RoboSub 2025 Technical Report

Matthew Jiang Ekamresh Vasudevan Ayush Agarwal
University of Southern California
Los Angeles, CA, USA
{jiangmy, evasudev, ayushaga}@usc.edu

Abstract

We present Barracuda - the newest installment in USC AUV's long line of autonomous underwater vehicles (AUVs). For the first time ever, our AUV is equipped with a fully integrated pneumatic system consisting of grabber, dropper and torpedo launcher. Our electrical systems integrate newly acquired active and passive sonar systems, Doppler Velocity Log (DVL), stereo cameras, and Inertial Measurement Unit (IMU) for comprehensive environmental sensing. The software architecture features a modular, containerized system with advanced 3D Simultaneous Localization and Mapping (SLAM) using the GLIM framework, RRT path planning, and Linear Quadratic Regulator (LQR) control for precise autonomous navigation and task execution.*

1. Competition Strategy

1.1. General Strategy

Barracuda is the newest installment in USC AUV's line of autonomous vehicles and hopes to usher in a new era for the team. Using insights from past competitions as well as refining our team's strategy, our team spent more time focusing on reliability and replicability for design, integration and testing. Overall, we sought to simplify the mechanical structure of the AUV while maintaining the same performance as previous iterations.

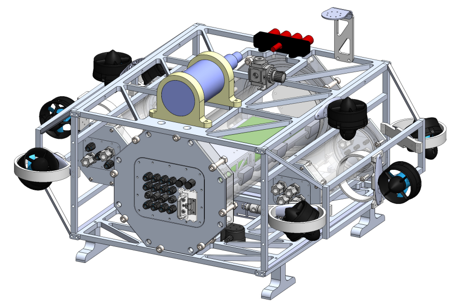

Figure 1. Full CAD model of the Barracuda, including its actuators, manipulators and sensors.

Given that our team only had around 8 months to complete the design of our new AUV, our design and manufacturing process was planned to ensure a modular design which could easily be revised after testing, without having to construct an entirely new AUV.

1.2. Course Strategy

Our course strategy involves completing tasks that rely on simple navigation and localization. This includes tasks such as Heading Out (Coin Flip), Collecting Data (Gate), and Navigate the Channel (Slalom). Manipulation tasks are planned, but not the priority.

To complete the navigation and localization tasks, the AUV will utilize its ability to localize to a map with the 360° sonar and depth camera SLAM. It will then be able to navigate a selfselected goal point using an RRT*-based planner, and LQR full state control.

In order to perform manipulation tasks such as Drop a BRUVS (Bin), Tagging (Torpedoes), and Ocean Cleanup (Octagon), the AUV will utilize a YOLO-based computer vision system to detect the goals and target them. The AUV will combine its navigation system with the pneumatic-actuated manipulators to perform BRUV dropping, torpedo shooting, and trash grabbing.

2. Design Strategy

2.1. Mechanical Systems

Barracuda features an aluminum 6061 T6 skeletal frame supporting three acrylic pressure vessels, eight thrusters, and a pneumatic system with four actuators. The design prioritizes modularity, stability, and straightforward assembly with a low center of gravity and high center of buoyancy for self-correcting stability.

2.1.1 Structural Framework

The frame consists of eight components: a top plate, bottom plate, four feet, and two wing assemblies. The 20×25-inch bottom plate mounts the main hull, pneumatic systems, DVL, and depth sonar. The top plate houses the 3500 psi air tank, manifold, and active sonar. Wing assemblies, fabricated from 0.5×0.5-inch aluminum bar stock with 0.4×0.4-inch diagonal reinforcements, mount four thrusters each at 30° inward angles.

2.1.2 Pressure Vessels

Three pressure vessels use 0.25-inch-thick acrylic tubing: an 8-inch-diameter main hull and two 4-inch-diameter side hulls. Octagonal flanges epoxied with Weld On 40 create airtight seals. The main hull uses 1.5-inch-thick flanges with dual O-ring grooves, while side hulls use 1-inch-thick flanges with single grooves. Aluminum bulkheads with removable panels house wet connections. The main hull features a clear polycarbonate panel for the depth camera and an aluminum panel for cable glands.

2.1.3 Thruster Configuration

Eight thrusters provide 6DOF control: two forward-facing (surge), two upward-facing (heave), and four angled 30° inward and 45° upward. The angled thrusters support primary motions while enabling sway, roll, pitch, and yaw control. Surge thrusters are positioned for maximum yaw torque. The vehicle maintains slight positive buoyancy, requiring continuous upward thrust for submersion.

2.1.4 Pneumatic System

The pneumatic system operates three attachments at 100 psi: grabber arm, torpedo launcher, and ball dropper. A two-stage regulator reduces tank pressure from 3500 psi to 350 psi (first stage) then to 100 psi (second stage). A 1-to-4 manifold distributes air to four double-acting cylinders controlled by solenoid valves. The grabber uses a scissor linkage with low-friction bearings, while the dropper employs movable extrusions for ball release.

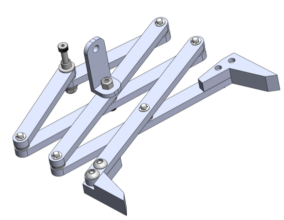

Figure 2. Barracuda's grabber, a scissor linkage with low friction bearings.

2.2. Electrical Systems

The main purpose of the electrical systems is to allow for the integration of the software systems with the mechanical design. By providing interfaces between micro-controllers and sensors, control of the pneumatic system, an external kill-switch for the safe power disconnection and more, the electrical systems ensure safe and reliable operation of the AUV.

Our electrical systems consist of 4 PCBs: one for power distribution, one for sensor integration, one for housing Electronic Speed Controllers (ESCs) and an internal thruster kill-switch and one for pneumatic control. We have two single board computers in our AUV, a Jetson AGX Orin and a Raspberry Pi 4B. Together, they work to allow for complete autonomous functionality of the AUV.

2.2.1 Thruster Control Board

The eight thrusters are powered by a pair of thruster control boards, each housing four Blue Robotics ESCs. On each board, a Teensy 4.0 microcontroller sends Pulse Width Modulation (PWM) signals to the thrusters and communicates with the Jetson AGX Orin. The Teensy was a later addition to the control board, thus an interposer board was designed to connect it with pre-existing pins and handle DC/DC power conversion from 18V to its operating voltage of 5V.

The Teensy communicates with the Jetson using I2C lines that originate off-board. Since the thruster boards are powered by different batteries from the other PCBs, we use optocouplers on each thruster board to ensure signal integrity and communication across different boards.

We implement a secondary kill-switch on these PCBs which automatically powers off the thrusters if a current limit is exceeded. Using a high-side gate driver, each thruster board will cut-off power if more than 30 Amps are drawn. This provides a native fail safe to prevent dangerous current spikes. In addition to this, we utilize a TLI4971A050T5UE0001XUMA1 to provide live current data to the Jetson AGX Orin. This gives the AUV an idea of how to adjust its thrust output during the mission.

2.2.2 Camera

Two cameras are used for navigation. The ZED M stereo camera is a lightweight solution for enabling depth-sensing capabilities. It also features an IMU to inform the Jetson AGX Orin of the AUV's positioning and acceleration. The Blue Robotics HD USB camera handles aligning with markers at the bottom of the pool. Both interface with the Jetson AGX Orin via USB connections.

2.2.3 Passive Sonar Board

Passive sonar is one method the AUV uses to autonomously navigate the pool in the RoboSub competition. Barracuda houses a 3-hydrophone L-shaped array interfaced with a dual-core Arduino Portenta H7 to handle digital signal processing.

Before the Portenta's STM32-based ADCs we design an analog signal conditioning board that integrates a preamp, variable gain, and a low pass filter to achieve a 40 dB signal to noise ratio (SNR). The preamp and variable gain stages employ a non-inverting op-amp at the input of an AD605 variable gain amplifier. This is then filtered using the LTC1562 configured to achieve a 4th-order Bessel low pass filter with a 70 kHz cutoff, well above the range necessary to receive the maximum pinger frequency. Finally, Schottky diodes at the ADC input protect the Portenta's internal circuitry.

The digital processing stage in the STM32 begins with time-gating a pinger pulse by checking if the slope of the incoming signal exceeds a specified threshold value. Then the signal is converted to the frequency domain (FFT) and band passed around the frequency of interest. This allows us to extract phase information and calculate the angle-of-arrival (AoA) using the formula

$$\theta_{\text{arrival}} = \arccos\left(\frac{\Phi\lambda}{2\pi d}\right) = \arccos\left(\frac{\Phi C}{2\pi d f}\right)$$

where λ is the wavelength, d is the distance between the hydrophones, C is the speed of sound in the pool, and f is the pinger frequency between 25 and 40 kHz.

2.3. Software Systems

The Barracuda software systems are a modular, containerized system built on Robot Operating System (ROS) 1 Noetic [11] with Ubuntu 20.04 LTS. Each functional node operates as an independent Docker container, enabling robust inter-node communication through standard ROS topics while maintaining system reliability and scalability. A full software architecture diagram can be found in Fig. 5.

2.3.1 System Architecture

The architecture centers on two compute platforms: a Jetson AGX Orin for main processing and a Raspberry Pi 4B for auxiliary tasks. Systems communicate through an internal Ethernet switch connecting the Waterlinked A50 DVL, while I2C links the Jetson AGX Orin to a Teensy 4.0 for low-level hardware control. The containerized approach ensures consistent deployment and simplified dependency management.

2.3.2 System Network Layout

The Jetson AGX Orin acts as DHCP server for devices across the AUV, utilizing `systemd-networkd` to allocate IP addresses to each device connected to it over the Ethernet switch, such as the Raspberry Pi 4B and the Waterlinked A50 DVL. Additionally, we have configured the Jetson AGX Orin to also act as a DNS Resolver to simplify network communication for our software team. Both of these configurations are also scalable, allowing for more devices to be added to the network later.

2.3.3 Perception and Sensor Integration

Visual perception uses a ZED M stereo camera for depth estimation and a Blue Robotics HD USB camera [3] for challenging conditions. The `barracuda-vision` node processes multi-camera input using custom PyTorch models trained through Roboflow for real-time object detection and classification.

Acoustic sensing includes a Ping360 sonar [1] for obstacle detection, Ping altimeter [8] for depth measurement and bottom tracking, and a passive sonar for landmark finding.

Active sonar data converts to occupancy grids for navigation planning. Passive sonar helps to localize the AUV against a known landmark. A Phidgets IMU [10] provides inertial measurement while the Waterlinked A50 DVL estimates velocity.

2.3.4 Localization and State Estimation

State estimation uses an Extended Kalman Filter (EKF) [12] through the `robot_localization` ROS package [9] in the `barracuda-localization` module. The filter fuses DVL, IMU, and sensor data to estimate position, orientation, and velocity in real-time within the map reference frame.

2.3.5 Mapping and 3D SLAM

The `barracuda-mapping` module uses the GLIM framework [6] for underwater 3D SLAM. GLIM uses fixed-lag smoothing and keyframe-based point cloud matching to handle degenerated range data while reducing drift. GPU acceleration enables real-time performance with multi-modal sensors including sonar and depth cameras.

2.3.6 Path Planning and Navigation

The `barracuda-navigation` module implements RRT* [5] algorithms for collision-free path planning. The system uses GLIM-generated submaps for obstacle detection and avoidance. Trajectory generation employs cubic spline interpolation to create smooth, dynamically feasible paths, visualized in Fig. 3.

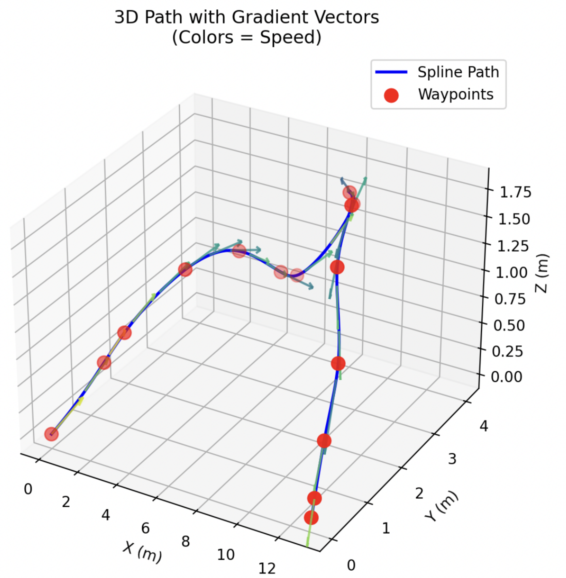

Figure 3. Demo calculating gradient vectors for a trajectory generated from a set of waypoints.

2.3.7 Control Systems

Barracuda implements a Linear Quadratic Regulator (LQR) [2] using ETH-ADRL's Control Toolbox [4] in C++ with 15 state variables, and incorporates vehicle dynamics, damping coefficients, and added mass effects.

The `barracuda-control` module receives target poses and computes thruster commands for smooth trajectory tracking under varying conditions. It uses a thruster manager [7] to compute desired thrust vectors for each of the eight thrusters based on an input URDF file.

2.3.8 Pneumatic Control

The `barracuda-manipulation` module manages GPIO pins on the Raspberry Pi 4B that are connected to each of the actuators in the pneumatic system, determining when the grabber, torpedo, and dropper are activated and deactivated. ROS services [11] that toggle these states are exposed to other nodes.

2.3.9 Mission Management

Mission logic coordinates high-level task execution and autonomous decision-making through structured task allocation and automatic sequencing. The system interfaces with all systems to coordinate sensor data collection, path planning, and control execution.

Barracuda's software systems implement different services for key tasks such as object location, navigating to a point, and toggling actuators. An action library combines these services into ordered groups that implement higher level tasks such as navigating to an object, passing through the gate, or doing a 360° depth scan using the depth camera.

The mission planner combines these actions into a final routine that completes the desired tasks in order.

2.3.10 Hardware Abstraction

Dedicated driver nodes provide standardized access to sensors and actuators: `barracuda-imu`, `barracuda-dvl`, `barracuda-sonar`, and `barracuda-camera`. The `barracuda-thruster-output` *-controller* interfaces with the Teensy 4.0 via I2C for precise motor control with electrical isolation.

The `barracuda-thruster-output -controller` node subscribes to each of the thruster topics, where the messages published to each represent the force in Newtons to be applied by the thruster. It then translates these force values into values that will be outputted on the PWM pins on the Teensys, which are connected to the thrusters. This node runs on a Raspberry Pi 4B which sends these PWM values to the Teensys over I2C.

2.3.11 Development Infrastructure

Development uses GitHub with git submodules, Docker containers, and Kubernetes orchestration. The `barracuda-simulation` repository provides URDF descriptions and Gazebo environments using the DAVE [7,13] framework for comprehensive testing prior to water trials.

3. Testing Strategy

Barracuda's control and navigation nodes are validated using a Gazebo-based underwater simulator called DAVE [13]. Models of important RoboSub landmarks such as the gate are created and placed in an underwater world.

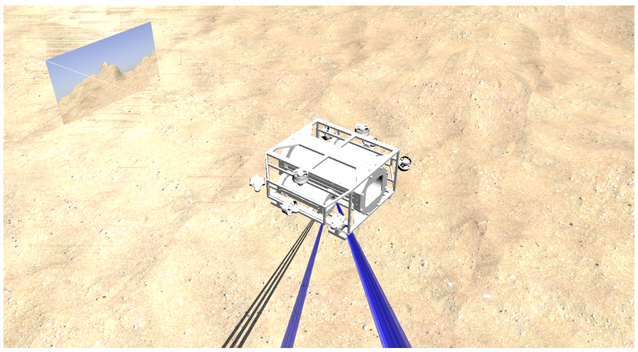

Figure 4. Barracuda shown in a simulation environment.

Safety checks and electrical tests are performed before water trials. To ensure that the vehicle is sealed before it is placed in water, we vacuum air from the hulls and measure pressure change inside the hull over time. If the pressure stays low, we know the hull is properly sealed.

Software and electrical components are first individually tested for failure cases. After ensuring they operate correctly on a consistent basis, they are end-to-end tested in simulation. Finally, they are tested with hardware. Before testing Barracuda underwater with its electrical components inside, we ensure that its kill switch is operable. Whenever the vehicle is being tested in the water, there is at least one member of our team accompanying it in case of failure.

Our testing strategy involves iteratively designing solutions to the problems we face. After conducting safety tests, we test the vehicle's possible failure points and identify problems that arise. After developing and implementing solutions to the problems, we repeat this process to retest and improve the vehicle.

4. Acknowledgements

We thank the University of Southern California Viterbi School of Engineering for their support of the USC AUV team, our faculty advisor and mentors for their guidance, and all team members who contributed to Barracuda's design and development. We also acknowledge our sponsors and industry partners for their technical expertise and financial support, and the RoboSub competition organizers for providing a platform to advance underwater robotics technology.

References

[1] Tudor Alinei-Poiana, David Rete, Davian Martinovici, Vicu-Mihalis Maer, and Lucian Busoniu. A bluerov2-based platform for underwater mapping experiments. *arXiv preprint*, 2024. Utilizes BlueROV2 with various sensors including Blue Robotics components for underwater mapping. 4

[2] Brian DO Anderson and John B Moore. *Optimal control: linear quadratic methods*. Prentice-Hall, Upper Saddle River, NJ, 1990. 5

[3] Blue Robotics. *Low-Light HD USB Camera with Wide-Angle, Low Distortion Lens*. Blue Robotics Inc., Torrance, CA, USA, 2025. Technical specifications: Sony IMX322/323 sensor, 1080p resolution, H.264 compression. 4

[4] Markus Giftthaler, Michael Neunert, Markus Stäuble, and Jonas Buchli. The control toolbox — an open-source c++ library for robotics, optimal and model predictive control. *2018 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, pages 123–129, 2018. 5

[5] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011. 4

[6] Kenji Koide, Masashi Yokozuka, Shuji Oishi, and Atsuhiko Banno. Glim: 3d range-inertial

localization and mapping with gpu-accelerated scan matching factors. *Robotics and Autonomous Systems*, 179:104750, Sept. 2024. 4

[7] Musa Morena Marcusso Manhães, Sebastian A. Scherer, Martin Voss, Luiz Ricardo Douat, and Thomas Rauschenbach. UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation. In *OCEANS 2016 MTS/IEEE Monterey*. IEEE, sep 2016. 5, 6

[8] Andreas B. Martinsen and Anastasios M. Lekkas. Bluerov2 benchmark: A benchmark model environment for remotely operated vehicle control. *Journal of Marine Science and Engineering*, 10(12):1898, 2022. Comprehensive BlueROV2 simulation model including sensor integration. 4

[9] Tom Moore and Daniel Stouch. A generalized extended kalman filter implementation for the robot operating system. In *Intelligent Autonomous Systems 13*, volume 302 of *Advances in Intelligent Systems and Computing*, pages 335–348, Cham, 2016. Springer. Originally presented at the 13th International Conference on Intelligent Autonomous Systems (IAS-13), Padova, Italy, July 2014. 4

[10] Phidgets Inc. PhidgetSpatial Precision 3/3/3 - MOT0110_0 Technical Documentation, 2023. 3-axis accelerometer, gyroscope, and magnetometer with AHRS/IMU algorithms. 4

[11] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng. Ros: an open-source robot operating system. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*, Kobe, Japan, May 2009. 4, 5

[12] Randall C Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research*, 5(4):56–68, 1986. 4

[13] Mabel M. Zhang, Woen-Sug Choi, Jessica Herman, Duane Davis, Carson Vogt, Michael McCarrin, Yadunund Vijay, Dharini Dutia, William Lew, Steven Peters, and Brian Bingham. Dave aquatic virtual environment: Toward a general underwater robotics simulator. In *2022 IEEE/OES Autonomous Underwater Vehicles Symposium (AUV)*, pages 1–8, 2022. 6

Appendix A. Technical Specifications

Parameter	Value
Dimensions (L×W×H)	3ft $\times 2ft \times 2ft$
Dry/Submerged Weight	85 lbs

Table 1. Key vehicle specifications.

Appendix B. Sensor Specifications

Device	**Specs**	**Vendor**
ZED M stereo camera	1280×720@60fps	StereoLabs
HD USB camera	1920×1080@30fps	Blue Robotics
PhidgetSpatial Precision 3/3/3	3-axis gyro/accel/mag	Phidget
Ping360 sonar	0.75-50 meter range, 0.16-4.1 centimeter accuracy, 2° resolution	Blue Robotics
Ping altimeter	0.3-100 meter range, 1-50 centimeter accuracy	Blue Robotics
Custom 3-hydrophone L-array	25-40 kHz, 40 dB SNR	In-house

Appendix C. Microcontrollers, Embedded Computers, Network Devices

Device	Vendor
Raspberry Pi 4B	Raspberry Pi Foundation
Jetson AGX Orin	NVIDIA
Teensy 4.0	PJRC
5-port Ethernet Switch	Blue Robotics

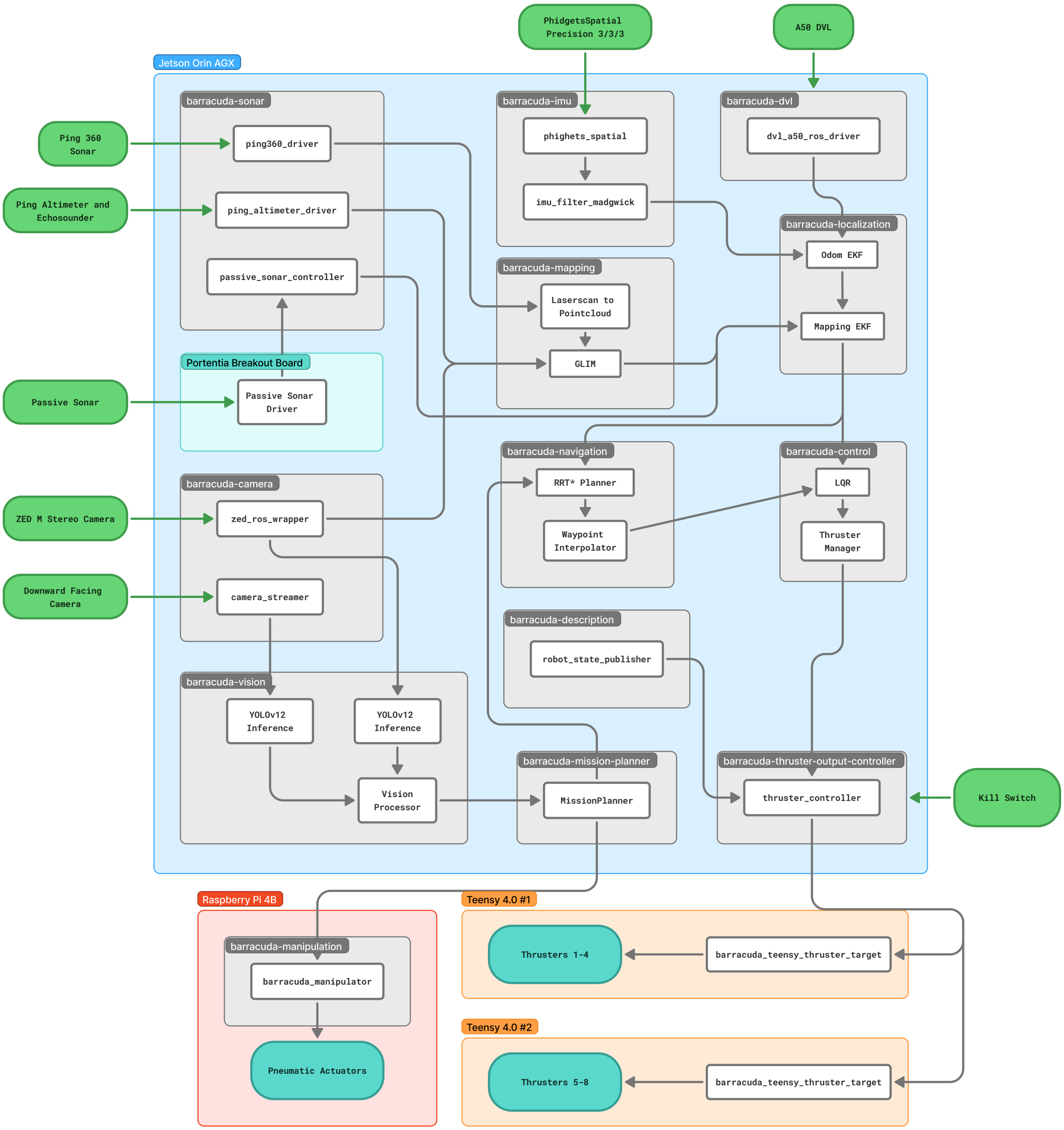

Figure 5. Complete Barracuda Software Architecture Diagram