Final Report

Tryton - Subsea Remotely Operated Vehicle (ROV)

Brandon Hargitay, Halvor Hafnor, Jayanti Lahoti
University of California, San Diego
{bhargitay, hhafnor, jlahoti}@ucsd.edu

1. Abstract

Tryton is a modular, low-cost, 3D-printable underwater remotely operated vehicle (ROV) designed for real-time video feedback, sensor-assisted control, and stable manual operation. Built on the open-source CPS5 platform, Tryton integrates open hardware and software components including a Pixhawk flight controller, depth and inertial sensors, and a Raspberry Pi onboard computer. Over the course of the quarter, the system was fully assembled, sealed, and validated through successful pool and open-water testing. With its sensor-guided stabilization and robust communication pipeline over Ethernet, Tryton establishes a functional and extensible foundation for future autonomous behaviors such as object tracking or mapping, leveraging onboard compute already in place.

2. Introduction

Underwater exploration and monitoring have historically required expensive and specialized equipment, placing advanced oceanographic tools out of reach for educational institutions, small research labs, and hobbyists. In recent years, the emergence of open-source hardware and low-cost manufacturing methods such as 3D printing has made it feasible to build affordable remotely operated vehicles (ROVs). However, most low cost ROVs remain limited in capability: they typically support only basic video streaming and manual control, with little to no onboard sensing or autonomy. This severely restricts their usefulness for precision tasks like marine surveying, object inspection, or scientific sampling, where stable positioning and environmental awareness are essential.

Tryton addresses this gap by providing a modular, low cost, open-source ROV platform that integrates core sensing, stabilization, and control features. Built on top of the CPS5 underwater drone design, Tryton combines 3D printed mechanical components with open source flight control software (ArduPilot), real-time video feedback, and onboard sensing using a depth sensor and inertial measurement unit (IMU). Our goal was to develop a minimum viable product (MVP) that demonstrates stable underwater operation, reliable teleoperation, and sensor-based depth hold all within a compact and affordable architecture that could be easily replicated and extended by others.

Tryton's software architecture leverages a Pixhawk flight controller running ArduSub (an ArduPilot variant), paired with QGroundControl as a surface interface. Commands are sent through an Ethernet tether, and manual control is achieved through a USB gamepad connected to a topside laptop. The ROV also includes a Raspberry Pi that serves as a flexible platform for potential computer vision or autonomy modules, though advanced features such as object detection were deferred in favor of focusing on core functionality and stability. This architecture was chosen to future proof the system for autonomous upgrades while keeping the current implementation accessible.

The physical structure of the ROV was fabricated using 3D-printed components derived from the CPS5 design, housing all electronics in a sealed acrylic enclosure. The vehicle is neutrally buoyant and capable of maneuvering in six degrees of freedom using four thrusters. All mechanical, electrical, and software systems were developed and tested in parallel throughout the quarter, with field tests conducted to validate control performance, waterproofing integrity, and underwater stability.

The broader significance of Tryton lies in its proof of concept for a low cost ROV architecture that supports modular autonomy. By demonstrating a fully operational underwater platform with sensor guided control and real-time feedback, we lay the groundwork for future extensions such as onboard object tracking, mapping, or environmental monitoring. These capabilities could transform Tryton from a teleoperated device into a smart subsea agent capable of executing mission-level objectives.

3. Related Works

Commercial Open-Source ROVs

OpenROV broke ground in 2012 with an affordable, palm-sized kit featuring a BeagleBone Linux board, three brushless thrusters, LED equipped camera, and a 100 Mbit/s Ethernet tether for real-time video streaming and control [1]. Its fully open hardware and software drew thousands of DIY builders, though many reported challenges sourcing specialized batteries and ensuring reliable waterproofing under extended use. OpenROV's commercial offering, the Trident underwater drone, then evolved toward a more professional design, with a hydrodynamic torpedo frame, up to 70 m depth rating, and two-hour runtime [8]. Building on this, Blue Robotics released the BlueROV2 in 2016: a 6-thruster vectore-drive vehicle powered by the ArduSub autopilot firmware and QGroundControl interface, rated to 100 m (upgradeable to 300 m) [2]. Its aluminum-and-acrylic frame is designed for modular expansion users routinely add manipulators, sonar units, and environmental sensors making it a versatile tool for inspection, research, and education.

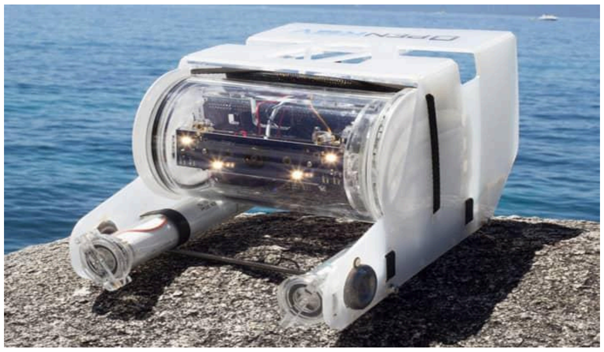


Figure 1: OpenROV Trident

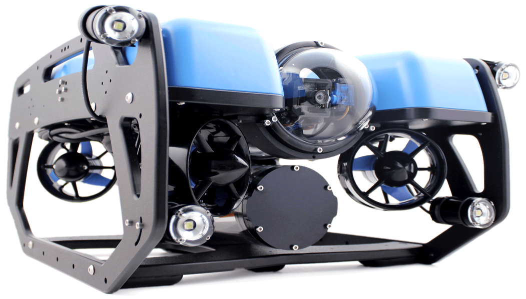


Figure 2: Blue Robotics Rov

Community & Educational Kits
Single maker projects like Aruna demonstrate that fully 3D-printed, sub-$500 ROVs can support real-time video and basic navigation, with design files and build instructions released under permissive open licenses to encourage remixing and improvement [3]. At the same time, SeaPerch kits have introduced thousands of middle and high school students to ROV design, using PVC-based frames, inexpensive bilge-pump thrusters, and simple wired controllers for under $200 [4]. Competitions like the Marine Advanced Technology Education (MATE) ROV Competition provide annual challenges for students to design modular, low cost vehicles, with open-source submissions and detailed technical reports freely available online [9]. Though rudimentary, these educational platforms power robotics contests and can be upgraded by students to carry cameras or custom microcontroller boards, illustrating how minimal cost can still foster deep learning and innovation.

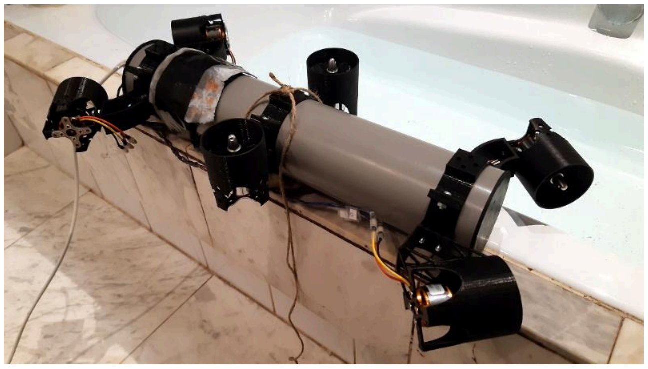


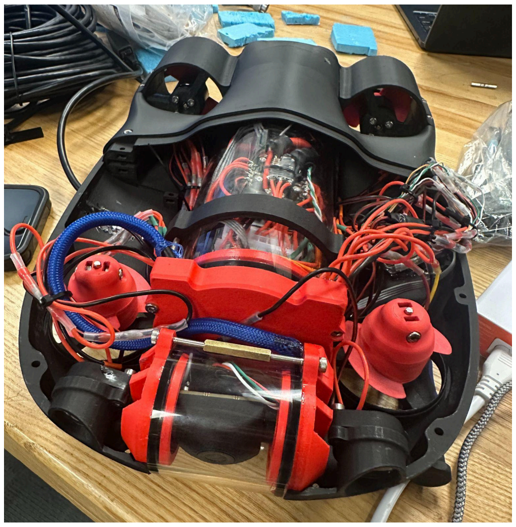
Figure 3: Aruna ROV

Academic Low-Cost & Modular Designs
Researchers have shown that a Raspberry Pi 3, six brushless thrusters, and open-source PID controllers can yield a 100 m-depth ROV streaming 42 fps video over Ethernet for under $1,000 complete with a Python-based GUI for topside control [5]. Projects like Modularis take modularity further, providing plug and play electronics bays and dual-mode ROS support (tethered ROS 1 topside, untethered ROS 2 onboard) to let scientists swap sensors and experiment with autonomy without rewiring [6]. Similarly, Aristizábal et al.'s Visor3 redesign replaced a legacy frame with open-source microcontrollers and a layered software stack, greatly simplifying integration of new payloads and rapid diagnostics [7]. The ros_rov GitHub project further demonstrates a ROS-based architecture for hobbyist ROVs, complete with plug-and-play sensor nodes, underwater communication protocols, and example autonomy scripts [10].

4. Technical Material

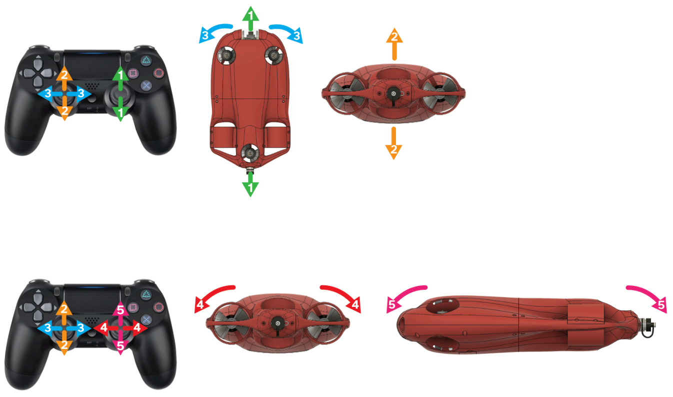
Hardware

Structural Frame and Enclosure
Tryton ROV uses a fully waterproof, 3D-printed frame and acrylic pressure hulls. The main body comprises two acrylic tubes one for the electronics and one for the camera sealed at each end with custom 3D-printed endcaps featuring O-ring gaskets. Cable penetrations are minimized; a single waterproof connector on the main enclosure houses the Ethernet tether feedthrough and is sealed with epoxy for reliability. The hull design is tuned so the vehicle is nearly neutrally buoyant, allowing it to hover when thrusters are idle.

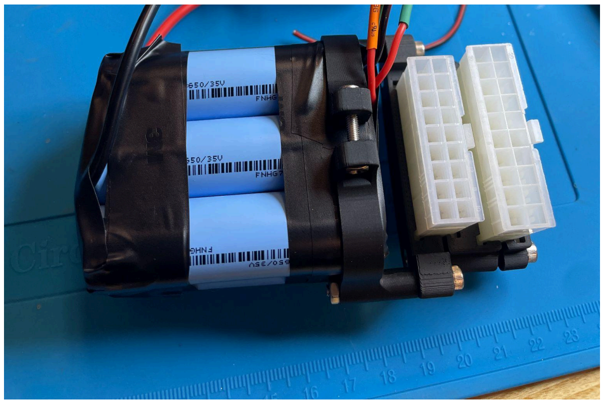

Inside of Tryton

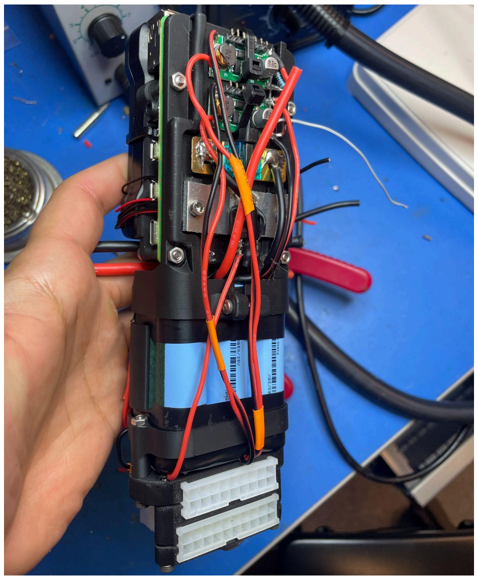
3D Printed Frame

Propulsion and Thrusters
Five brushless DC thrusters provide the vehicle's propulsion. Two horizontal thrusters mounted on opposite sides handle forward/reverse motion and yaw control, while three vertical thrusters (one front, two rear) manage depth control and pitch/roll stabilization. Each motor is potted in epoxy and fitted with ceramic bearings to resist corrosion. Custom 3D-printed propellers attach directly to the motor shafts. Thruster speed is controlled via individual electronic speed controllers (ESCs) mounted inside the electronics tube, which receive PWM signals from the flight controller.

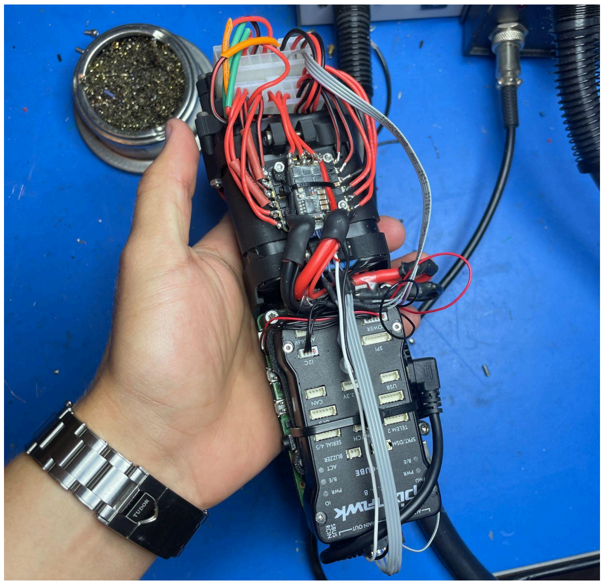

Five Degrees of freedom

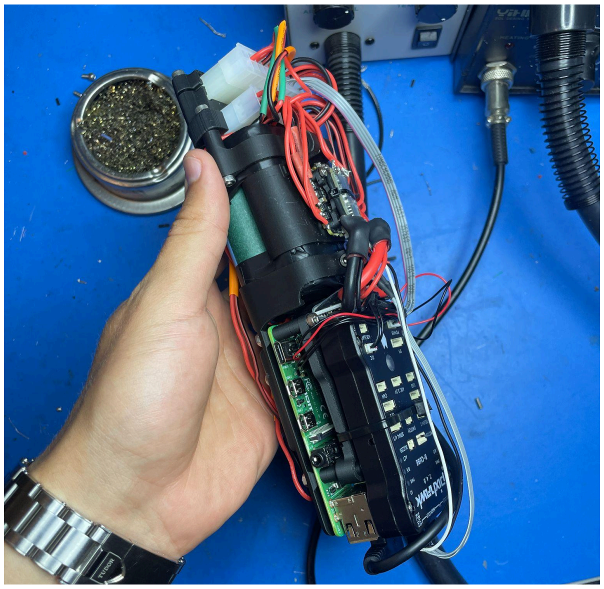
Power Supply and Wiring
A rechargeable lithium-ion battery pack (11.1 V nominal) powers the ROV for approximately one hour. The battery resides in the main hull, connected to a DC–DC converter that provides 5 V for low-voltage electronics. High-current wiring and connectors are marine-grade and sealed with epoxy or waterproof plugs. Inside the hull, wiring is neatly organized: the ESCs draw directly from the battery bus, and signal lines run to the Pixhawk controller. A power-monitoring circuit tracks battery voltage and current, and includes a MOSFET-based cutoff to protect against deep discharge.


11.1V Battery


Battery wired to electronics

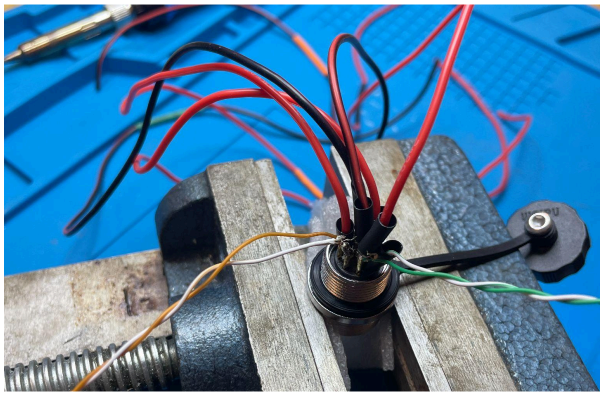
Onboard Electronics
The primary autopilot is a Pixhawk board running ArduSub firmware. It interfaces with an onboard IMU for orientation sensing and a depth sensor for real-time depth feedback. An external magnetometer provides heading information. A Raspberry Pi companion computer connects to the Pixhawk over MAVLink, handling high-level tasks such as video streaming and additional sensor polling. The Pi manages the HD camera, encoding the video feed for transmission and logging mission data. All electronics are mounted on a sled inside the pressure tube, designed to withstand pressures down to 100 m depth.





Tether and Topside Connection

Communication and command signals travel over a lightweight, neutrally buoyant Ethernet tether rated for up to 100 m. The tether connects to a top side computer running a mission control software. This setup separates power and data, as the ROV runs on its own battery, avoiding power loss over long cables and simplifying the tether design.


Soldering of ROV connector

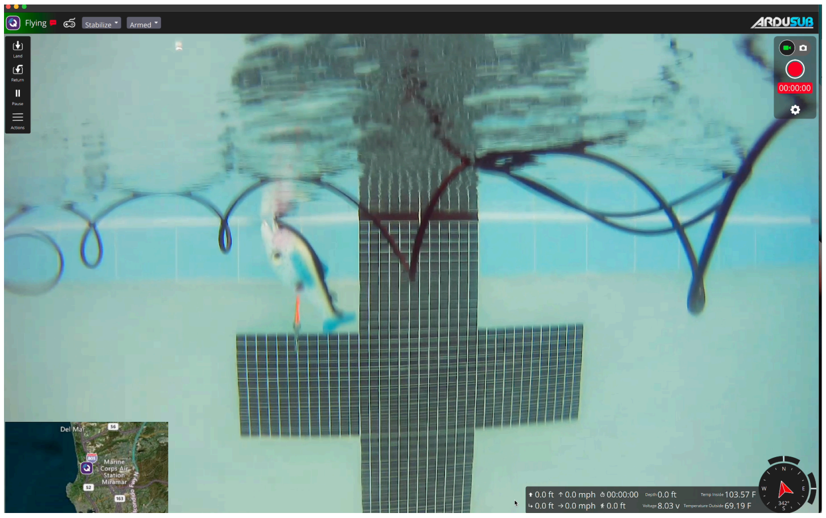
Ethernet Tether

Software Architecture

ArduSub Firmware
The Pixhawk runs ArduSub, an open-source autopilot firmware for underwater vehicles. ArduSub implements PID-based control loops for depth hold, auto-level, and heading hold modes. It reads IMU and depth sensor data at high rates and outputs PWM signals to the ESCs. Pilot inputs (from a joystick or gamepad) are mixed by ArduSub according to the vehicle's five-thruster configuration to achieve the desired motion.

Companion Computer Software
The Raspberry Pi runs a lightweight Linux environment with a MAVLink router that bridges Pixhawk telemetry to the surface over Ethernet. It also runs a video-streaming pipeline (e.g., GStreamer) to capture and encode HD camera output. The Pi can host additional scripts or services for logging and interfacing with extra sensors, but leaves real-time control to the Pixhawk.

Topside Control Interface
On the surface, the operator uses QGroundControl on a laptop or tablet. QGroundControl displays live video, telemetry dashboards (attitude, depth, battery status), and offers modes like depth hold and auto-level. Control inputs via joystick are translated into MAVLink commands and sent down the tether. The low-latency link enables responsive teleoperation and easy mission configuration, including parameter tuning and sensor calibration.



QGroundControl

5. Milestones

At the beginning of the quarter, we defined a series of technical deliverables to structure the Tryton ROV project. These milestones included mechanical assembly, electronics integration, control system configuration, and real-world testing. Our stretch goal included implementing computer vision-based object detection. Over the course of the project, we completed all core milestones and made design adjustments based on testing outcomes and time constraints.

Completed Milestones

D1 – Frame and Electronics Assembly

We successfully assembled the full Tryton frame using 3D-printed CPS5 components and enclosed the electronics in a sealed acrylic housing. All internal wiring, ESCs, sensors, and power components were integrated and validated through continuity and voltage testing.

During early electrical integration, we discovered multiple solder joints had become disconnected during handling. We used a multimeter extensively to trace signal paths and verify continuity, identifying and resoldering problem points across ESC, power, and signal wires. Additionally, the Pixhawk's main connector cable physically detached from the board three separate times over the quarter. Chris resoldered it once, his friend another time, and we did it ourselves the third — highlighting how fragile certain parts of the board are under even light strain.

D2 – ArduPilot + QGroundControl Setup

The Pixhawk was flashed with ArduSub, and joystick controls were mapped through QGroundControl. We confirmed motor actuation, telemetry streaming, and basic control loop functionality.

D3 – Depth and Position Hold Tuning

We tuned the depth sensor and IMU to allow for stable underwater control. The ROV was able to maintain depth and orientation with minimal drift during pool tests, achieving stable hover using onboard sensor feedback.

One major hurdle in achieving stable underwater hover was misconfigured motor directions. Several thrusters initially spun the wrong way, causing the ROV to pitch unpredictably and fail to stabilize. We resolved this by inverting motor channels in software and verifying orientation manually through a controlled test routine.

D4 – Real-Time Video Streaming

A Raspberry Pi onboard the ROV successfully transmitted live video over Ethernet to the surface station. Integration with QGroundControl allowed for simultaneous control and video monitoring.

D5 – Open Water Field Test

We conducted a successful open-water test, validating waterproofing, communication stability, and control performance under real-world conditions.

During our first open-water test, one of the vertical thrusters detached from its mount mid-mission. Fortunately, the affected area was fully waterproofed, and no internal components were exposed. We recovered the vehicle without damage and re-secured the motor with additional epoxy and mechanical reinforcement.

D6 – Computer Vision Integration

While we prepared the software environment for object detection using ROS 2 and MAVLink, we chose to defer implementation of computer vision in order to prioritize system stability, sensor tuning, and field testing. The Raspberry Pi remains onboard and configured to support future autonomous functionality.

Technical Challenges and Resolutions

During the project, we encountered several challenges:

Sensor Calibration Issues: Initial IMU readings showed drift, and motor directions required remapping. We resolved this by recalibrating the IMU under real conditions and creating a test routine to verify thruster orientation.

Voltage Drop Over Tether: Brownouts during high-throttle maneuvers were traced to voltage sag over the Ethernet tether. We mitigated this by adjusting power distribution and current limits.

Enclosure Sealing: Minor leakage during early submersion tests led us to revise the sealing method. We replaced the original gaskets with higher-quality O-rings and validated the seal with extended vacuum and dunk tests.

Despite these setbacks, we achieved full MVP functionality and a stable underwater platform ready for future autonomous upgrades.

6. Conclusion

The Tryton project set out to design and implement a functional, modular underwater ROV that combines open-source hardware, real-time feedback, and sensor-assisted stability all within a low-cost and accessible platform. Over the course of the quarter, we completed the mechanical assembly, integrated all major electrical subsystems, and developed a reliable software control stack using ArduPilot and QGroundControl. The system was validated through successful field testing, demonstrating stable manual operation, depth hold, and live video transmission.

Tryton's architecture emphasizes extensibility. By embedding a Raspberry Pi alongside the Pixhawk controller and establishing sensor feedback loops, the ROV is already equipped to support more advanced features such as onboard autonomy, computer vision, and navigation. These capabilities were deliberately scoped for future work, but the current implementation proves that the platform is technically sound and ready to support them.

This project not only delivers a working underwater ROV but also contributes a replicable framework for future development in low-cost subsea robotics. Whether for education, research, or hobbyist use, Tryton provides a clear path forward for affordable, intelligent underwater systems.

7. Acknowledgments

We used OpenAI's ChatGPT to assist with troubleshooting hardware and software integration issues throughout development, and to help structure and edit portions of this report. All design, implementation, and final decisions were made by the project team.

8. References

[1] OpenROV. "OpenROV 2.8 Kit." OpenROV, 2016. https://github.com/OpenROV
[2] Blue Robotics. "BlueROV2 Overview." Blue Robotics, 2016.
https://bluerobotics.com/store/rov/bluerov2/
[3] Stackpole, Eric, and David Lang. "Aruna: An Open Source ROV." Hackaday, 7 Oct. 2020.
https://hackaday.com/2020/10/07/aruna-an-open-source-rov-for-affordable-research/
[4] MIT Sea Grant. "SeaPerch Build Guide." SeaPerch, 2020.
https://seagrant.mit.edu/basic-seaperch-build-guide/
[5] Aguirre-Castro, Oscar A., et al. "Design and Construction of an ROV for Underwater Exploration." Sensors, vol. 19, no. 24, 2019. https://doi.org/10.3390/s19245387
[6] Herrin, Baker, et al. "Modularis: Modular Underwater Robot for Rapid Development and Validation of Autonomous Systems." arXiv, 2024. https://arxiv.org/abs/2401.06243

[7] Aristizábal, Luis M., et al. “Design of an Open Source-Based Control Platform for an Underwater ROV.” Dyna (Medellín), vol. 83, no. 195, 2016, pp. 198–205. https://www.redalyc.org/journal/496/49644128025/html/
[8] OpenROV. “Trident Underwater Drone.” OpenROV, 2018. https://github.com/OpenROV
[9] Marine Advanced Technology Education (MATE) Center. “MATE ROV Competition.” MATE, 2025. https://www.marinetech.org/rov-competition
[10] ros_rov Contributors. “ros_rov: An Open-Source ROS Platform for Underwater Vehicles.” GitHub, 2024. https://github.com/CentraleNantesROV/bluerov2
[11] OpenAI. “ChatGPT.” Accessed June 2025. https://chat.openai.com/