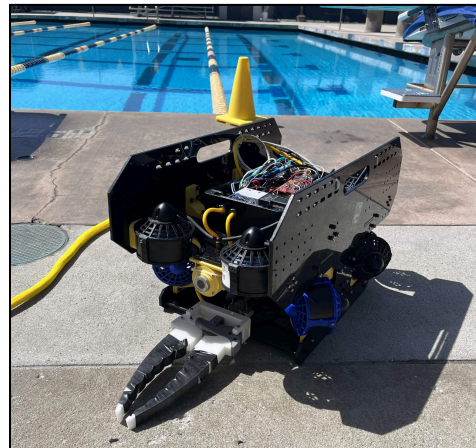
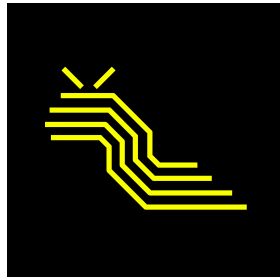


Slugbotics

MATE ROV Explorer 2026 Technical Documentation

UC Santa Cruz, Santa Cruz, CA



Full Roster

Andy Wu – CEO

Jacob Long – CFO, Electrical, Movement subteam lead

Sebastian Palmerin – Mechanical, Movement

Stas Novikov – Electrical, Movement

Michael Wray – Electrical, Movement

Sean Goudie – Safety Officer, Electrical, Integration subteam co-lead

Nathan Pham – Electrical, Integration subteam co-lead

Ethan James – Electrical, Integration

Trillian Williams – Electrical, Integration

Jiaying Xu – Electrical, Integration

Shari Hadi – Electrical, Integration

Ollie Wendt – Electrical, Integration

Leah Do – Electrical, Integration

Jaren Tennort – Mechanical, Frame subteam lead

Zander Mendoza – Mechanical, Arm subteam lead

Jason Yang – Software, Software subteam lead

Audencio Dimas – Software, Software

Rafael Garcia – Software, Software

Hiep Ho – Software, Software

Zander Mendoza – Mechanical, Arm

Megan Lee – Mechanical, Arm

No mentors

Table of Contents

Full Roster	2
Table of Contents	3
Abstract	3
Team Structure	4
Project Schedule	5
Project Workflow	6
Design Rationale	7
Integration	7
Software	11
Movement	16
Frame	20
Arm	21
Safety	22
Testing and Troubleshooting	23
Acknowledgements and Sponsors	24
Appendix	24

Abstract

The Monterey Bay is a stronghold of immense biodiversity and at the forefront of marine conservation efforts. With its campus located within a short bus ride from its shoreline, UC Santa Cruz's relation with the bay has historically been symbiotic. As stewards of this environment, it's our responsibility to study and learn as much as we can from the bay. For Slugbotics in particular, UC Santa Cruz's robotics collective, the bay offers untapped potential to explore the intersection between marine biology and robotics.

For the 2026 Marine Advanced Technology Education (MATE) Center's Remotely Operated Vehicle (ROV) competition, Slugbotics MATE presents the *Sea Slug Mk. 2*. Through the academic year, our team of 21 active members has worked diligently to design, fabricate, debug, test, secure funding for, and compete with our ROV, all while balancing university life and academics. The result is a drastic improvement on our previous design, with reduced weight, increased controllability, and far better reliability.

With this year being the second time our team has ever competed, we hope to be able to see the growth of our team reflected in our performance. This technical documentation outlines our complete process of designing, fabricating, and testing the *Sea Slug Mk. 2*.

Team Structure

The Slugbotics robotics collective as a whole is a collection of several teams, all working on separate robotics projects. The Slugbotics MATE team is one of such teams, and is financially and administratively attached to the collective, and shares the same general brand image. Slugbotics MATE further organizes itself into several subteams, each of which is responsible for the design and development of a major component of the ROV. This approach allows individual team members to only need to focus on the development of their specific component, leaving higher level systems design and administration to the subteam leads.

At the head of the chain of command are the Chief Executive Officer (CEO), the Chief Financial Officer (CFO), and the Safety Officer. The CEO is primarily responsible for deciding the year-long project timeline, leading the team-wide general meetings, and making sure the subcomponents being developed in parallel are compatible with each other. The CFO is responsible for procuring funds for the project, managing the budget and Bill of Materials (BOM) of the project, as well as working out competition and travelling logistics. Finally, the Safety Officer is responsible for making sure all safety protocols are followed in the workspace and during testing.

This year, Slugbotics MATE organized itself into 5 subteams:

- 1) **Frame:** This subteam is responsible for designing and fabricating the external structure of the ROV. This includes the structural High-Density Polyethylene (HDPE) sheets that form the bulk of the ROV's structure and the 3D printed brackets that reinforce components of the ROV.
- 2) **Integration:** This subteam is responsible for the entire electrical system of the ROV, and also responsible for modifying the enclosure that houses the major electronics of the ROV. This team is also responsible for the construction of the tether.

- 3) **Arm:** This subteam is responsible for designing and building the entire underwater manipulator system. This includes the pneumatic tubing, its physical actuation, and the construction and placement of the manipulator.
- 4) **Movement:** This subteam is responsible for everything related to the movement of the main ROV chassis. This includes deciding on the number of thrusters, their placements, the embedded code and processors that control the thrusters, and the design of the motor shrouds.
- 5) **Software:** This subteam is responsible for the software on our topside computer, the software on the Raspberry Pi in the ROV, and their interfaces. Notably, they are responsible for developing our computer vision models and managing the cameras.

Project Schedule

At the start of the academic year, the team began with 3 weeks of onboarding to train the influx of new members, and to review the knowledge base of our existing members. This year, our team saw a sizable change in team composition, with some members from last year graduating or leaving, and many underclassmen joining our ranks. Over half of our team this year consists of new members, so we made it a priority to get members to understand the development process of an ROV.

In order to better accomplish this, we decided to practice by building a small-scale and cheap ROV out of spare materials and Commercial Off-the-Shelf (COTS) components. The prototype design featured 6 motors and camera, and had a frame constructed out of PVC pipes. The goal was to simulate the development of the final ROV, experiment with different design ideas, and to provide a rapid way for new members to understand the workflows and skills that go into building an ROV. After adequate testing of the prototype, we began the development of the final ROV, using the lessons learned from building the prototype to inform the design of the final product.

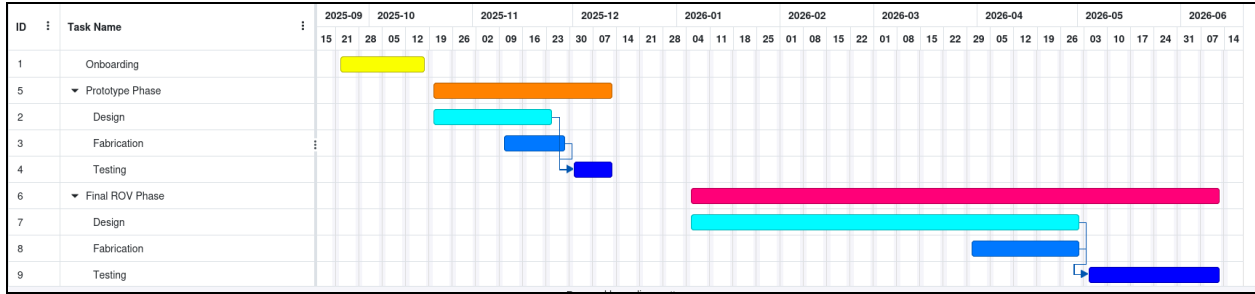


Figure: High-level Gantt chart of our development timeline

Project Workflow

Communication

Our team's primary form of communication is through Discord, a free messaging app that offers convenient tools to organize members, send files, and meet remotely.

File Sharing and Storage

We used a shared Google Drive to hold documentation, meeting notes, plans, and design docs, as well as administrative and financial documents. We used Git as our version control application, and use Github to store our repositories for topside code, embedded code, and circuit schematics.

Computer-Aided Design (CAD) and Manufacturing (CAM)

Onshape was our choice of parametric 3D modelling software, and was used to design the 3D printed and milled components on the ROV. It's web-based, allowing for anyone on any machine to use it, and it offers cloud collaboration, which drastically improved the speed that we could integrate designs together. Autodesk's CAM was used to route tooling paths, and was used to manufacture our HDPE frame. We used KiCad to create our custom PCB boards. It's free and open-source, and offers an easy-to-use workflow to integrate external libraries.

Design Rationale

In this section, each subteam's components and subsequent design choices on the final ROV are discussed.

Integration

Electrical Enclosure

The enclosure serves as the central hub of the ROV, housing all onboard electronics and power distribution systems. Therefore, key design requirements included complete waterproofing, an IP68 rating to withstand underwater submersion and space for electronics. For the prototype, a plastic enclosure was selected due to its low cost and ease of machining. Additionally, its transparent lid allowed internal components to be visually inspected without opening the enclosure.

However, testing revealed several limitations. The plastic enclosure raised concerns regarding long-term durability and leak resistance, particularly as additional holes and penetrators were added throughout development. Structural weakening from repeated drilling also reduced confidence in the enclosure's ability to support future expansion. As SeaSlug Mk2 was being designed, the power conversion system moved from externally mounted buck converters to an onboard PCB integrated directly within the enclosure. This introduced thermal management challenges due to increased heat generation in a confined environment with limited airflow.

To address these concerns, the final design transitioned to an aluminum enclosure. Aluminum provided significantly improved structural rigidity while acting as a passive heat sink for the onboard buck converter, which was mounted via thermal pads to the enclosure, improving thermal dissipation and reducing overheating risk. Unlike plastic, aluminum maintained mechanical integrity despite the addition of multiple penetrators and mounting locations. Because aluminum is electrically conductive, additional precautions were taken to isolate and secure electronics to prevent accidental short circuits or enclosure contact. Selecting a

commercially available enclosure also reduced manufacturing complexity while improving cost efficiency, internal organization, and overall system reliability.

Water Proofing



Figure: Penetrators, numbered 1 to 3 from left to right

The SeaSlug Mk2 was designed with an emphasis on modularity, maintainability, and reliability. Three penetrator types were utilized throughout the system, each selected for a specific function while maintaining manufacturing consistency by requiring the same drill bit size for enclosure integration (See above figure). Penetrator 1 allowed full cable or tubing pass-through while preserving cable integrity but required epoxy sealing after installation to maintain waterproofing. Penetrator 2 prioritized removability but required stripping the outer cable jacket to route individual conductors through the seal. Penetrator 3 functioned as the pressure release valve interface. For the motors, removable penetrators (Type 2) were selected to preserve cable length and provide additional slack outside the ROV for motor mounting and maintenance. Air tube, ethernet, cameras, and power wiring utilized Type 1 penetrators to maximize waterproof reliability while maintaining cable integrity. While this solution simplified sealing and reduced cost, it required cable lengths to be finalized before epoxy installation, increasing the importance of planning during assembly. All penetrator systems were designed to remain removable from the enclosure to improve serviceability and future modifications. Careful planning of penetrator sizing, placement, spacing, and internal electronics layout maximized enclosure space utilization while improving wire management and accessibility during maintenance.

Power Distribution

The *SeaSlug Mk. 2* is powered through a surface-side power supply system that delivers 48V power to the ROV through the tether. Because the propulsion system operates at 12V and onboard electronics require 5V, onboard power conversion and distribution are necessary. The original prototype design utilized three independent waterproof buck converters, each rated for 48V to 12V conversion at up to 40A rated continuous current. Each converter powers two motors, allowing the converters to remain external to the ROV enclosure to improve cooling performance and conserve internal space. Testing revealed several limitations with this architecture. The commercially available buck converters were unable to sustain their advertised 40 A continuous output, reducing overall system efficiency and reliability. Additionally, distributing motors across isolated converters created uneven load conditions.

To address these issues, the final design integrated a custom Power Distribution Printed Circuit Board (PCB) that centralized power conversion and distribution. Inside the ROV, a high-current DC-DC converter mounted directly to the PCB steps the voltage down from 48V to 12V, rated up to 120A continuously. The PCB distributes 12V power through nine output XT60 connectors: eight dedicated to propulsion motors and one shared between the lighting system and the pneumatic solenoid. To improve electrical reliability and fault protection, each output channel includes dedicated capacitive filtering and independent fuse protection. Individual fuses provide overcurrent protection for each XT60 output. A secondary 12V to 5V DC-DC converter mounted on the PCB supplies regulated 5V power through an XT30 connector to the Raspberry Pi and Pixhawk flight controller. Additional mechanical improvements included custom 3D-printed structural frames that secured components, and provided consistent thermal contact between the buck converter and enclosure floor for improved heat dissipation. These improvements increased reliability, improved power delivery consistency, and enhanced overall system maintainability during competition operations.

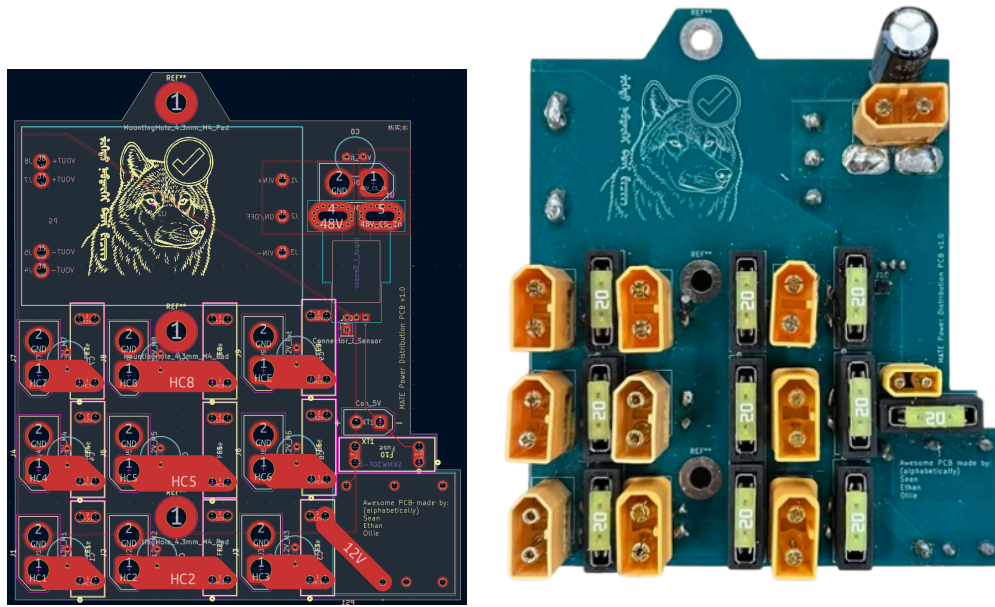


Figure: PCB layout and physical end product

Tether

The tether serves as the primary connection between the ROV and the surface control station, providing power, communication, and air while maintaining flexibility, durability, and reliability during operation.

The tether is composed of an Ethernet cable, 48V power and ground conductors, and two pneumatic air tubes bundled together within a braided expandable sleeve. One air tube supplies air into the ROV system while the second provides an air return path. The 48V power conductors deliver electrical power from the topside power system to the onboard Power Distribution PCB, where voltage is converted and distributed to propulsion and electronics systems. The Ethernet cable provides communication between the surface laptop and onboard Pi, enabling controls and camera data transmission. Bundling tether components into a single expandable sleeve improved organization, reduced snags, and simplified deployment while maintaining flexibility during vehicle operation. To improve mechanical reliability, the tether incorporates a dedicated strain relief system designed to prevent loading forces from being transferred directly to other connections. A shortened bungee cord is integrated alongside the tether components and attached to a dedicated structural mounting point on the ROV frame.

The Slugbotics tether management procedure was developed to improve operational consistency and reliability. Prior to deployment, the tether is visually inspected for damages. During deployment, a dedicated team member actively manages tether slack to reduce excess tension. Continuous tether monitoring also helps minimize drag and prevents unnecessary loading during maneuvering. Following recovery, the tether is cleaned, inspected, and recoiled to reduce long-term wear and maintain system reliability. Together, the tether design, strain relief implementation, and management protocol improved operational reliability while supporting the SeaSlug Mk2's design goals of modularity, maintainability, and robustness during competition operations.

Software

The Slugbotics topside vision infrastructure manages four high-definition video streams and handles real-time computer vision inference on the primary stream via an asynchronous, multi-processing architecture. High-resolution 1080p frame buffers are transferred across isolated process boundaries using zero-copy shared memory blocks to eliminate resource contention caused by Python's Global Interpreter Lock (GIL). A dedicated software watchdog monitor isolates individual stream failures and executes autonomous camera recovery without impacting overall pilot control loops.

Trade Off's & Limitations

While C++ extensions for image decoding (`cv2.read()`) and inference (`model.predict()`) release the GIL during execution, the surrounding Python framework required for frame handling and orchestration forces severe lock contention under multi-threaded models. Moving operations to isolated operating system processes achieves true multi-core parallel processing without thread blockage.

Shared Memory

Standard Python inter-process pipes rely on object serialization (`pickle`), creating severe CPU overhead when passing uncompressed 1080p images ($1920 * 1080 * 3 \sim 6.2$ MB). By

memory-mapping NumPy arrays directly onto pre-allocated raw byte blocks in RAM and using in-place pointer updates (`np.copyto()`), deep array allocations are entirely eliminated.

Race Conditions & Fixed Allocations

If synchronization flags (`frame_tick_X`) are mismanaged between processes, workers can read partially overwritten data or re-read stale frames. Additionally, buffers are pre-allocated at a rigid byte size based on expected image geometry, preventing dynamic resolution scaling on the fly without unlinking memory blocks and freezing the stream. Offloading our ~6.4 GFLOPS model to external coprocessors (Edge TPUs/GPUs) was rejected to eliminate physical bus bottlenecks (PCIe/USB data transfer latency). Performing inference on the host CPU takes 15–17 ms, keeping frames in host memory to maximize cache-locality for subsequent layout tiling (`combine()`) and pilot HUD overlays.

Autonomous Watchdog Monitor

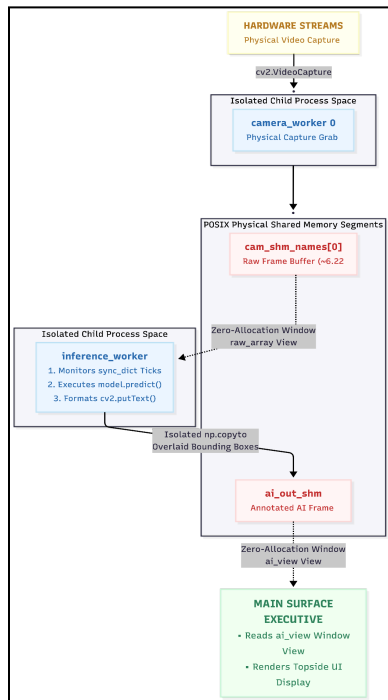
To mitigate stream hangs where a camera driver stalls but remains registered in the OS, an autonomous loop monitors the frame update interval (Δt). If a stream stops providing data for >2.0 seconds ("Soft Stall"), the worker process breaks execution, runs a forced `video.release()`, and re-initializes the handle without disrupting pilot control loops.

Computer Vision & Crab Image Detection

Slugbotics developed a custom deep learning pipeline using the YOLO11 architecture (`yolov11n.pt`). Processing at a designated 640×640 resolution, the lightweight 2.6M parameter model puts a computational load of exactly 6.4 GFLOPs per frame on the central processing pipeline, maintaining a stable multi-camera stream on the CPU. A programmatic data augmentation pipeline was engineered using the Albumentations library to simulate underwater pool distortions. To eliminate Inter-Process Communication (IPC) bottlenecks and Python Global Interpreter Lock (GIL) constraints, the pipeline utilizes an isolated multi-process producer-consumer architecture communicating via POSIX shared memory allocations. Real-time telemetry integrity is preserved through an independent watchdog loop and metric

logger, ensuring reliable, zero-allocation counting of invasive European Green crabs within a 0.5m to 1.2m operating envelope.

Multi-Process Producer-Consumer Architecture



Slugbotics allocated dedicated byte slots directly in physical RAM using Python's `shared_memory.SharedMemory` (Frame Size=1080×1920×3 bytes=6,220,800 bytes ~6.22 MB per frame). The `inference_worker` wraps this raw segment as a local `np.ndarray` window view (`raw_array`). This allows the model to execute predictions directly from the memory buffer written by the camera capture process, achieving zero-allocation data access.

Synchronized Sequence Tracking & CPU Throttling

1. To prevent the inference process from spinning continuously on duplicate data, frame state synchronization is handled via an integer counter inside a shared multiprocessing. The inference process reads `sync_dict["frame_tick_0"]`. An evaluation is executed if current frame tick diverges from the worker's tracked state (`current_tick != last_tick`).

2. If a new frame has not yet arrived, the worker enters a micro-sleep state (`time.sleep(0.002)`). This rate-limits the waiting cycle, preventing unnecessary CPU core utilization and thermal throttling.

3. Once `model.predict()` resolves, the resulting bounding box boundaries and classification target counts are overlaid directly onto the matrix via OpenCV.

4. The final annotated array is passed back to the master topside process using an in-place `np.copyto` command into the pre-allocated `ai_out_shm` region.

5. When an operator saves an inspection image ('A'), `cv2.imwrite` reads directly from the `ai_viewreference`, routing the image data straight to disk without needing high-overhead array extraction or duplication steps.

Photogrammetry

Slugbotics uses a pipeline for 3D modelling the ROV's environment by using COLMAP, an open source Structure-from-Motion (SfM) and Multi-View Stereo (MVS) to gather 2D images and convert them into a sparse 3D point cloud. COLMAP is then paired with OpenMVS for dense meshing to produce a .ply mesh file, analyzed on FreeCAD. To accommodate CPU-only usage, Slugbotics uses COLMAP instead of other software, as COLMAP is natively made for Linux. We also reduced the image quality to 1080p, capped the image count to 30, and mounted a white light source to provide flat lighting onto the target to counteract underwater distortion.

- *Average Image Count: 28.3 images*
- *Average Render Time 10 min 10 sec*

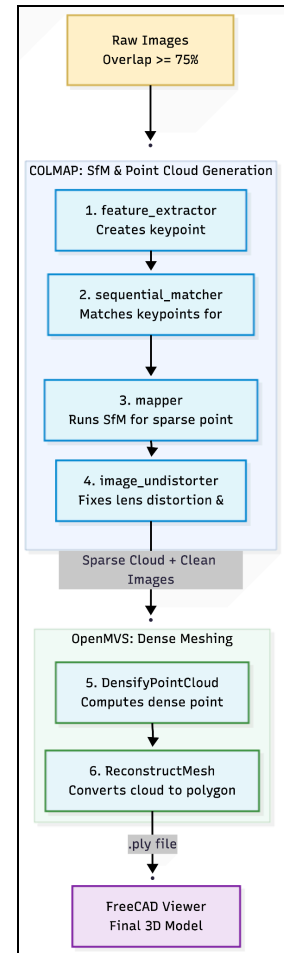
Manipulator Actuation Software

Software Design & System Integration

Slugbotics designed a custom Python-based software module to actuate the ROV's arm/claw. Rather than relying on standard PWM passthrough, we opted for a robust software-based control system integrated directly with the vehicle's BlueOS MAVLink router. The ClawController architecture utilizes the pymavlink library to establish a UDP connection, capturing telemetry and command data. The primary advantage of this software design is its ability to seamlessly intercept MANUAL_CONTROL MAVLink messages directly from the topside controller.

Operator Performance Enhancements

The software implements state-tracking toggle logic rather than a momentary hold. The MANUAL_CONTROL message array is continuously polled, and a bitwise operation isolating the 4th bit ($1 \ll 3$) checks for the "Y" button input. By implementing a debounce and state-tracking flag (`y_was_pressed`), the pilot only needs to press "Y" once to lock the claw open,



and press it again to lock it closed. This logic significantly reduces pilot cognitive load and hand fatigue during complex mission tasks requiring prolonged object manipulation.

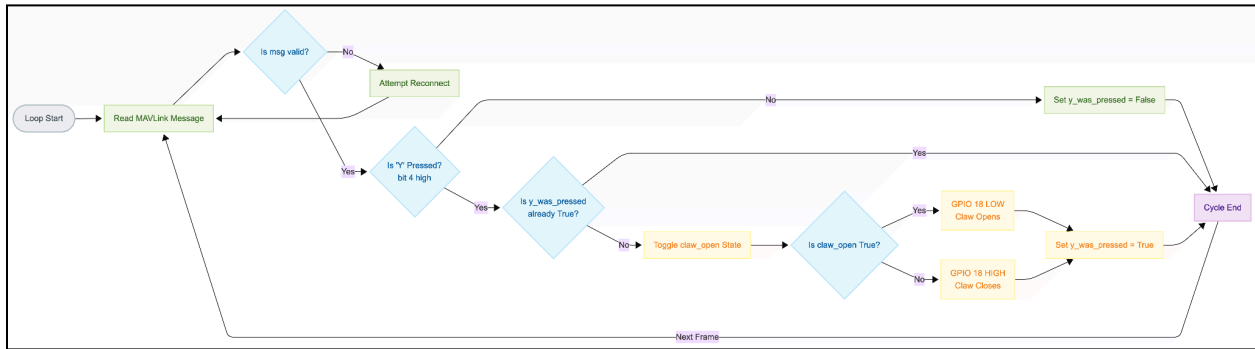


Figure: The progression from data ingestion (left) through debouncing logic (center), ending in physical GPIO manipulation.

Safety Fallbacks and Troubleshooting Equipment

Operational safety was the primary driver in our software development lifecycle. The script features robust troubleshooting strategies and error handling to protect the manipulator hardware and maintain mission uptime.

1. Fail-Safe Initialization: Upon startup, the GPIO pin is explicitly forced HIGH (`GPIO.output(self.pin, 1)`). This ensures the manipulator relay defaults to a safe, unpowered state before the pilot takes control.

2. Connection Resilience: The system relies on a `wait_heartbeat(timeout=10)` protocol to verify active communication with the vehicle. If the `MANUAL_CONTROL` stream drops, the software logs the null message and automatically triggers a `connect()` loop to re-establish communication without requiring a full system reboot, serving as an automated troubleshooting technique.

3. Graceful Degradation: A dedicated `cleanup()` procedure intercepts manual keyboard interrupts or fatal loop errors. This routine forces the relay back to the safe HIGH state and clears all GPIO allocations, preventing erratic actuator behavior during an unexpected software crash.

Movement

Raspberry Pi 4B

We chose the Raspberry Pi 4B to serve as the interface between the on-board systems and topside. This specific model was chosen due to us having it from previous years, as well as the performance requirements being suitable for our use case. We attached a HAT to it to allow for permanent connections, this HAT also contains an ADC0804LCN chip and a HUF75645P3 N-Channel Power Mosfet used to read analog voltage from the PDB's current sensor and sink the necessary current to control the pneumatic system's relay. We also added some thermal heatsinks to prevent the Pi from overheating.

The Pi is connected via GPIO pins to the pneumatic system's relay and through USB-C connection to the Flight Controller. This allows the Pi to act as the main bridge between software control, sensor feedback, and the physical hardware of the vehicle. Keeping the relay connected to the Pi also allowed us to keep the pneumatic control separate from the flight controller, reducing the amount of extra configuration needed on the Pixhawk.

BlueOS

We chose it since it immensely accelerated our development speed. It gave us a much easier way to configure, monitor, and communicate with the ROV without needing to create all of the software infrastructure ourselves. This was useful during testing because we were constantly adjusting motor settings, checking sensor values, and verifying that the flight controller was communicating correctly.

Flight Controller

The Pixhawk 6C served as our microcontroller; it shined as the most affordable, well maintained board that supported the ability to use the ArduSub firmware. We also had to consider noise and power stability, since motor current spikes or poor wiring could interfere with the flight controller and cause unreliable behavior. In combination with BlueOS it allowed us to have plug and play level usage and easy testing. It includes onboard sensors, gyroscope,

pressure, three axis-accelerator, that, in combination with multiple external sensors, allows us to properly maintain low-level vehicle control and stabilization on the ROV. The Pixhawk outputs PWM to the ESCs to control the motors, as well as reading inputs from the pressure and leak sensors, and controlling the brightness of the light. This made it the central control point for most of the ROV's movement and safety systems. Because of this, a large part of our testing was making sure the Pixhawk could reliably communicate with the Pi, read onboard sensors, and output the correct motor signals without resetting or browning out.

The firmware we loaded onto the flight controller was ArduSub, a derivative of the open-source ArduPilot operating system. This software provided pre-built control software specifically for underwater vehicles, including six-degree-of-freedom control, sensor integration, joystick input, and motor configuration. It is easily configurable, allowing for instant tuning of motors, frame configuration, and throttle limits. This helped us avoid having to write our own thruster mixing and stabilization software from scratch. This was especially useful when testing the eight-motor configuration, since motor direction, thruster mapping, and throttle limits could be adjusted without rewriting the control system. It also gave us a way to confirm that no axes were reversed and that the ROV would respond correctly before entering the water.

Thrusters

We used an eight-thruster setup of Blue Robotics T200 Motors, this configuration allows us to have six degrees of freedom in our movement alongside smoothing out any disturbance in pitch, yaw and roll. The motors are attached to custom 3D-printed mounts and guards, increasing the safety and protecting the propellers from damage while also letting the thrusters fit within the frame's dimensional constraints. A major design focus was placing the thrusters relative to the center of mass so that vertical movement would not accidentally create unwanted pitch or roll.

In the original prototype, we used 6 thrusters, with two vertical thrusters and four thrusters in the horizontal plane.

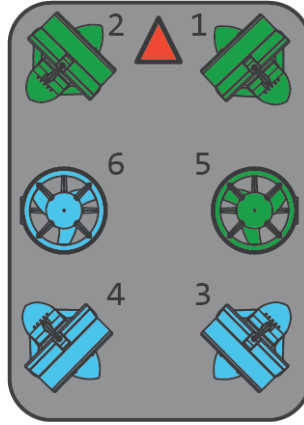


Figure: Layout of our 6 thruster configuration, top view (image taken from Ardusub)

However, we found through testing that this configuration was vulnerable to pitching forwards or backwards when thrusting up or down. Because of this, we decided to use an 8 motor configuration for the final product, featuring 4 vertical motors.

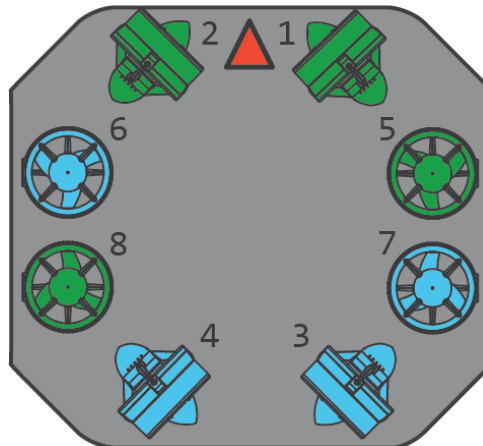


Figure: Layout of our final 8 thruster configuration, top view (image taken from Ardusub)

Electronic Speed Controllers(ESCs)

To provide the necessary three-phase voltage control that the T200 motors require, we made use of the Blue Robotics Basic ESC. Since the eight motors used at full throttle steady-state would draw too much power, we made use of the BLHeli software to reduce their max current draw. As we run the motors at 12V we chose to limit the ESCs to an ESC value: 1160 low 1840 high, which corresponds to a maximum current draw per motor of 12 amps. This reduced the total power demand of the vehicle while still allowing the motors to provide enough

thrust for our expected use. Before final assembly, the older ESCs were tested, cleaned, resoldered, and checked to make sure they could still reliably drive the motors. This also helped us avoid buying more new ESCs, reducing the overall cost of the movement system.

Leak Detection

In order to protect ourselves from any possible water coming through the aluminum enclosure, we attached a Blue Robotics' Leak Sensor. This sensor only has a single digital "Leak Detected" output, which reports to the flight controller. The water-sensing sponges are spread throughout the base of the enclosure to detect water in the areas where it would most likely collect first. This gives us a simple but important safety warning before the electronics are seriously damaged. Before water testing, we tested the leak sensor to confirm that it was sensitive enough to detect small amounts of water without giving constant false alarms. This was important because the leak sensor acts as one of the final safety checks before the electronics are at risk.

Pressure Sensing

We placed it outside of the direct path of any motors to limit the false readings that could be caused by thruster wash or rapid water movement. This allows the sensor to give more accurate depth readings and helps the vehicle maintain more consistent vertical positioning. This gives the pilot a better understanding of the ROV's depth and helps with any tasks that require holding a stable vertical position.

Lighting

Making use of an ROV Maker branded Undersea light, allows for better visibility especially in environments where the surface is covered. The light is useful for both navigation and task completion because the cameras become much more reliable when the area in front of the ROV is evenly illuminated. The light brightness is controlled through the flight controller, allowing visibility to be adjusted depending on the environment. This is important because the cameras are one of the main ways the pilot understands the vehicle's surroundings during underwater operation.

Frame

Material

The final material used on the *Sea Slug Mk. 2* was High-Density Polyethylene (HDPE). We decided on this material because it was cheaper than metals like Aluminum, had suitable strength, was relatively easy to machine, and is corrosion-resistant. HDPE was also machinable enough to allow us to drill in additional mounting holes after the sheets were already machined.

To support the HDPE, we used Polylactic Acid (PLA) 3D prints extensively for their ease of fabrication and affordability. PLA is relatively brittle and prone to heat, so we avoided its use near potential hot spots, such as the ground plane of the 12V net.

Exterior Structure

The exterior structure of the ROV was designed with three principles in mind: modularity, iterability, and ergonomics. Major issues from our previous ROV stemmed from its weight, size, and lack of mounting positions for peripherals. With these flaws in mind, the frame for *Sea Slug Mk. 2* was designed to be more compact, with plentiful mounting holes and room for different cable routing configurations.

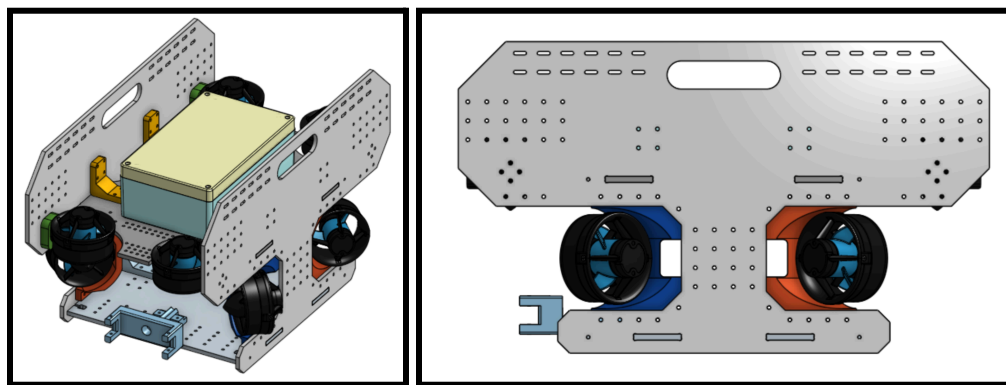


Figure: Views of the final CAD assembly of the frame. Final dimensions: 40cm x 33cm x 74cm.

Each mounting hole was designed to fit M3 screws, which reduces the amount of inventory management needed to construct and deconstruct the ROV, which improves the ergonomics of transporting the ROV.

Each HDPE sheet was designed to fit into perpendicular sheets using tabs and slots. These constrain the sheets together along the plane of the sheet with the slot, but does not prevent the sheets from moving apart from each other. So, T-slots were used to constrain the remaining axis, providing rigid perpendicular connections. PLA brackets were added to the sheets in order to increase rigidity, especially in the torsional axes of the ROV.

The electronics enclosure was the only necessary part on the ROV that had a significant amount of buoyancy. Thus, it was placed relatively high on the ROV, while the thrusters – being significantly less buoyant – were placed below the enclosure to provide hydrostatic stability.

A notable design decision was adding a carry handle on the top of the ROV, providing a safe way of transporting the ROV by hand.

Arm

Pneumatics

The pneumatic claw was chosen because of its high gripping strength and ease of use within the competition. The claw uses a piston with a 2.5 in stroke length to open and close the claw. The piston is controlled by a solenoid connected to a relay that is controlled by the Raspberry Pi. Air is delivered to the system through a line in the tether, which directs pressurized air to the solenoid that then delivers the air to the piston from two different outputs. The air is exhausted from the piston through the solenoid every use which is sent through a second line in the tether, back to topside to avoid a buildup of pressurized air in the enclosure.

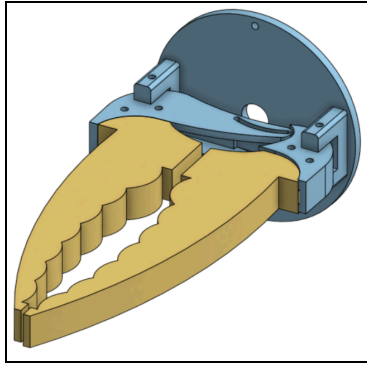


Figure: CAD Assembly of Arm Mechanism

Claw Mechanism

Both finger connections of the claw are independently connected to the piston at the center and a hinge. When the piston pushes the finger connections forward, they open on the hinge connection allowing the system as a whole to open and close.

Fingers

The finger connections and fingers were designed to allow us to easily remove and detach fingers from the system. This would allow us to test different finger designs to best decide which best fits the applications of the project. The fingers we are using for the project are as close to each other as possible to allow for maximum grip strength as well as feature grooves on the inside to increase friction between the fingers and objects grabbed.

Safety

Along with standard required safety features, the *Sea Slug Mk. 2* offers far more safety features than its predecessor. Firstly, the enclosure has a leak detector hooked up to the Pixhawk flight controller. Upon detecting a leak, the autopilot automatically attempts to resurface. If water begins to damage the onboard electronics, then our second safety feature kicks in. The buck converter on the power distribution PCB has short circuited output protection, automatically shutting off power to the entire ROV.

As mentioned in the frame section, the ROV has a carry handle that improves the ergonomics of transporting the ROV on the poolside, reducing strain on the lifters.

Testing and Troubleshooting

Before even designing the *Sea Slug Mk. 2*, the team rapidly built a PVC prototype to explore initial design options, and to help onboard new members of the team. The prototyping phase followed the same design-fabricate-test process that the final ROV ended up taking, allowing us to test project management strategies as well.

For testing the prototype and final ROV, we developed a set of procedures and observations to make at each stage of setting up and tearing down the ROV. This involved testing electrical connections, validating correct voltages, measuring temperature, and taking steady state and transient current draw readings while dry running the ROV – before sealing the enclosure and launching it into the water. While the ROV is running, at least 2 members are watching the ROV's behaviour at all times. Along with observations from the pilot, errors in the control of the ROV are noted. After the ROV is brought out of the water, the enclosure is immediately dried off and opened to check for leaks. If no leaks are detected, another post-run dry test is performed. If leaks are detected, the power is immediately shut off (as per our JSEA routine), and a case study on the leakage point is performed.

Electrical issues were our most common setback. Troubleshooting these issues involved using oscilloscopes and simulated waveforms to unit test portions of the electrical systems. Software issues were often unit tested in isolated Docker containers before being deployed to the topside laptop.

Acknowledgements and Sponsors

We would like to acknowledge UC Santa Cruz, Slugworks, and Baskin Engineering for graciously providing Slugbotics with space, tools, and funds to be able to build and participate in this competition. We would also like to acknowledge MATE ROV staff for providing feedback on our ROV.



UC SANTA CRUZ



Appendix

Bill of Materials

Material	16 Gauge Silicone Wire Kit	Amazon	https://www.amazon.com/203/	Received	\$37.98	1	\$37.98	\$1,183.81
Integration	SBC006UJ Arduino Ceramic	DigiKey	https://www.digikey.com/en/c	Received	\$9.11	3	\$27.33	
Material	1200 Threader w/ Stand M10 (	Blue Robotics	https://www.bluerobotics.com/stor	Received	\$240.00	2	\$480.00	
Material	Basic ESC	Blue Robotics	https://www.bluerobotics.com/stor	Received			\$0.00	
Integration	Powerwater 0-Ring Set (M10)	Blue Robotics	https://www.bluerobotics.com/stor	Received	\$3.00	1	\$3.00	
Item	Push to Connect Filings Pin	Amazon	https://www.amazon.com/Tab/	Received	\$12.99	1	\$12.99	
Integration	48V to 12V 120A Buck Converter	DigiKey	https://www.digikey.com/en/c	Received	\$124.88	1	\$124.88	
Integration	Enclosure	DigiKey	https://www.digikey.com/en/c	Received	\$49.80	1	\$49.80	
Integration	S0010 Blade Comp 10 13Aer	DigiKey	https://www.digikey.com/en/c	Received	\$8.89	4	\$35.56	
Integration	93 SC 02 connector (93000)	DigiKey	https://www.digikey.com/en/c	Received	\$29.25	1	\$29.25	
Integration	XT30 connectors	Amazon	https://www.amazon.com/Con/	Received	\$9.99	1	\$9.99	
Integration	XT30P connectors	Amazon	https://www.amazon.com/STL/	Received	\$8.99	1	\$8.99	
Integration	XT160 w/o casing	DigiKey	https://www.digikey.com/en/c	Received	\$2.50	15	\$37.50	
Integration	Current Sensor	DigiKey	https://www.digikey.com/en/c	Received	\$7.82	1	\$7.82	
Integration	20A fuse	DigiKey	https://www.digikey.com/en/c	Received	\$0.37	50	\$18.70	
Integration	Fuse holder	DigiKey	https://www.digikey.com/en/c	Received	\$0.75	25	\$18.75	
Integration	Thermal Pad	Amazon	https://www.amazon.com/3MS/	Received	\$8.99	1	\$8.99	
Material	Lumen Subsea Light for ROV, ROV/MAKER		https://www.lumenrobotics.com/produc	Received	\$79.00	1	\$79.00	
Software	Liquid epoxy		https://www.granquest.com/bran	Received	\$44.66	1	\$44.66	
Integration	JBLWest Tube epoxy			Received		2	\$0.00	
Item			https://www.digikey.com/en/c				\$0.00	
Item	Pressure integrator	Amazon	https://www.amazon.com/204/	Received	\$22.00	1	\$22.00	
Item	Tube Fitting	McMaster	https://www.mcmaster.com/2/	Received	\$4.00	2	\$8.00	
Integration	48V to 12V 120A Buck Converter	DigiKey	https://www.digikey.com/en/c	Received	\$126.76	1	\$126.76	

Figure: Full bill of Materials (BOM) of the parts needed for Sea Slug Mk. 2. Note that this BOM does not include parts already in inventory.

Budget as of 2/9/26			
Vendor	Amount (USD)	Item	
MATE	\$650.00	Explorer Competition Reg FEE	
Transportation(Airport)	\$3,002.00		
Transportation(Uber)	\$72.00		
Lodging	\$67.00		
Shipping	\$300.00		
Food	\$480.00		5429
Total	\$5,109.00		\$7,663.50
Per Person	\$1,277.25		
Recommend that we collect a total of \$XXX for each participant. We will refund any leftover money.			
We can collect in 2 installments:			
\$100/person by April 30			
\$100/person by May 31			

Location	Link	Cost(CAD)	Cost(USD)	Travel Method to Comp	Cost per Person	Includes Breakfast?
The Murray Premises	https://www.24hrs.ca	\$209.00	\$151.54	Shuttle	\$75.77	☐
Hilton Inn Garden	https://www.24hrs.ca	\$259.00	\$197.70	Shuttle	\$46.95	☐
Harfield by Marriott	https://www.marriott.com	\$229.00	\$168.66	Shuttle	\$67.22	☐
The Capital Hotel	https://www.mate.ca	\$169.00	\$122.54	Shuttle	\$30.43	☒
Memorial University Housing	https://www.mun.ca	\$110.00	\$79.70	Shuttle	\$39.88	☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐
			\$0.00			☐

FLIGHTS

Location	Link	Cost	Cost(US\$)	Travel Method to Comp	Cost per Person
GFO - YYT	http://www.airwestcan.ca	\$9,260.00	\$6,940.00		
GFO - YNT	http://www.airwestcan.ca	\$4,680.00	\$7,444.00		
GFO - YWF	http://www.airwestcan.ca	\$6,667.00	\$7,440.79		
GFO - YYT	http://www.airwestcan.ca	\$3,800.00	\$760.00		
GFO - YYT	https://www.airwestcan.ca	\$9,240.00	\$1,026.67		

SID

