing task count through rapid feature expansion, the team prioritized repeatable and predictable system behavior, ensuring that foundational autonomy capabilities remain stable under realistic competition conditions.

The adopted two-layer computing architecture separates high-level autonomy from safety-critical control, enabling robust fault isolation and deterministic fail-safe behavior.

Multi-modal perception, combining stereo vision and 4D LIDAR, addresses known maritime sensing challenges such as glare, reflections, and sparse visual features, while behavior-tree-based mission execution ensures transparent, testable, and maintainable decision logic. These design choices collectively limit unnecessary system complexity while preserving extensibility for advanced tasks.

A simulation-first, test-driven validation workflow played a central role in reducing integration risk. Autonomy behaviors were systematically verified in a competition-relevant simulation environment before hardware deployment, and subsequently refined through staged subsystem and controlled field testing. This structured progression enabled efficient iteration and informed design refinement

grounded in observed system behavior rather than assumptions. At its current stage, Kılıç represents a mature and competition-ready autonomous surface vehicle, with stable baseline autonomy and well-understood performance boundaries.

Ongoing efforts focus on final sim-to-real parameter refinement and repeatability testing to further increase confidence ahead of RoboBoat 2026. The engineering practices and lessons documented in this report establish a strong foundation not only for competition performance, but also for future research-driven development and platform evolution.

References

[1] Canonical Ltd. (2022). Ubuntu 22.04 LTS (Jammy Jellyfish). Retrieved from https://releases.ubuntu.com/22.04/

[2] Macenski, S., et al. (2022). Robot Operating System 2: Design, architecture, and uses in the wild. Science Robotics, 7(66).

[3] CubePilot. (2023). The Cube Orange Standard Set (ADS-B Carrier Board). Retrieved from https://docs.cubepilot.org/user-guides/carrier-boards/ads-b-carrier-board

[4] Stereolabs. (2024). ZED X Industrial Stereo Camera Datasheet. Retrieved from https://www.stereolabs.com/zed-x

[5] Unitree Robotics. (2023). Unitree L2 LIDAR Datasheet. Retrieved from https://www.unitree.com/L2

[6] Faconti, D. (2020). BehaviorTree.CPP: A C++ framework for Behavior Trees. Retrieved from https://www.behaviortree.dev/

[7] Bingham, B., et al. (2019). Toward Maritime Robotic Simulation in Gazebo. In MTS/IEEE OCEANS Conference.

[8] Open Robotics. (2023). Virtual RobotX (VRX) Documentation. Retrieved from https://github.com/osrf/vrx

[9] Xi, Z., et al. (2023). The Rise and Potential of Large Language Model Based Agents: A Survey. arXiv preprint arXiv:2309.07864.

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

[11] Macenski, S., et al. (2020). Smac Planner: Hybrid-A and State Lattice Path Planners for ROS 2*. Retrieved from https://docs.nav2.org/configuration/packages/smac /configuring-smac-hybrid.html

[12] Williams, G., Aldrich, A., & Theodorou, E. (2017). Model Predictive Path Integral Control: From Theory to Parallel Computation. Journal of Guidance, Control, and Dynamics.

[13] Faconti, D. (2023). Groot2: The advanced Behavior Tree Editor. Retrieved from https://www.behaviortree.dev/groot/

[14] Jocher, G., et al. (2024). Ultralytics YOLO. Retrieved from https://github.com/ultralytics/ultralytics

APPENDIX

Table 1: Components Lists

	Vendor	Model / Type	Specs	Custom/Purchased	Cost	Year of Purchase
ASV Hull Form/Platform	ZeroneTech Kılıç	V3	Fiberglass	Custom	550$	2025
Waterproof Connectors	Generic	Waterproof Connector / Cable Gland	IP68	Purchased	N/A	2025
Propulsion	Apisqueen	X8	"24V BLDC 9KgF Thruster"*2	Purchased	1200$	2026
Propulsion (Backup)	DEGZ	Utras	"24V BLDC 7.5 KgF Thruster"*2	Purchased	860$	2024
Power System	Zeee	Li-Po	"6S 10000mAh"*2	Purchased	300$	2026
Motor Controls	Apisqueen	24V 100A Feather ESC	Bidirectional ESC	Purchased	360$	2026
CPU	NVIDIA	Jetson Orin NX	8Gb Developer Kit	Purchased	950$	2024
Here 4 GPS	CubePilot	Here 4	Multiband RTK GNSS	Purchased	460$	2025
Here 4 Base	CubePilot	Here 4 Base	Here4 Base F9P RTK Module	Purcased	250$	2026
Teleoperation	Holybro	Telemetry Radio V3	433MHz 500mW	Purchased	285$	2024
Compass	Pixhawk Orange Cube	Built-in	MEMS Digital Compass	Purchased	510$	2025
Inertial Measurement Unit (IMU)	Pixhawk Orange Cube	Built-in	3-axis Gyroscope + 3-axis Accelerometer	Purchased	N/A	2025
Camera	Stereolabs	Zed X	Stereo Camera	Purchased	1200$	2026
Controller	iFlight	Commando 8	915 MHz 1000mW	Purcased	230$	2024

Table 2: System-Level Validation Against Mission Requirements

Test Category	Metric / Requirement	Targeted Value	Measured / Simulated Result	Status
Propulsion	Top Speed (Calm Water)	> 1.0 m/s	**1.5 m/s** (Field Test)	Pass
Endurance	Battery Runtime (Nominal)	> 20 min	**~45 min** (Logged)	Pass
Simulation	Nav. Channel Success Rate	> 90%	**95%** (VRX/Gazebo)	Pass
Perception	Max Detection Range (Buoy)	> 10 m	**15 m** (ZED X + Lidar)	Pass
Acoustics	Horn Freq. Classification	> 90%	**98%** (Bench Test)	Pass

This table presents the verification of core mission requirements through simulation, bench tests, and field trials. Measured results consistently exceeded minimum target values, demonstrating that the system is fully compliant with performance, perception, endurance, and autonomy requirements.

1. Trade-Off Analysis

Throughout the system design process, key architectural decisions were evaluated by comparing alternative approaches under RoboBoat competition constraints, including outdoor lighting conditions, dynamic marine environments, safety requirements, and system reliability. Each decision was validated through research, simulation results, and prior field experience.

1.1. Stereo Camera vs. Depth Camera

Alternative: Depth Camera

Decision: Stereo Camera (ZED X)

Rationale: Depth cameras are highly sensitive to direct sunlight, which significantly degrades depth accuracy in outdoor marine environments. Based on prior field testing experience, stereo vision provides more robust depth estimation under varying illumination conditions, making it better suited for RoboBoat competition settings.

1.2. 3D LIDAR vs. 2D LIDAR

Alternative: 2D LIDAR (YDLIDAR G2)

Decision: 3D LIDAR (Unitree L2)

Rationale: Two-dimensional LIDAR provides limited environmental awareness in dynamic marine scenarios, where object heights and wave effects influence perception. A 3D LIDAR enables full spatial awareness, improving obstacle detection reliability and navigation safety during autonomous operation.

1.3. Rolling Window SLAM vs. Global Mapping SLAM

Alternative: Full Global SLAM

Decision: Rolling Window (Local) SLAM

Rationale: The RoboBoat environment lacks stable landmarks for consistent loop closure, and water-induced drift reduces the effectiveness of global mapping approaches. A rolling window strategy allows real-time obstacle awareness and adaptive navigation without reliance on long-term map consistency, improving robustness during competition runs.

1.4. Trimaran vs. Catamaran Hull Design

Alternative: Catamaran

Decision: Trimaran

Rationale: The trimaran configuration offers increased lateral stability and improved resistance to wave-induced roll motions. This stability is particularly beneficial during low-speed maneuvers, perception-driven tasks, and precise navigation required in RoboBoat missions.

1.5. Behavior Tree vs. Learning-Based Decision Making

Alternative: Learning-based or LLM-driven decision systems

Decision: Deterministic Behavior Trees with Cognitive Support

Rationale: While learning-based models offer high-level reasoning capabilities, they may produce unpredictable outputs under unseen conditions. For safety-critical autonomous operation, deterministic Behavior Trees provide transparent logic, predictable behavior, and ease of debugging.

A lightweight LLM module is employed only as a diagnostic and recovery support tool, assisting in system state interpretation and suggesting predefined recovery actions without directly commanding vehicle motion.

This trade-off driven design approach prioritizes predictability, safety, and competition reliability over experimental complexity, ensuring stable performance throughout RoboBoat 2026 missions.

2. System Overview

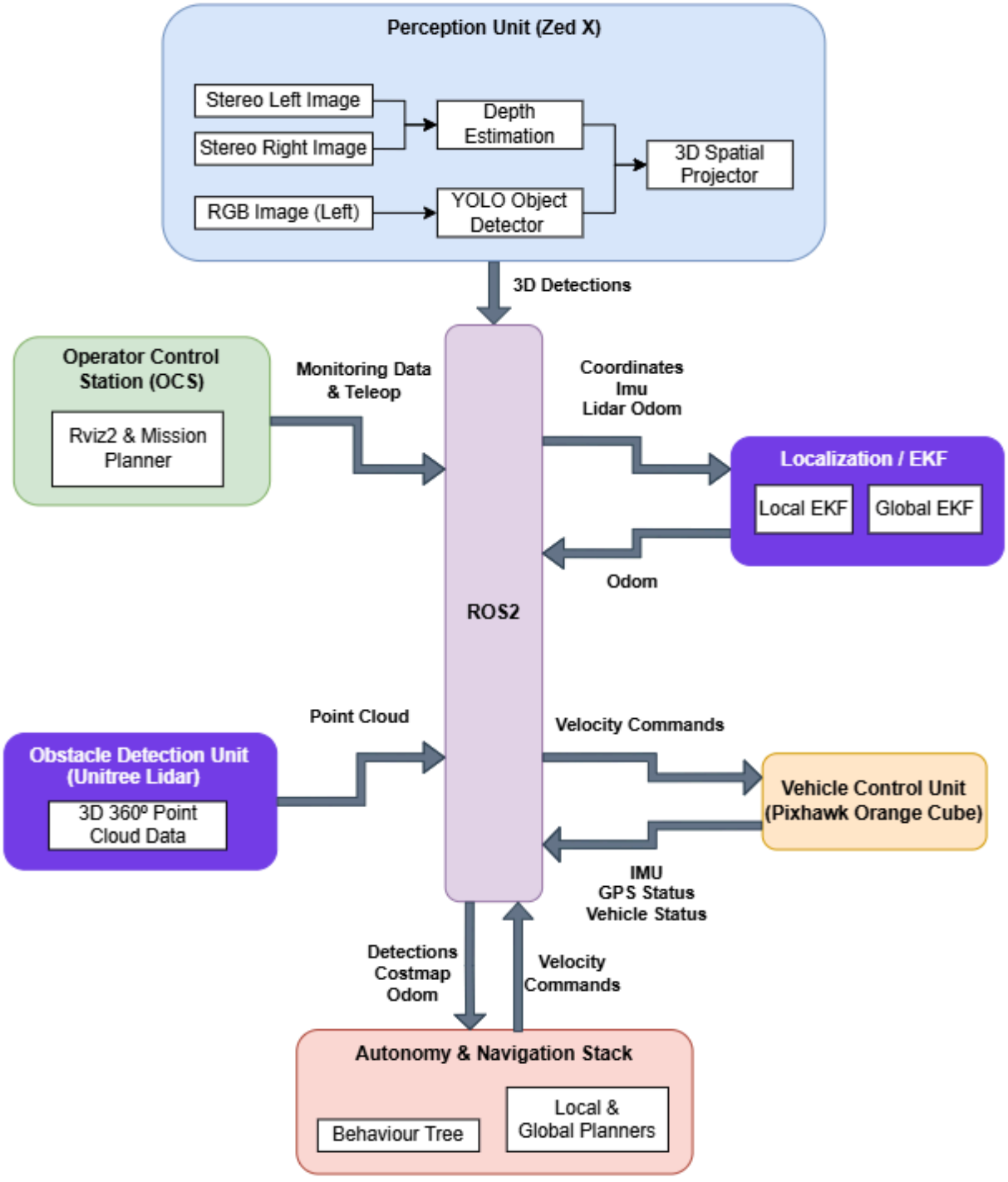


Figure 1: System Architecture

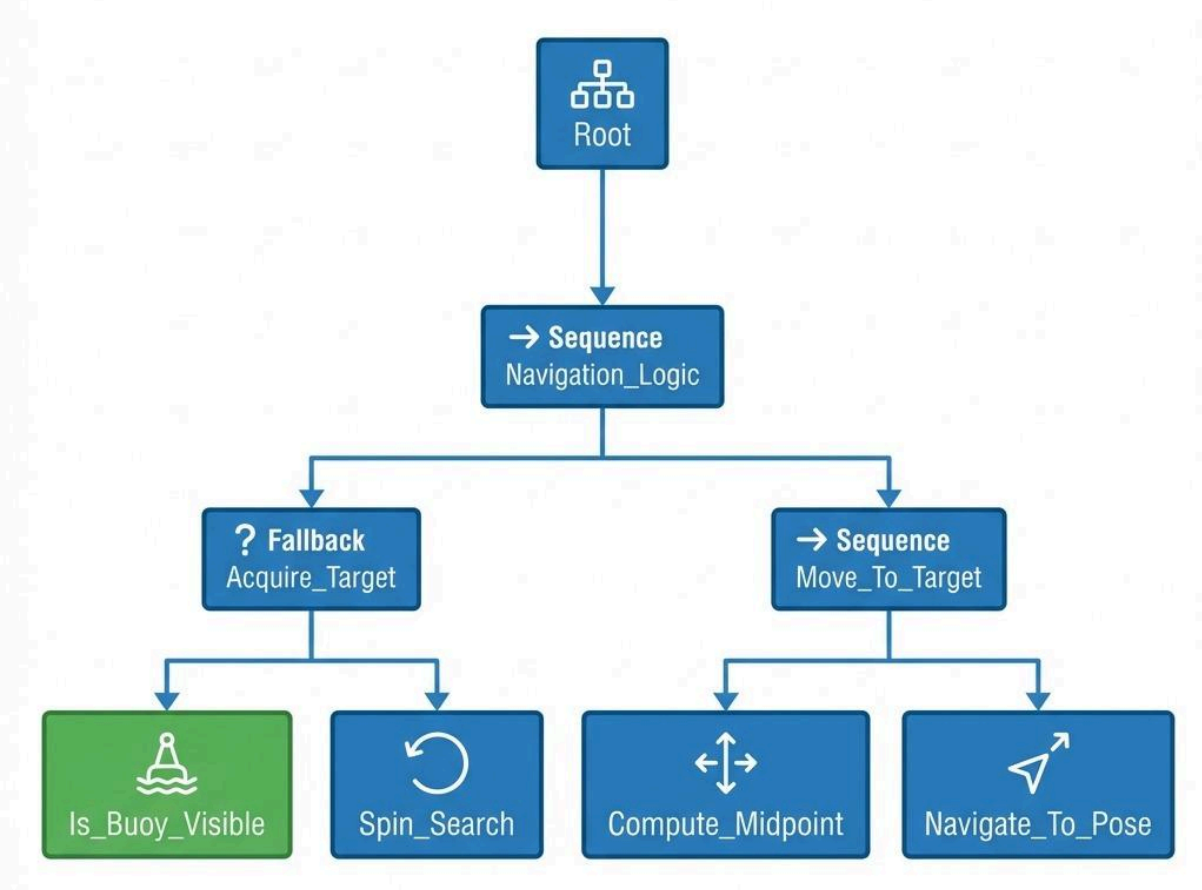


Figure 2: Sample schematic representation of the Behavior Tree decision mechanism designed on Groot.

This system architecture emphasizes modularity, fault isolation, and competition robustness, allowing the ASV to adapt to varying mission requirements while maintaining safe and reliable autonomous operation during RoboBoat 2026.

3. Vehicle Design

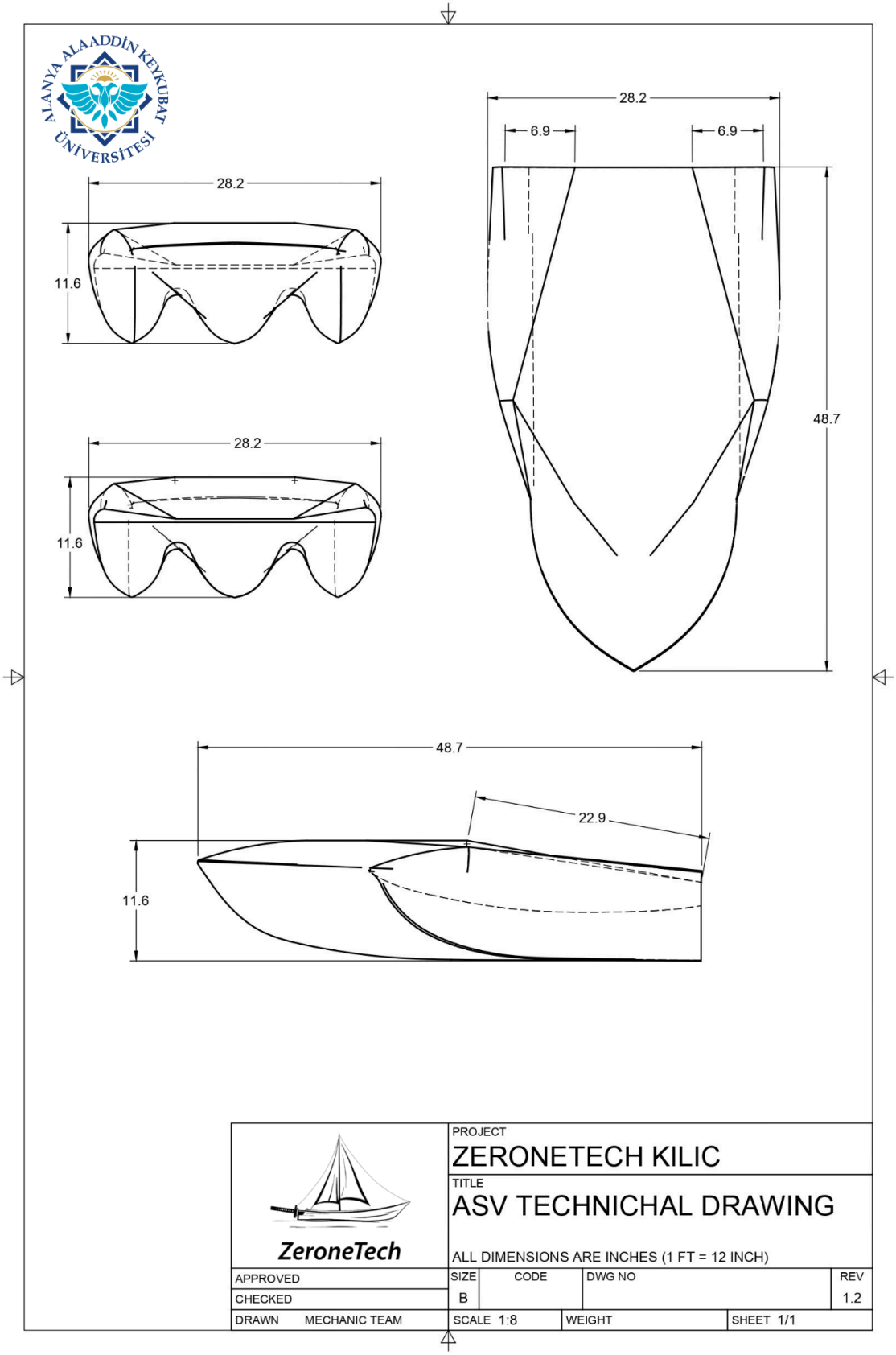


Figure 3: Technical hull drawing of the ZeroneTech Kılıç autonomous surface vehicle (ASV)

3.1. Mechanical & Propulsion Design

The mechanical design of the Kılıç Autonomous Surface Vehicle was developed with a primary focus on stability, maneuverability, and modularity, in direct alignment with the operational demands of RoboBoat competition tasks. The platform adopts a trimaran hull configuration, consisting of a central main hull supported by two lateral outriggers.

The hull dimensions are as follows: 48.7 inches in length, 28.2 inches in width, and 11.6 inches in height. This trimaran architecture was selected due to its superior lateral stability compared to mono-hull and catamaran alternatives. The increased stability is particularly advantageous during low-speed autonomous maneuvers, perception-driven navigation tasks, and docking-related operations, where excessive roll or yaw oscillations can degrade control accuracy. The distributed buoyancy provided by the side hulls also contributes to predictable hydrodynamic behavior under wave disturbances commonly encountered in outdoor competition environments.

Table 3: ApisQueen X8 Thruster Performance (Single Motor – 24V)

PWM Signal (µs)	Throttle	Thrust	Current	Power Usage
1500	0%	0.00 kgf	0.25 A	6.0 W
1550	10%	0.95 kgf	4.50 A	108.0 W
1600	30%	2.80 kgf	14.50 A	348.0 W
1700	60%	6.10 kgf	28.00 A	672.0 W
1900	100%	9.20 kgf	45.00 A	1080.0 W

This table presents the thrust, current, and power output of a single ApisQueen X8 thruster as a function of the PWM control signal. While low throttle levels provide relatively high energy efficiency, current and power consumption increase nonlinearly at higher throttle settings. These measurements were used to determine efficient and safe operating thrust ranges during mission execution.

Propulsion is achieved using a dual-motor differential thrust configuration, enabling yaw control through asymmetric thrust rather than mechanical steering components. This approach reduces mechanical complexity, improves reliability, and simplifies waterproofing requirements. Differential thrust control is well-suited for precise heading adjustments required during gate traversal, obstacle avoidance, and speed-regulated navigation tasks.

Overall, the mechanical and propulsion design of Kılıç emphasizes robustness, maintainability, and control compatibility, enabling reliable autonomous operation while allowing flexible adaptation throughout the development and competition phases.

3.2. Electrical & Power System

The electrical power system of the Kılıç Autonomous Surface Vehicle is designed to ensure reliable energy delivery, voltage stability, and operational safety throughout all mission phases. Power is supplied by two 6S Li-Po batteries (10,000 mAh, 120C), selected to provide sufficient current capacity for propulsion, onboard computing, and sensing systems while maintaining redundancy during extended operation.

Both batteries are directly connected to a centralized Power Distribution Board (PDB), which serves as the core energy management unit of the vehicle. The PDB regulates and distributes power across multiple voltage levels, including 5V, 9V, 12V, and 24V, to meet the specific requirements of sensors, control electronics, onboard computers, and motor controllers. This centralized distribution approach reduces wiring complexity, minimizes voltage drop, and improves system maintainability.

Table 4: Power Budget (ApisQueen X8 High-Power Configuration)

Component	Qty	Voltage	Avg. Current (A)	Peak Current (A)	Avg. Power (W)	Peak Power (W)
NVIDIA Jetson Orin NX	1	12V	1.67	2.08	20.0	25.0
ZED X Stereo Camera	1	12V	0.30	0.50	3.6	6.0
Unitree L2 LiDAR	1	12V	1.00	1.50	12.0	18.0
Pixhawk Cube Orange	1	5V	0.50	0.80	2.5	4.0
ApisQueen X8 Thruster (X2)	2	24V	5.00	45.00	240.0	2160.0
Network Switch & Peripherals	1	12V	0.50	0.80	6.0	10.0
TOTAL	-	-	-	-	**284.1 W**	**2223.0 W**

This table summarizes the average and peak power consumption of all major system components. The propulsion system is identified as the dominant power consumer, especially during high-thrust maneuvers.

To meet competition safety requirements, a physical Emergency Stop (E-Stop) system is implemented between the batteries and the PDB. When activated, the E-Stop mechanically disconnects the battery supply, immediately cutting all power to propulsion and electronic subsystems. This design ensures that, in emergency situations, the vehicle can be rendered completely inert without reliance on software-level intervention.

In addition to the physical E-Stop, the system incorporates layered fail-safe mechanisms at the controller level. In the event of communication loss, abnormal sensor readings, or manual override commands, propulsion outputs are disabled, and the vehicle transitions to a safe state. This multi-layered safety approach ensures compliance with RoboBoat regulations while protecting both the platform and its surroundings.

The electrical and power system architecture prioritizes safety, reliability, and modularity, supporting stable autonomous operation and rapid troubleshooting during testing and competition deployment.

Table 5: Power System Load & Thermal Endurance (X8 High-Current)

Operating Mode	Total Power	Voltage (V_batt)	PDB Temp	System Status
Idle	190 W	24.8 V	42°C	Stable
Cruise (30%)	750 W	23.0 V	55°C	Nominal
Maneuver (60%)	1400 W	21.5 V	68°C	High Load
Max Thrust (100%)	2400 W	19.8 V	85°C	Critical (<10s)

This table illustrates power system loading and thermal behavior under different operating modes. The system remains stable during idle and cruise conditions, while maximum thrust operation is suitable only for short-duration use. Based on these results, software-level power and thermal limits were implemented to ensure safe operation.

4. Sensors, Perception & Software

The perception system of Kılıç is designed around a multi-modal sensor fusion architecture to achieve robust and reliable environmental awareness under dynamic maritime conditions. By combining vision-based semantic perception with geometric sensing from depth cameras and LiDAR, the system mitigates the limitations of individual sensors and ensures consistent performance against lighting variations, reflections, and water surface artifacts. Vision-based perception provides semantic understanding of the environment. For object detection, the YOLOv11 architecture, one of the most widely adopted models in industry due to its favorable speed–accuracy trade-off, is employed. The model is trained specifically for the RoboBoat competition environment to detect mission-critical objects such as red, green, black, and yellow buoys, as well as small yellow and navy-colored marked boats.

The training dataset is constructed using images collected from RoboBoat-like environments and labeled on the Roboflow platform. Both real-world and synthetically generated images are included to improve generalization. Data preprocessing techniques such as resizing, normalization, and augmentation are applied to enhance robustness against environmental variability. During training, an early stopping strategy is used to prevent overfitting. The trained model achieves an overall accuracy of 89%, with plans to further improve this performance through iterative dataset expansion and retraining. Each detected class is assigned a confidence score, enabling downstream filtering and reliability assessment.

For real-time deployment, the trained YOLOv11 model is converted into a TensorRT engine and executed on the NVIDIA Jetson Orin NX using FP16 precision, achieving inference speeds exceeding 30 FPS. This optimization ensures that perception latency remains sufficiently low for closed-loop autonomous navigation.

While vision provides semantic classification, geometric understanding is achieved through depth sensing and LiDAR based validation.

The 2D bounding boxes produced by YOLO are projected into three-dimensional space using depth information obtained from the ZED stereo camera. If depth uncertainty is detected-commonly caused by water reflections or low-texture regions the system cross-validates the measurement using data from the Unitree LiDAR.

This dual-verification (cross-validation) mechanism significantly reduces false positives and improves spatial consistency by confirming object presence and distance through independent sensing modalities. LiDAR point cloud data further contributes to accurate obstacle geometry extraction and is used to populate local costmaps for navigation.

The final fused perception output is delivered to the navigation stack as a structured environmental representation, combining semantic labels, confidence scores, and precise 3D localization. By separating semantic classification from geometric validation, the perception architecture maintains both high accuracy and operational reliability, enabling stable obstacle avoidance and path planning throughout autonomous mission execution.

4.1. Localization, Navigation & Autonomy Software

Accurate localization and reliable navigation are fundamental to the autonomous operation of Kılıç. To achieve this, the system employs a multi-layer state estimation and navigation architecture designed to handle drift, sensor uncertainty, and environmental disturbances.

Vehicle state estimation is performed using a dual-layer Extended Kalman Filter (EKF) structure. The local estimation layer fuses inertial measurements with LIDAR-based odometry with Inertial Measurement Unit (IMU) to provide smooth and responsive motion estimates in the vehicle reference frame. To ensure long-term global consistency, a secondary estimation layer integrates RTK-enabled GNSS data, correcting accumulated drift while maintaining stability against transient positioning errors.

Navigation is executed through a combination of global path planning and local trajectory optimization. Global planners generate feasible routes that respect the vehicle’s non-holonomic motion constraints, while local controllers continuously adapt velocity commands in response to real-time perception updates and environmental disturbances such as wind and wave effects.

Mission-level autonomy is governed by a deterministic Behavior Tree framework, enabling transparent decision-making and modular task execution. Each competition task is implemented as an independent behavior module, allowing flexible task sequencing and graceful degradation in the event of partial subsystem failures. This structure ensures predictable system behavior while maintaining adaptability across diverse mission scenarios.

5. Operator Control Station (OCS)

5.1. Core Functionality

The Operator Control Station (OCS) serves as the primary interface between the Kılıç Autonomous Surface Vehicle and the human operator, providing real-time situational awareness and system oversight throughout all mission phases. While the vehicle operates fully autonomously, the OCS ensures the operator remains informed and able to intervene when necessary.

The OCS displays critical telemetry, including vehicle orientation, linear and angular velocity, position estimates (global and local), and system health indicators. Environmental representations such as local costmaps, detected obstacles, and navigation paths allow operators to monitor perception and decision-making in real time. High-level autonomy status including current mission phase, active behavior tree nodes, and task execution states is also visualized, enabling operators to understand why the vehicle performs specific actions, enhancing trust and operational confidence during competition runs.

5.2. Safety & Manual Override

Operational safety is a core requirement of Kılıç, and the OCS ensures safe operation through a manual override mechanism implemented on a dedicated RC transmitter. Operators can switch instantly between autonomous and manual modes via a hardware-level switch. In case of communication loss or manual mode engagement, autonomous velocity commands are immediately suppressed, and full control is returned to the operator.

Telemetry integrity is continuously monitored. If control signals or telemetry are interrupted beyond a defined threshold, the vehicle enters a fail-safe state by reducing propulsion and maintaining stable orientation. This layered safety approach—combining real-time telemetry, physical manual override, and fail-safe mechanisms—ensures that autonomous operation remains continuously supervised while enabling rapid, safe intervention throughout RoboBoat missions.

6. Simulation & Analysis

6.1. Simulation-Based Validation

Due to the high cost, logistical complexity, and safety risks associated with early-stage on-water testing, the development process of Kılıç follows a Simulation-In-The-Loop (SITL) validation strategy. This approach enables systematic verification of autonomy, perception, and control algorithms before deployment on physical hardware.

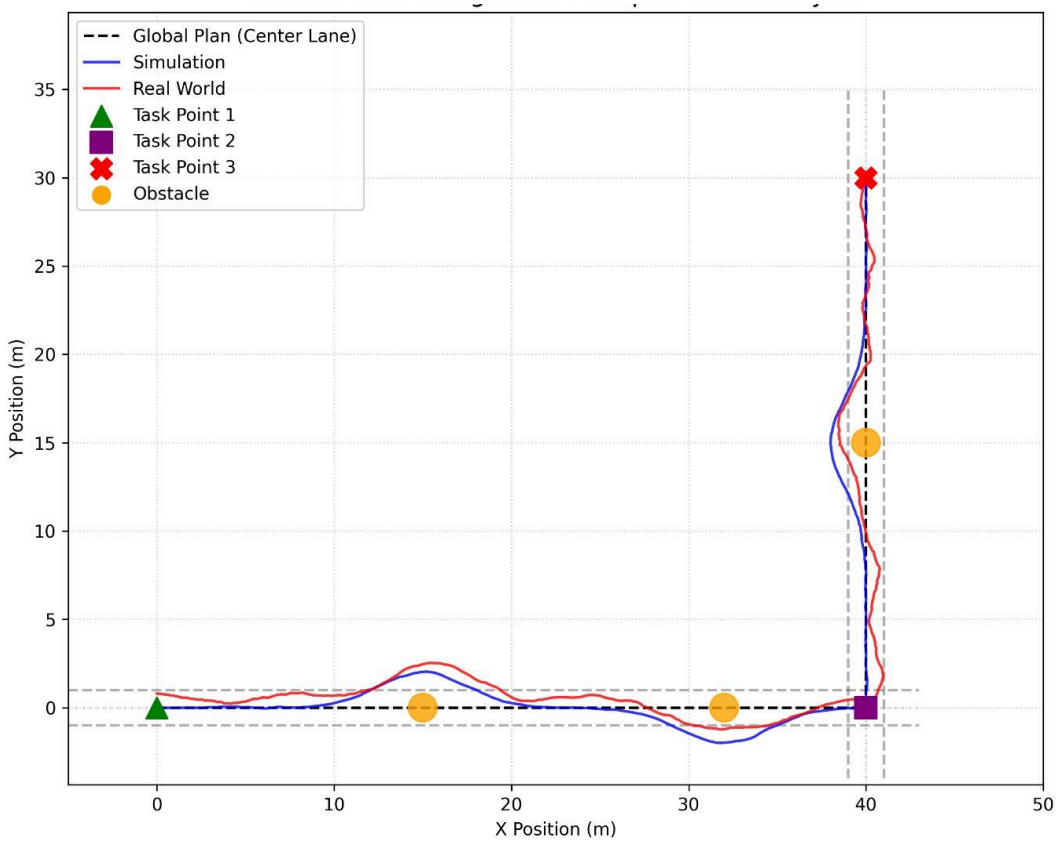


Figure 4: Simulation vs. Real-World Autonomous Navigation Comparison

This figure compares the reference trajectory with paths obtained from simulation and real-world field tests. Although minor deviations are observed in real-world execution, overall path tracking closely matches simulation results. This confirms the practical deployability of the proposed planning and control algorithms.

Navigation and mission-planning algorithms are validated using standardized test scenarios within the VRX environment, ensuring consistent benchmarking across development iterations. Through repeated simulation cycles, the majority of autonomy logic and decision-making behavior is stabilized prior to physical integration, significantly reducing risk during subsequent field testing.

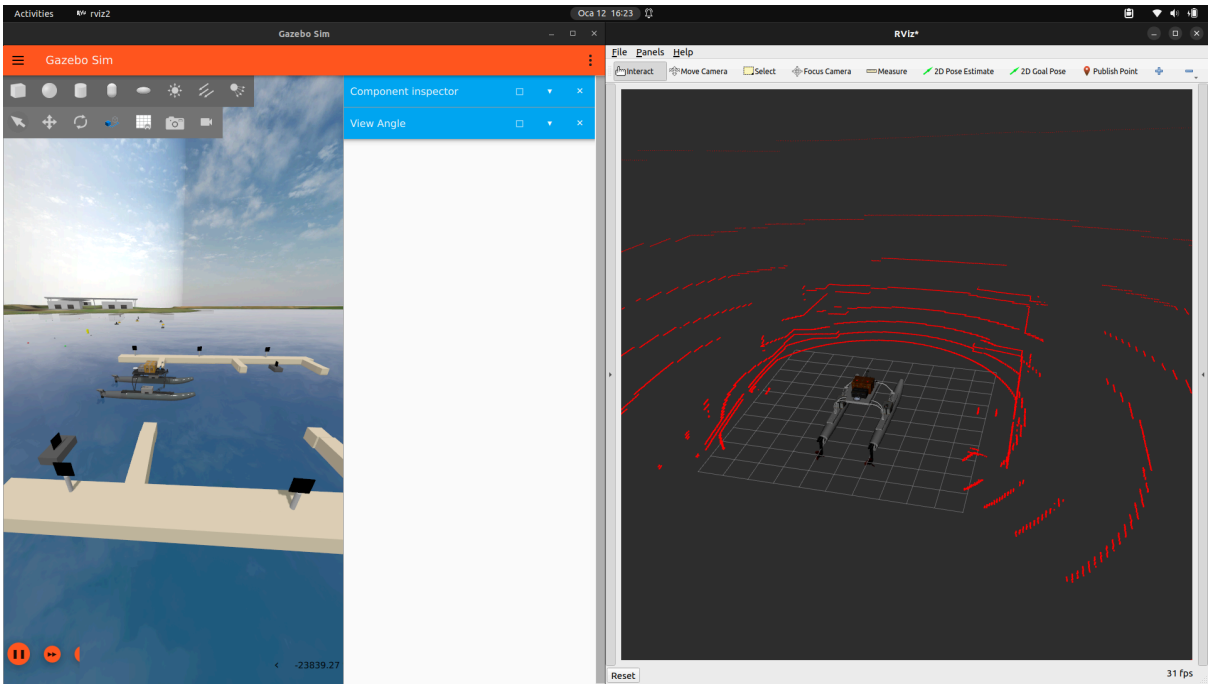


Figure 5: Developed SITL test infrastructure.

Physical course and dynamic environment simulation on Gazebo Sim in the left panel; LiDAR Point Cloud and the robot's current position visualized simultaneously in the RViz2 interface in the right panel.

The simulation environment is built upon the Virtual RobotX (VRX) framework integrated with the Gazebo Garden physics engine. This platform is specifically designed for maritime robotics and allows realistic modeling of hydrodynamic effects such as wave interaction, wind-induced drift, and vehicle inertia. By operating within a competition-relevant simulation environment, the team ensures that algorithmic performance translates meaningfully to real-world RoboBoat conditions.

To evaluate robustness, simulated sensor streams including stereo vision and LIDAR are intentionally degraded using noise models that approximate real-world measurement uncertainty. This stress-testing approach exposes perception and localization algorithms to adverse conditions, enabling early detection of failure modes and facilitating targeted improvements.

Table 6: Simulation vs. Real & Perception System Accuracy

Performance Metric	Gazebo Simulation	Real World Data	Deviation Analysis
Buoy Detection Range	20.0 m	16.5 m	-17.5% (Glare/Haze)
Lidar Distance Error	±0.01 m	±0.04 m	Surface Reflectivity
False Positive Rate	0.0%	3.2%	Water Reflection
Object Classification	99.5%	94.2%	Lighting Variation
SLAM Loop Closure	Perfect	±0.15 m	Minor Drift

This table compares perception system performance between simulation and real-world operation. Reduced detection range and classification accuracy in real conditions are primarily caused by surface glare, lighting variation, and water reflections. Nevertheless, the perception system maintained sufficient accuracy to meet mission-level requirements.

6.2. Subsystem & Integration Testing

Beyond simulation, the validation process is structured around incremental subsystem and integration testing, ensuring that each component performs reliably before full system deployment. Mechanical, electrical, and software subsystems are initially tested in isolation to verify functional correctness and performance boundaries.

Propulsion and power systems are evaluated under controlled conditions to validate thrust response, current consumption, and thermal behavior. Sensor subsystems are tested independently to confirm data integrity, synchronization, and resistance to environmental disturbances. Software modules including perception, localization, and navigation are validated using recorded sensor data and simulated inputs prior to live integration.

Once individual subsystems meet performance criteria, staged integration tests are conducted. These tests focus on inter-module communication, timing consistency, and fault isolation within the ROS 2 framework. By progressively increasing system complexity, integration issues are identified early and resolved without cascading failures.

This layered testing methodology ensures that system-level behavior emerges from well-understood and validated components, rather than ad hoc integration during competition preparation.

Table 7: Simulation vs. Real & Navigation & Control Performance

Metric	Simulation Value	Field Test Value	Variance / Note
Cross-Track Error (Straight)	0.05 m	0.18 m	+0.13 m (Wind/Current)
Heading Error (RMS)	0.5°	2.1°	+1.6° (Compass Noise)
Station Keeping Drift	±0.15 m	±0.45 m	Acceptable (<0.5m)
Max Autonomous Speed	2.5 m/s	2.2 m/s	-12% (Safety Limit)
Docking Accuracy	±0.05 m	±0.12 m	Within Tolerance

This table compares navigation and control performance metrics obtained in simulation with real-world field test results. Increased errors in real conditions are mainly attributed to wind, current, and sensor noise. Despite these effects, all measured values remained within acceptable limits, indicating that the simulation model represents real system behavior with sufficient accuracy.

6.3. Field Testing & Competition Readiness

Following simulation and staged integration testing, controlled on-water field tests are conducted to validate system behavior under real environmental conditions. These tests focus on evaluating sensor performance, propulsion response, and autonomous navigation stability in the presence of wind, waves, and lighting variability.

Initial field trials prioritize low-speed autonomous navigation and manual override verification to ensure safety and controllability. Subsequent tests incrementally introduce perception-based navigation and obstacle avoidance behaviors, allowing performance tuning under progressively complex scenarios.

Insights gained from field testing are used to refine control parameters, perception thresholds, and mission logic, bridging the gap between simulated and real-world operation. This structured progression from simulation to field deployment ensures that Kılıç approaches RoboBoat 2026 with a high level of operational readiness and confidence.

7. Challenges & Lessons Learned

Throughout the development of the Kılıç Autonomous Surface Vehicle, the team faced several technical and organizational challenges that shaped the final system architecture and development methodology, providing valuable lessons for autonomous system design under real-world conditions.

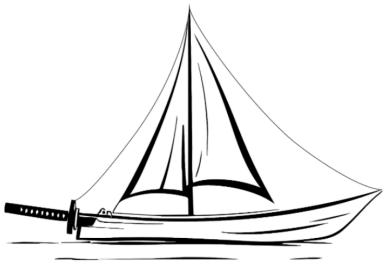
One primary challenge was achieving robust perception in dynamic marine environments. Variations in lighting, water surface reflections, and glare caused inconsistencies in vision-based detection. To address this, the team implemented multi-modal sensor fusion, integrating a ZED X stereo camera with a LIDAR module for cross-validation of detected obstacles and navigation features. This approach significantly improved detection reliability, highlighting the importance of redundant sensing modalities.

Another key challenge was maintaining stable navigation and control under environmental disturbances such as wind and wave-induced motion. Initial tests with conventional PID control revealed oscillations during low-speed maneuvers. The team iteratively tuned control parameters and validated performance in Gazebo simulations before on-water deployment, emphasizing the value of sim-to-real testing and early consideration of environmental dynamics in design.

System integration also posed challenges, particularly in synchronizing data flow between perception, localization, and navigation modules. Incremental integration and staged testing were crucial for identifying timing and communication issues, reinforcing the benefits of modular system architecture and disciplined, test-driven development.

Finally, the project offered lessons in team coordination and development workflow. Clear subsystem ownership, version-controlled code management, and scheduled integration tests were critical for reducing delays and ensuring consistent progress.

These challenges underscored the importance of redundancy, conservative design choices, and structured validation. The lessons learned from Kılıç's development will directly inform future iterations of the platform and provide guidance for autonomous marine vehicle projects beyond RoboBoat 2026.



ZeroneTech