Technical Design Review (TDR)

Abstract

This Technical Design Review (TDR) presents the design, implementation, and testing of a custom Autonomous Surface Vehicle (ASV) developed for autonomous navigation tasks in a competitive environment. The vehicle design emphasizes reliability, modularity, and computational efficiency while managing system complexity. Competition strategy directly informed decisions in hull design, propulsion layout, sensor selection, control architecture, and autonomy software. The resulting system integrates a lightweight composite hull, multi-thruster propulsion, ROS2-based autonomy, and onboard perception to achieve robust and repeatable autonomous performance within a constrained development timeline.

Acknowledgements

The team acknowledges the guidance and support of faculty advisors, mentors, and peers who contributed technical feedback throughout the design process. Appreciation is extended to the open-source communities supporting ROS2, OpenCV, and Linux-based embedded systems, as well as hardware vendors whose components enabled rapid prototyping and system integration.

Competition Strategy

The competition strategy prioritizes reliable completion of mandatory autonomous navigation tasks, specifically start gate traversal, while selectively attempting additional tasks such as waypoint navigation, obstacle avoidance, and vision-based object detection. Given limited testing time, the team deliberately constrained system complexity to maximize robustness and repeatability.

Key strategic considerations included:

- Focusing on surface navigation tasks that can be completed using vision and 2D localization
- Selecting mature, well-documented sensors and software frameworks
- Excluding high-risk subsystems such as DVLs and hydrophones to reduce integration overhead

This approach ensures consistent baseline performance while preserving extensibility for future iterations.

Design Strategy

System Architecture

The ASV is organized into modular subsystems: hull and structure, propulsion, power distribution, computing and control, sensing, and autonomy software. Electrical and software interfaces are standardized, enabling rapid troubleshooting and incremental upgrades.

Hull and Mechanical Design

The vehicle uses a custom Bluewave 2.0 lightweight composite hull with modular mounting brackets. The hull geometry provides inherent stability and low hydrodynamic drag. Mass distribution was optimized to maintain positive buoyancy and a low center of gravity, improving stability during low-speed maneuvers and sensor operation.

Propulsion and Power

Propulsion is achieved using four BlueRobotics M080 underwater thrusters arranged in a differential configuration, enabling forward/reverse motion and yaw control through differential thrust. This configuration provides redundancy and precise maneuverability. Power is supplied by a 3S 2200mAh LiPo battery (11.1V, 25C), selected to support peak thrust demands while maintaining safe operating margins. IP67-rated connectors ensure electrical reliability in wet environments.

Computing and Control

A Raspberry Pi 5 serves as the primary onboard computer, offering sufficient processing capability for real-time perception, localization, and control while maintaining low power consumption. Low-level motor commands are generated via PWM-controlled ESCs. A ROS-based teleoperation interface using a game controller provides manual override and safety control during testing.

Sensors and Perception

The sensor suite supports navigation and perception tasks while minimizing system complexity. Orientation and heading are provided by an Adafruit BNO055 IMU and QMC5883L digital compass. Visual perception is enabled by a Raspberry Pi Camera Module 3, supporting real-time object detection. Localization is achieved using 2D SLAM via an RP LIDAR A1M8 fused with inertial data. Sensors requiring complex calibration or specialized environments were intentionally excluded.

Autonomy and Software

The autonomy stack is implemented using ROS2 and structured as a finite state machine (FSM). The FSM manages mission sequencing, navigation behaviors, obstacle avoidance, and failsafe logic. Vision processing uses OpenCV and a lightweight YOLOv5-based model optimized for embedded execution. PID controllers govern heading and velocity control, enabling stable trajectory tracking.

Testing Strategy

Testing followed a staged systems-engineering approach. Initial algorithm validation was performed in simulation using ROS tools. Component-level testing verified thruster performance, power delivery, and sensor accuracy. Subsystem testing integrated propulsion, sensing, and control in controlled environments. Full-system testing evaluated autonomous navigation and obstacle avoidance in pseudo-competition scenarios. Testing outcomes guided iterative improvements in controller tuning, sensor placement, and FSM logic.

References

[1] Open Robotics, "ROS2 Documentation."
[2] OpenCV, "Open Source Computer Vision Library."
[3] BlueRobotics, "M080 Thruster Technical Specifications."
[4] Raspberry Pi Foundation, "Raspberry Pi 5 Hardware Documentation."

Appendix (Optional): Test Plan and Results

The test plan defined objectives, test cases, schedules, and risk mitigation strategies. Results demonstrated stable autonomous navigation, accurate heading control, and reliable vision-based detection under expected operating conditions.