

UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Design, Control, and Experimental Validation of a Six-Degree-of-Freedom Underwater Remotely Operated Vehicle for Photogrammetric Mapping

Permalink

<https://escholarship.org/uc/item/1nt2v0sn>

ISBN

9798197826664

Author

MacDonald, Reece

Publication Date

2026-07-02

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA SAN DIEGO

Design, Control, and Experimental Validation of a Six-Degree-of-Freedom Underwater
Remotely Operated Vehicle for Photogrammetric Mapping

A Thesis submitted in partial satisfaction of the requirements
for the degree Master of Science

in

Structural Engineering

by

Reece Gregory MacDonald

Committee in charge:

Professor Falko Kuester, Chair
Professor Stuart Sandin
Professor Mehran Tehrani

Copyright

Reece Gregory MacDonald, 2026

All rights reserved.

The Thesis of Reece Gregory MacDonald is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2026

TABLE OF CONTENTS

THESIS APPROVAL PAGE	iii
TABLE OF CONTENTS.....	iv
LIST OF FIGURES	vi
LIST OF TABLES.....	vii
ACKNOWLEDGEMENTS.....	viii
ABSTRACT OF THE THESIS	ix
Chapter 1 INTRODUCTION.....	1
1.1 Background and Motivation.....	1
1.2 Problem Statement	2
1.3 Approach.....	2
Chapter 2 LITERATURE REVIEW.....	4
2.1 The Use of ROVs for Photogrammetry	4
2.2 Existing Six-DOF ROV designs	5
2.3 Ardusub/MAVLink.....	7
Chapter 3 DESIGN CONSTRAINTS.....	10
3.1 Maneuverability and Control Authority.....	10
3.2 Precision, Automation, and Accessibility	11
Chapter 4 SIMULATION AND MODELING	12
4.1 Simulation Setup	12
4.2 Thrust Kinematics	13
4.3 Fluid Simulations	17
Chapter 5 SYSTEM DESIGN	23
5.1 Mechanical Design.....	23
5.2 Electronics and Hardware	25

5.3 Software Architecture	26
Chapter 6 AUTONOMOUS OPERATION FRAMEWORK.....	29
6.1 Flight Mode Customization	29
6.2 Photogrammetry Workflow	31
Chapter 7 EXPERIMENTAL SETUP AND TESTING.....	33
7.1 Final Assembly	33
7.2 Test Setup.....	34
Chapter 8 RESULTS.....	37
8.1 Angle Convergence.....	37
8.2 Pattern Thrust Verification.....	39
8.3 Photogrammetry	42
Chapter 9 CONCLUSION AND FUTURE WORK.....	46
9.1 Summary of Contributions and Key Findings	46
9.2 Limitations	46
9.3 Next Steps	48
REFERENCES	49

LIST OF FIGURES

Figure 2.1: (From left to right) Blue Robotics Blue ROV 2 Heavy, Oceanbotics SRV-8.....	6
Figure 4.1: Top view of the vehicle with thrust directions required to produce net F_x and M_z	16
Figure 4.2: Rear view of the vehicle with thrust directions required to produce net F_y and M_x ..	16
Figure 4.3: Side view of the vehicle with thrust directions required to produce net F_z and M_y ...	17
Figure 4.4: Cut plot of final mesh for rolling simulation.....	18
Figure 4.5a to c: Velocity flow trajectories used to calculate normal forces for surge, sway, and heave respectively	19
Figure 4.6a to c: Velocity flow trajectories used to calculate moments for roll, pitch, and yaw respectively	21
Figure 5.1: CAD model of entire ROV with numbered thrusters.....	24
Figure 5.2: Vehicle electronics ecosystem.....	25
Figure 5.3: Data path from user input to actuation	28
Figure 6.1: Onboard and offboard photogrammetry workflow	32
Figure 7.1: Final assemblies of the ROV frame and electronics sled.....	34
Figure 7.2: Initial testing of the platform verifying proper control in multiple DOFs	35
Figure 8.1: Attitude and commanded moments compared over time for rotational DOFs	38
Figure 8.2: Resultant forces on vehicle during horizontal lawnmower pattern (yawing to heading between rows)	40
Figure 8.3: Resultant forces on vehicle during horizontal lawnmower pattern (vehicle moving laterally between rows)	41
Figure 8.4: Resultant forces on vehicle during vertical lawnmower pattern (vehicle moving laterally between rows)	42
Figure 8.5: Comprehensive photogrammetry scan of pool testing area	43
Figure 8.6: (From top to bottom) Model sections reconstructed using horizontal and vertical lawnmower patterns	44
Figure 8.7: (From left to right) Close-up of prop chair in reconstruction compared to raw image from the onboard GoPro used to construct the model	45

LIST OF TABLES

Table 4.1: Thruster orientations (unit vectors)	13
Table 4.2: Thruster distances to the geometric center of the ROV (mm).....	13
Table 4.3: Summary of maximum theoretically achievable thrust for translational DOFs and their corresponding maximum velocities	20
Table 4.4: Summary of maximum theoretically achievable thrust for rotational DOFs and their corresponding maximum angular velocities.	22

ACKNOWLEDGEMENTS

I would like to acknowledge my parents, my brother, and my close friends for supporting me through the most difficult technical project I have ever embarked upon. Without their support, I would not be where I am today.

I would also like to acknowledge Professor Falko Kuester, as well as many other members of the Cultural Heritage Engineering Initiative, for providing me with the trust and resources necessary to complete my research.

ABSTRACT OF THE THESIS

Design, Control, and Experimental Validation of a Six-Degree-of-Freedom Underwater
Remotely Operated Vehicle for Photogrammetric Mapping

by

Reece Gregory MacDonald

Master of Science in Structural Engineering

University of California San Diego, 2026

Professor Falko Kuester, Chair

This thesis presents the development of an underwater remotely operated vehicle intended for photogrammetric mapping of complex underwater structures, including near-shore coral reef monitoring and structural health assessment of artificial structures. Current designs are either too large for shallow water deployment or lack the orientation flexibility required for capturing overhanging or occluded geometry - this design intends to address both limitations. A vectored eight thruster layout was devised to prioritize authority in all six degrees of freedom (DOFs). Multiple thruster configurations were assembled in CAD software and their authority

(available thrust or torque) in every DOF was calculated using simple MATLAB scripts to inform the design (The MathWorks, Inc. 2025). With a final configuration chosen, a simplified model was simulated in six fluid flow conditions, calculating drag forces to determine translational and rotational velocity limits. The platform was assembled and theoretical results were validated through numerous pool sessions, testing general functionality as well as the capability of pre-programmed python scripts for scanning purposes. Large, high-quality 3D models were constructed with the images captured in these tests. The success in operational capability of the platform creates a foundation for future systems to expand the envelope of what is possible in the field of autonomous underwater sensing.

Chapter 1 INTRODUCTION

This chapter introduces the motivation for developing a highly maneuverable underwater robotic platform for coral reef monitoring and photogrammetric mapping. By utilizing a six-DOF design, the platform aims to overcome the limitations of traditional top-down scanning methods and achieve high-resolution 3D reconstructions of complex underwater structures.

1.1 Background and Motivation

Research ROVs have developed significantly in the past 25 years - this industry has moved rapidly from strictly industrial platforms costing tens or even hundreds of thousands of dollars to platforms that can be assembled with some experience for less than a thousand. As with most other vehicles, the specific design is strongly influenced by its intended purpose. Some ROVs are designed to be small and nimble to navigate through tight areas, while some are the size of cars and are expected to carry heavy equipment or retrieve scientific samples.

One of the largest focuses of the current ROV landscape is collecting image data in order to construct a photogrammetric scan. A complete scan consists of sometimes thousands of individual images taken of an object from different angles, where a computer program will calculate distances between common reference points and produce 3D models with potentially greater than millimeter accuracy. One of the challenges with using ROVs to take these pictures instead of a diver is the level of freedom needed to position a camera to get an accurate scan - this is why most professional photogrammetry scans are performed by humans today. However, involving humans in any underwater application like this introduces a certain level of risk that can be mitigated by using a robotic platform instead, provided this platform can reach the same level of function (or greater) as a human operator. The ability to do this is in particular relevant

to the field of ecology; coral reefs cover only 0.1-0.5% of the ocean floor but support almost a third of the world's fish species (Moberg et al. 1999), and continuous monitoring is necessary as we continue efforts to rehabilitate those environments. Due to high irregularity in reef geometries, a platform capable of operation in arbitrary orientations would be beneficial for capturing odd angles, ensuring complete and detailed 3D models.

1.2 Problem Statement

There are plenty of ROV platforms commercially available that offer all six DOFs of control. These platforms, however, are still designed to be primarily operated with one configuration acting as its base state and it is expected that regardless of its movement capabilities, it will always return to this state. Thrusters are usually configured to have the highest authority in translation and yaw DOFs. While platforms like these are sufficient for most applications, detailed scanning operations might require sustained operation in alternate orientations. For example, when scanning a large, tall object on the seafloor, a pattern where the platform slowly spirals around the object while changing its pitch might be the best way to capture the most detail. To maximize efficiency and accuracy for these operations, there is a need for a platform that maintains near-equal authority in any orientation. The goal is to bridge the gap between the maneuverability of a human diver and the precision and repeatability of a robotic system.

1.3 Approach

A structured approach was used to complete the overall objective of constructing a fully functional platform capable of detailed photogrammetry scans. In the design phase, multiple motor configurations were modeled; since there is no such thing as a perfect configuration, it was

necessary to determine which DOFs needed stronger authority and choose a configuration that could satisfy those requirements. A complete CAD model was constructed in SolidWorks to ensure proper fitment of manufactured parts and electronics, as well as a basis for fluid simulations to be completed. A simplified version of the CAD was brought into ANSYS to determine drag produced under specific fluid conditions and further calculations were performed to extrapolate those findings into theoretical maximum velocities and angular velocities per DOF (Ansys, Inc. 2023). After ensuring that those results matched the initial actuation goals of the platform, the frame was manufactured and assembled along with all of the electronic components. Thrusters, cameras, and control scripts were tested out-of-water, and waterproofing was checked without electronics inside. With all systems functioning properly, carefully planned tests were performed to validate the control in each DOF, functionality of the pre-programmed movement scripts, and photogrammetry capabilities. This thesis will discuss the background knowledge necessary to understand these systems, the choices made for the design and what influenced them, how the platform was tested to confirm theoretical function, and where this project could be taken in the future.

Chapter 2 LITERATURE REVIEW

This chapter explores the academic and industrial landscape surrounding underwater robotics and 3D reconstruction. By examining the current use of ROVs in photogrammetry, evaluating existing six-DOF designs, and detailing the communication protocols that enable robotic control, this section establishes the technical foundation upon which this research is built.

2.1 The Use of ROVs for Photogrammetry

Photogrammetry is a method of creating three-dimensional models of an environment using a compilation of two-dimensional pictures. This works by using pictures from at least two different locations and using an algorithm to calculate the intersection of “lines of sight” (Teague & Scott 2017), which describe single point coordinates of a distinct shared feature between the images. Using hundreds or thousands of these images, more complex models can be reconstructed by solving for the coordinates of a large number of points. This is what is known as the sparse point cloud. The points can also be used to describe the positions and orientations of the images that provide the data for the model. After generating a sparse point cloud, a dense cloud is created using multi-view stereo (MVS) algorithms, which compute depth for almost every pixel across overlapping images (Nocerino et al. 2020). This is then converted into a triangular surface mesh and color information from the original images is mapped onto the surface, producing a highly accurate model without the need for advanced mapping sensors.

A large portion of photogrammetry conducted has been from the air; multirotor drones or even planes are very good at capturing images in very regular patterns; air as a medium causes negligible error and GPS allow for extremely accurate positioning. This all changes when the cameras are taken underwater - light behaves differently, turbidity causes locational disruptions,

and GPS is no longer available to hold position. Divers are often used instead of robots for these reasons but are significantly limited again by positional accuracy as well as a limit to the time spent in water (McCarthy & Benjamin 2014). With a properly constructed robotic platform, these issues could be eliminated completely.

Specific photogrammetry methods are outlined in various studies for high quality results using divers, which are similarly applicable to use on a robotic platform. 50%-75% overlap of sequential images is recommended (Bayley et al. 2020), and collecting images in a ‘lawnmower’ pattern (Burns et al. 2015) or an expanding spiral pattern (Pizzaro et al. 2017). A camera with a high megapixel image sensor, a wide-angle lens, and auto-focusing is ideal for capture. Fast shutter speeds reduce low-light image blur, fixed white-balance ensures similar color profiles between images, and oblique lighting angles help reduce backscatter. DSLRs are recommended for the breadth of camera settings and large image sensors, but with sufficient lighting a high-quality action camera is sufficient in test environments.

2.2 Existing Six-DOF ROV designs

As previously mentioned, many six-DOF control schemes exist for underwater ROVs. One of the most popular designs in the consumer market is the Blue ROV Heavy (Blue Robotics 2025), which uses four upward facing thrusters that actuate heave (vertical motion), pitch, and roll. Four vectored thrusters below the platform are responsible for surge (forward motion), sway (sideways motion), and yaw. This system has good authority in all DOFs, but its asymmetrical design could result in unfavorable flight characteristics if attempting to operate the vehicle in different orientations. For a system to be designed for maximum mobility and operation in a multitude of orientations, it was decided to use a configuration with its center of mass and geometric center in the same location.

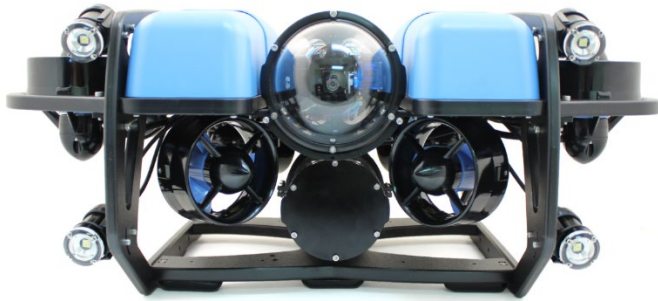


Figure 2.1: (From left to right) Blue Robotics Blue ROV 2 Heavy, Oceanbotics SRV-8

A more compelling design was found with the Oceanbotics SRV-8 (Oceanbotics 2025), which uses 8 vectored thrusters positioned around a central enclosure in a mostly symmetrical fashion. In order to achieve all DOFs with this kind of configuration, the thrusters must be vectored slightly more or less in the z direction than the y direction in order to achieve a moment arm about the x axis for roll actuation. In the case of the SRV-8, roll is not actually actuatable manually from the controller, rather it relies on it solely for stabilization purposes. Researchers Salatiel Garcia-Nava et. al. uses a very similar motor configuration to the SRV-8 (Garcia-Nava et al. 2023), however their design features thrusters that angle outwards as opposed to inwards. The angles they chose for the thrusters are equally biased towards the translation directions, but significantly more biased towards roll and pitch compared to yaw. Again, a perfect configuration does not exist, only a perfect configuration for a particular application. In the case of this project, more authority is desired in surge since it will be the primary direction of travel, as well as pitch and yaw which will be actuated more frequently than roll. Given these conditions, a configuration that falls between the Oceanbotics SRV-8 and the platform designed by Garcia-Nava et al. is ideal.

2.3 Ardusub/MAVLink

The basis for control systems on our platform is Ardusub, which communicates between a topside computer and the ROV via a protocol called MAVLink. Ardusub is based on Ardupilot a long-time favorite of the drone hobbyist community for its open-source software and extensive documentation. The project was originally developed for aerial vehicles but was expanded for use with rovers and submersibles through firmware designed for specific vehicles. Ardusub is the marine-vehicle branch of Ardupilot designed for ROVs. It provides the low-level control, state estimation, thruster mixing, sensor implementation, and communication protocols required for reliable underwater operation. Ardusub is often used in underwater research applications because it removes the need to build custom firmware from scratch. This allows researchers to focus on higher-level autonomy, mapping, perception, or mission-planning problems. For these reasons, Ardusub is an ideal control scheme for an experimental vehicle that will require frequent iterative modifications. It also provides support for flight modes that will directly support data collection such as “Depth Hold” and “Stabilize” (as well as Orbit with the use of additional position sensors).

MAVLink (Micro Air Vehicle Link) is a lightweight, binary communication protocol widely used in aerial, ground, and marine robotics. It defines a standardized way for an onboard autopilot, companion computers, payloads, and ground stations to exchange information with a ground station computer. MAVLink has become the dominant protocol for open-source robotics platforms such as Ardupilot and PX4, making it a critical technology for autonomous ROVs and AUVs. For underwater robotics, where communication latency, limited bandwidth, and deterministic behavior are important, MAVLink is ideal because it is compact, extensible, and designed for real-time control scenarios.

MAVLink provides a structured way to send and receive:

- State information (attitude, depth, velocity, battery, system status)
- Control setpoints (velocity, position, attitude, thrust)
- Mission commands (waypoints, guided mode instructions)
- Parameter changes (tuning PID gains, control settings)
- High-level events (arming, mode changes, failsafe)
- Custom payload data (camera triggers, sensor messages, research-specific data)

Pymavlink is the Python implementation of MAVLink and serves as an interface for communicating with autonomous vehicle firmware such as ArduSub. MAVLink structures the message format, and pymavlink provides the software interface that allows external programs, such as a companion or ground station computer, to send commands and receive telemetry using Python by means of the MAVLink messages. It provides a simple, scriptable way to implement mission-level autonomy, structured motion patterns, and custom sensing or control algorithms without modifying the underlying autopilot firmware. pymavlink enables Python programs to:

- Establish a communication link (serial, UDP, or TCP) with the autopilot
- Send standardized MAVLink messages (velocity, attitude, position setpoints, camera triggers, mode changes)
- Receive telemetry streams from the autopilot (depth, IMU data, battery, state information)
- Parse and log MAVLink packets for post-processing
- Create custom message handlers or event loops for autonomy

To summarize, pymavlink turns Python into a high-level mission-planning and control environment for ROVs. This is what will be used to develop low-level control scripts for the actuation of different lawnmower-style patterns for photogrammetry use later in the study.

Chapter 3 DESIGN CONSTRAINTS

This chapter establishes the technical and operational boundaries that guided the development of the platform. By defining specific constraints in maneuverability, symmetry, and cost, the design focuses on the main objective of high-resolution photogrammetric mapping in complex underwater environments.

3.1 Maneuverability and Control Authority

The most significant constraint governing this project is the need for 6-DOF movement, ensuring adequate authority in all possible orientations. Unlike traditional ROVs which are designed with a heavy bias toward relying on one level flight state, this platform must be capable of sustained operation in alternate orientations (i.e. being pitched ninety degrees nose-down or rolled on its side) to capture the oblique views necessary for reconstructing complex 3D geometries like coral overhangs. This objective necessitates a symmetric thruster layout and a centered mass distribution to ensure that control authority remains predictable and effective regardless of how the vehicle is oriented in the water column. While the platform is capable of movement in all six DOFs, the design specifically prioritizes authority in surge, pitch, and yaw to accomplish precise scanning paths. These specific DOFs are the most critical for executing the complex trajectories required for photogrammetry, such as spiraling around tall structures or maintaining a specific heading during lateral shifts. To remain effective in varied aquatic environments and overcome local currents, the platform is further constrained by a requirement to reach a maximum translational velocity of approximately 1 m/s.

3.2 Precision, Automation, and Accessibility

To provide a level of precision and repeatability that exceeds the capabilities of a human diver, the system is constrained by the need for a robust autonomous operation framework. A software architecture capable of redefining its coordinate system based on any arbitrary orientation and executing pre-programmed scripts is necessary to achieve precise, reliable scans. The platform must be able to maintain a commanded attitude and depth when subjected to external disturbances, and it must execute precise acquisition patterns, such as the 3D lawnmower or orbit paths, to ensure 75% image overlap or greater for photogrammetric processing. Finally, the project is bound by logistical and financial constraints intended to keep the technology accessible for small-scale research teams. The total system cost of the system must be much lower than other commercially available options, proving that high-level six-DOF capability does not require industrial-scale funding. Additionally, the vehicle must be portable enough for deployment by a small team and be rated for depths of at least 100 meters. To facilitate rapid iteration and lower manufacturing costs, the frame was designed to be produced using consumer 3D-printers without compromising the structural integrity required for underwater operations.

Chapter 4 SIMULATION AND MODELING

This chapter covers the thrust kinematics and computational fluid dynamics (CFD) used to validate the design before beginning physical assembly. Based on these two key modeling and simulation methods, boundary conditions and performance requirements defined by the design constraints were verified to ensure that the vehicle would achieve necessary authority in all six DOFs.

4.1 Simulation Setup

Before finalizing the physical construction and material selection, a preliminary CAD model was developed to serve as the foundation for both kinematic and fluid analysis. This digital prototype allowed for the iterative refinement of motor positions and angles, ensuring that the thrust vectors were optimized for the prioritized DOFs. By testing various configurations in a simulated environment, the thruster geometry could be adjusted to balance control authority in specific DOFs.

This preliminary model served as the basis for CFD simulations in ANSYS. By analyzing the simplified geometry under specific fluid conditions, it was possible to predict drag coefficients and verify that the vehicle could meet the translational velocity requirements established in the design constraints. Only after these simulations confirmed that the theoretical performance matched the project objectives did the design proceed to the finalization of material choices, component sourcing, and fitment. This workflow ensured that the physical implementation, described in the following chapters, was backed by a validated theoretical framework.

4.2 Thrust Kinematics

To confirm that the thrusters provide sufficient authority in each DOF, a thrust configuration matrix is assembled. First, the moment arms for each thruster are found by locating the geometric center and a central point on the thruster and drawing a vector between the two. This was done for all 8 thrusters, described as distance vectors. Two parallel points on the thruster mounts were also located using SolidWorks, and drawing a vector between those two points gives the orientation vectors. These matrices are depicted in tables 5.1 and 5.2.

Table 4.1: Thruster orientations (unit vectors)

	T1	T2	T3	T4	T5	T6	T7	T8
x	0.7668	-0.7668	0.7668	-0.7668	0.7668	-0.7668	0.7668	-0.7668
y	-0.3916	-0.3916	-0.3916	-0.3916	0.3916	0.3916	0.3916	0.3916
z	0.5085	0.5085	-0.5085	-0.5085	0.5085	0.5085	-0.5085	-0.5085

Table 4.2: Thruster distances to the geometric center of the ROV (mm)

	T1	T2	T3	T4	T5	T6	T7	T8
x	77.99	-77.99	77.99	-77.99	77.99	-77.99	77.99	-77.99
y	112.39	112.39	112.39	112.39	-112.39	-112.39	-112.39	-112.39
z	-80.16	-80.16	80.16	80.16	-80.16	-80.16	80.16	80.16

A simple MATLAB script constructs a 6x8 thrust mapping matrix T , where each column represents how one thruster contributes to the whole vehicle's motion. Each column of T is formed as:

$$T_i = \begin{bmatrix} f_i \\ r_i \times f_i \end{bmatrix}$$

And

$$T = [T_1 \quad T_2 \quad \dots \quad T_8]$$

The top three rows represent the linear forces in surge, sway, and heave directions, whereas the bottom three rows represent the rotational moments (roll, pitch, and yaw) produced by the individual thrusters about the vehicle's geometric center. This system uses more thrusters than DOFs, so the system would be considered over actuated - in reality, having a greater number of thrusters than DOFs can drastically improve authority in multiple directions (Johansen & Fossen 2013). Because of this, we need to solve the system of equations using the pseudoinverse of the matrix to decompose the desired body-frame wrench into individual motor thrusts:

$$\tau = Tu$$

Where

$$T = \begin{bmatrix} f_1 & f_2 & \dots & f_8 \\ r_1 \times f_1 & r_2 \times f_2 & \dots & r_8 \times f_8 \end{bmatrix}$$

- $\tau \in \mathbb{R}^{6 \times 1}$: desired wrench (forces and torques in the body frame)
- $T \in \mathbb{R}^{6 \times 8}$: thrust allocation matrix
- $u \in \mathbb{R}^{8 \times 1}$: vector of individual thruster forces
- f_i : unit thrust direction vector of thruster i
- r_i : position vector of thruster i relative to the center of gravity

The pseudoinverse of the thrust allocation matrix T was used to obtain a minimum-norm solution for u , ensuring that the commanded forces and torques are achieved while minimizing

total thrust effort. This approach allows the control system to distribute loads efficiently among the available thrusters.

$$u = T^+ \tau$$

This solution allows for τ to be an input that describes the desired force on the vehicle, and u gives a positive or negative number for each thruster signifying the required thrust direction to make that happen. Of course, the opposite calculation can be done to determine net forces and moments on the vehicle based on the sum of individual thruster forces and moments (later used to assist a visual simulation of predetermined flight patterns). This provides valuable information regarding the expected authority in different directions when compared to fluid simulation data. The thruster directions necessary for each DOF are assembled into a table and dropped into the “APmotors.cpp” file in the Ardusub firmware directory where the motor configurations are established, allowing for a completely custom thruster configuration. This modification was the only change made to the original firmware and was re-flashed onto the Pixhawk 6C once finalized. The following figures depict a summary of the thruster directions necessary for motion in all six DOFs. Note that arrows depict the direction of force produced by the thruster, which will be opposite from the direction of thrust produced by the thruster. For example, thruster 2 receives a forward PWM signal for a move forward command, pushing water towards the rear of the vehicle resulting in a force in the positive x direction. Inversely, thruster 1 receives a reverse PWM signal, which, due to the mirrored orientation of the thruster, also pushes water towards the rear of the vehicle resulting in a force in the positive direction.

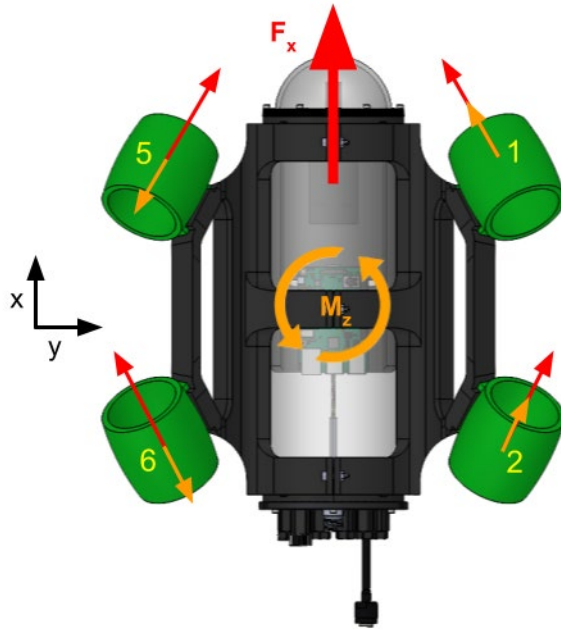


Figure 4.1: Top view of the vehicle with thrust directions required to produce net F_x and M_z

Forward signal to thrusters 2, 4, 6, and 8 and reverse signal to thrusters 1, 3, 5, and 7 result in a net positive forward force F_x . Forward signal to thrusters 1, 3, 6, and 8 and reverse signal to thrusters 2, 4, 5, and 7 provide a net clockwise moment around the z axis M_z (yaw).

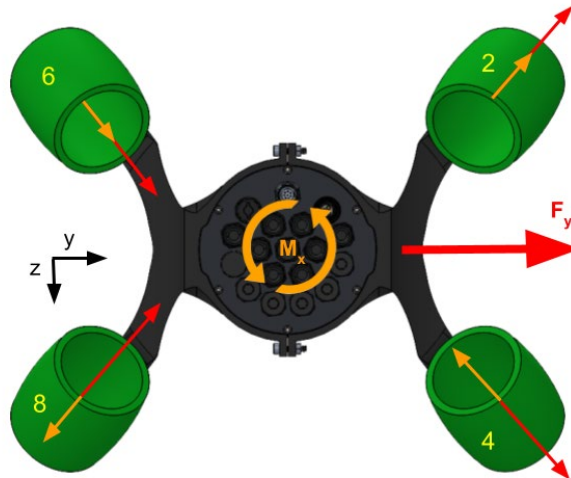


Figure 4.2: Rear view of the vehicle with thrust directions required to produce net F_y and M_x

Forward signal to thrusters 1, 2, 3, and 4 and reverse signal to thrusters 5, 6, 7, and 8 result in a net positive sideways force F_y . Forward signal to thrusters 3, 4, 5, and 6 and reverse signal to thrusters 1, 2, 7, and 8 provide a net clockwise moment around the x axis M_x (roll).

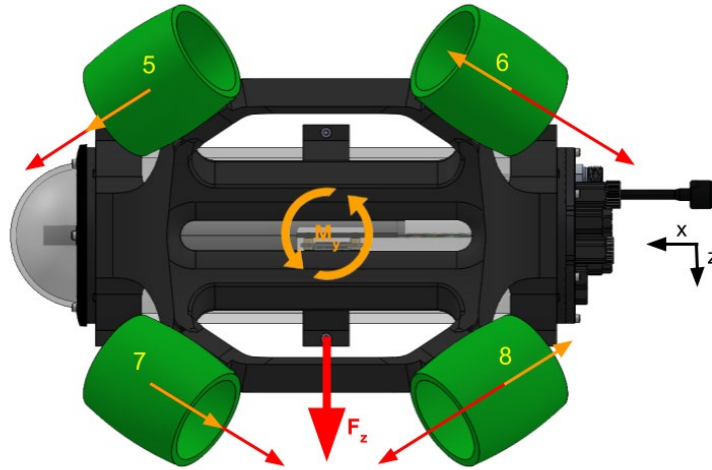


Figure 4.3: Side view of the vehicle with thrust directions required to produce net F_z and M_y

Forward signal to thrusters 1, 2, 5, and 6 and reverse signal to thrusters 3, 4, 7, and 8 result in a net downward force F_z . Forward signal to thrusters 1, 4, 5, and 8 and reverse signal to thrusters 2, 3, 6, and 7 provide a net clockwise moment around the y axis M_y (pitch).

4.3 Fluid Simulations

Simple fluid simulations were conducted in ANSYS Fluent 2023 R1 to determine normal forces due to drag for surge, sway and heave. Drag forces were calculated based only on pressure integration due to lack of accuracy at the boundary layer. A standard 1 m/s was used for all of the simulations.

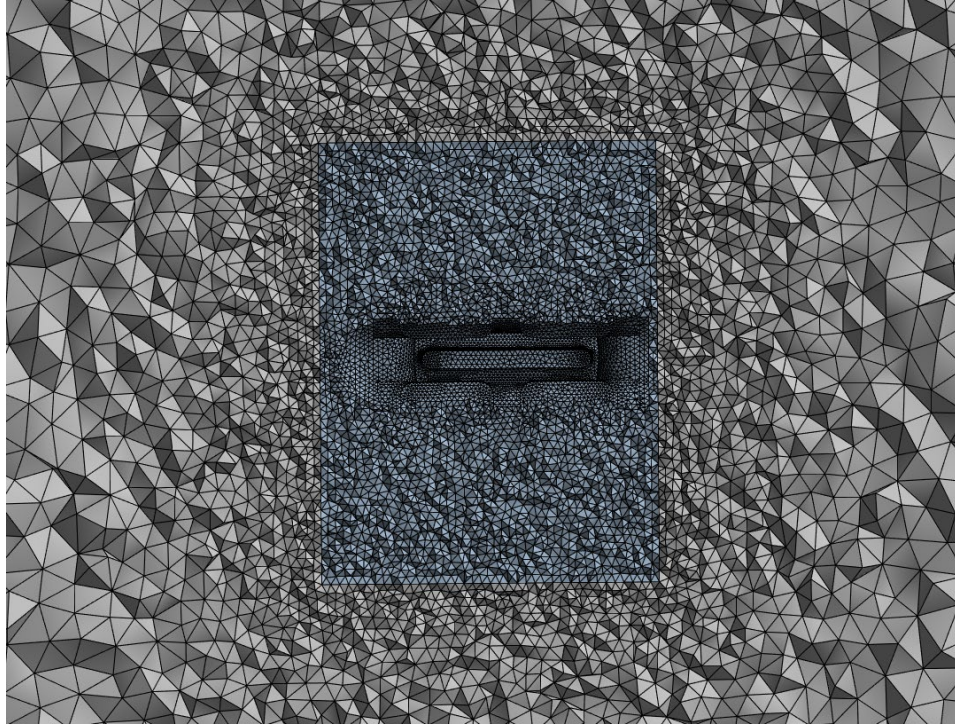


Figure 4.4: Cut plot of final mesh for rolling simulation

For the translation simulations, identical meshes were used for the three different flow setups by simply changing which sides of a cubic fluid volume were the inlet and outlet. Face sizing was applied to the surface of the ROV walls and a small growth rate was used to obtain more accuracy in the immediate proximity to the ROV.

A slightly different approach was needed to simulate the rotating motion of the ROV. Similar to how propellers are typically simulated in ANSYS, a cylindrical rotating fluid volume was constructed as close to the ROV features as possible (see blue area of Figure 5.4). The larger cylindrical volume that encloses this rotating volume is stationary; all outside surfaces are defined as an outlet (since there is no inlet defined, the amount of fluid inside the computational volume does not change). It is worth noting that the most accurate way to model a rotating ROV would be to have the ROV itself move and use a moving mesh, but this is extremely computationally expensive and using a rotating fluid volume achieves sufficient accuracy.

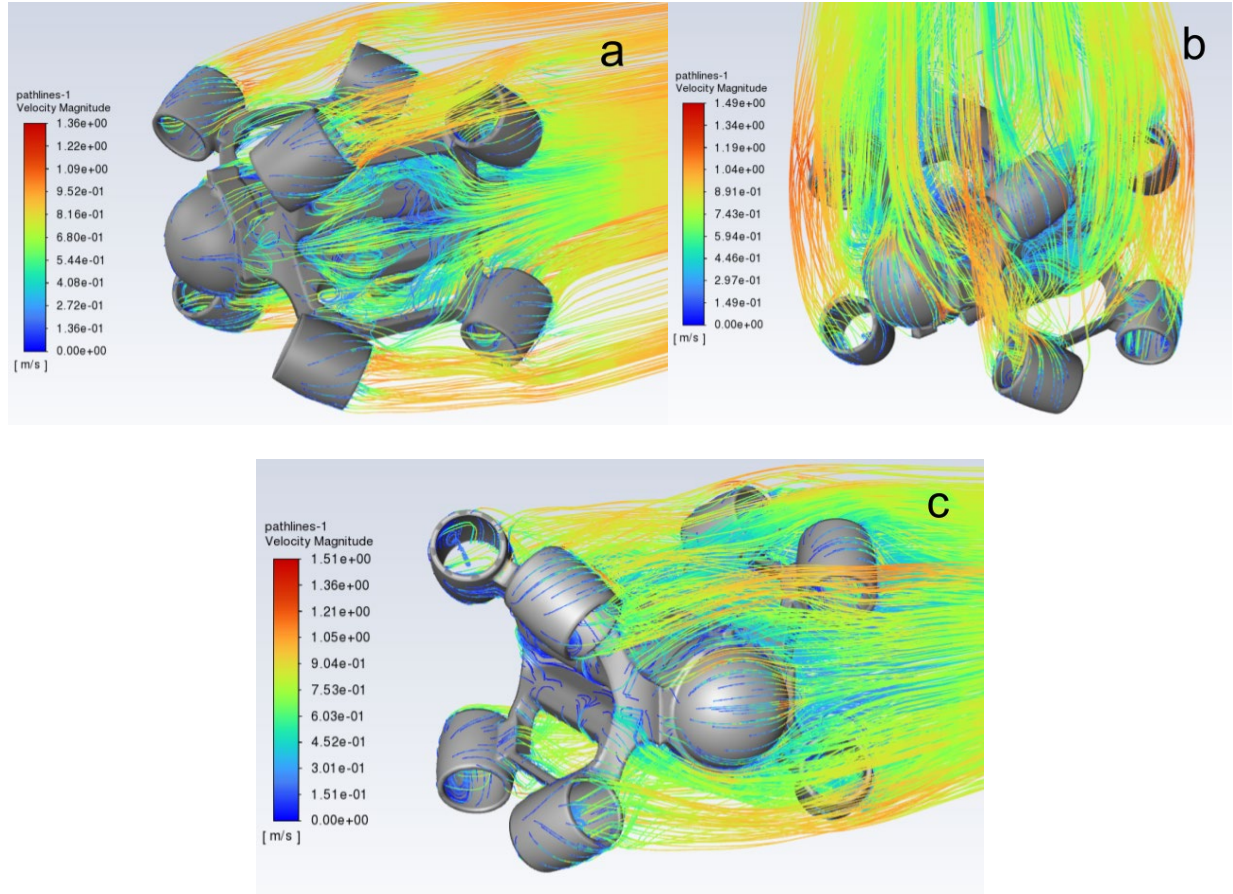


Figure 4.5a to c: Velocity flow trajectories used to calculate normal forces for surge, sway, and heave respectively

Using the calculated drag force, density of water, and fluid velocity, values for C_dA (a constant pulled from the standard equation for drag) are calculated for each translational DOF and substituted back into the drag force expression, this time solving for velocity and using the maximum theoretically achievable thrust by all 8 thrusters in that direction (calculated by using the thruster allocation matrix).

$$F_{thrust,DOF} = F_{drag}(V) = \frac{1}{2} \rho V^2 (C_dA)_{DOF}$$

$$(C_dA)_{DOF} = \frac{2F_{drag}(V)}{\rho V^2}$$

$$V_{max} = \sqrt{\frac{2F_{thrust,DOF}}{\rho(C_d A)_{DOF}}}$$

Table 4.3: Summary of maximum theoretically achievable thrust for translational DOFs and their corresponding maximum velocities

	Maximum Thrust (N)	Maximum Velocity (m/s)
Surge	79.75	1.81
Sway	40.73	1.07
Heave	52.88	1.02

Given that forward motion will still likely be the most utilized DOF, having the best authority in that direction is ideal. Sway and heave, less utilized but still essential to operation, certainly have enough authority for rapid manual adjustments and preprogrammed flight paths.

A similar procedure was used to find the maximum theoretical torque produced by thrusters in the rotational DOFs and the maximum angular velocity produced by that torque. Take a rigid body rotating about its yaw axis with angular speed ω . Consider a small surface element dA at a radius r from the axis. The local tangential speed is $v = \omega r$. If the local tangential drag force on that element follows the usual quadratic drag law,

$$dF = \frac{1}{2} \rho C_{D,local} v^2 dA = \frac{1}{2} \rho C_{d,local} (\omega r)^2 dA$$

Then the contribution of that element to the torque about the axis is:

$$d\tau = r dF = \frac{1}{2} \rho C_{D,local} \omega^2 r^3 dA$$

Integrate over the total wetted area:

$$\tau_{drag} = \frac{1}{2} \rho \omega^2 \int_A C_{D,local} r^3 dA$$

We can lump the constants into a single rotational drag constant term C_{rot} :

$$C_{rot} = \frac{1}{2} \rho \int_A C_{D,local} r^3 dA$$

Giving us a simple equation we will use to calculate the maximum angular velocity based on the rotation drag calculated in ANSYS after solving for C_{rot} :

$$\tau_{drag} = C_{rot} \omega^2$$

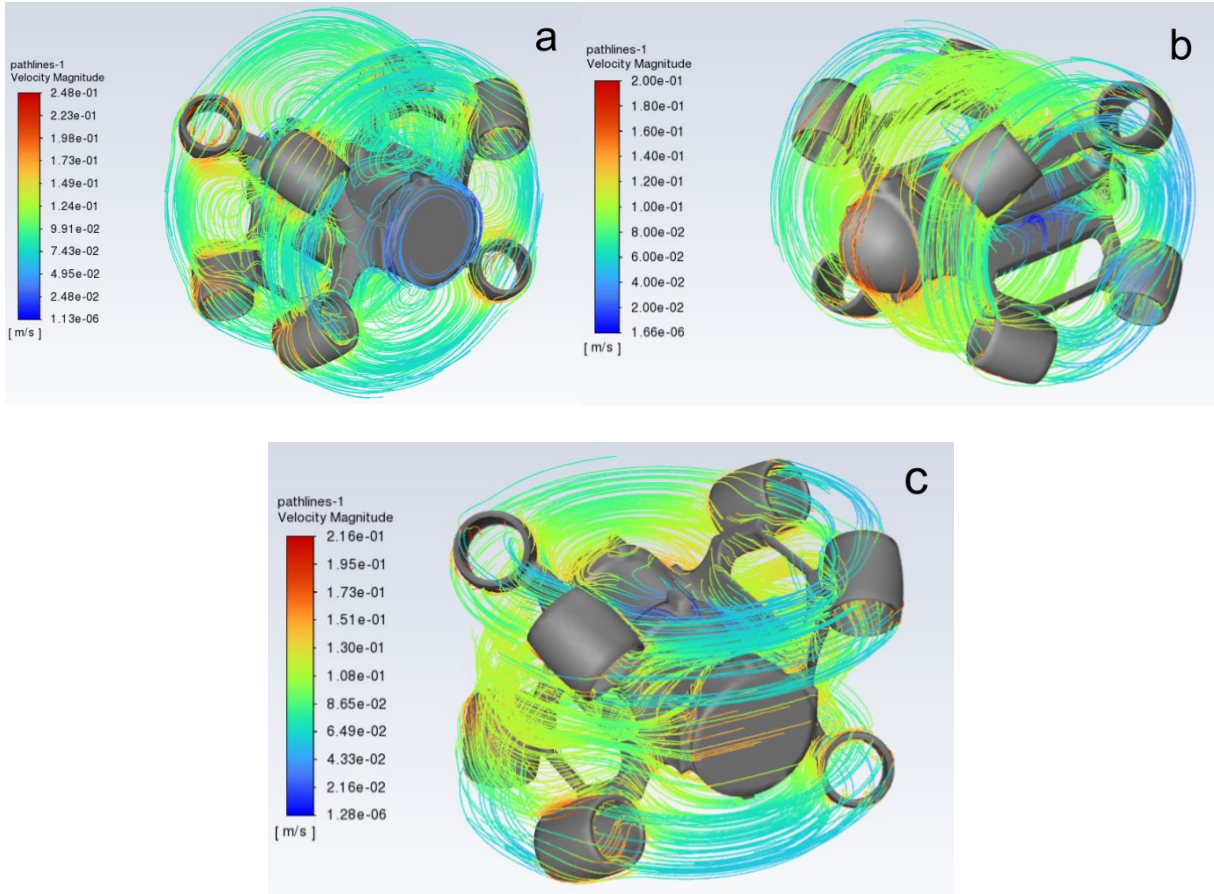


Figure 4.6a to c: Velocity flow trajectories used to calculate moments for roll, pitch, and yaw respectively

Table 4.4: Summary of maximum theoretically achievable thrust for rotational DOFs and their corresponding maximum angular velocities.

	Maximum Thrust (Nm)	Maximum Velocity (rad/s)
Roll	2.68	1.13
Pitch	10.52	3.30
Yaw	12.14	3.34

The platform is designed to have maximum authority in the pitch and yaw DOFs, which will be frequently actuated in the preprogrammed flight paths. Roll still has plenty of authority for stabilization in turbid environments but is not intended to be actuated as often and thus the platform is designed to favor the other two rotational DOFs. These theoretical maximum velocities in translation and rotation were later verified visually during testing to be within 15% of the true velocities.

Chapter 5 SYSTEM DESIGN

Following the theoretical validation provided by simulation, this chapter details the physical realization of the ROV. It covers the mechanical integration of the 3D-printed frame, the selection of electronic components for the control stack, and the software architecture that translates pilot commands into vehicle movement.

5.1 Mechanical Design

The structural frame itself consists of only two identical parts: the right and left halves. These parts were manufactured using a Prusa XL 3D printer, which was capable of printing each of these pieces whole. The advantage of designing the parts to be printed this way is that there are no places for adhesives or fastener interactions to fail - the entire part is as strong as the Prusa is capable of printing. A Prusa PLA was used for all of the printing necessary. This would not be the first choice for a vehicle ready to deploy for ocean use, but is perfectly satisfactory for prototyping and pool testing. The parts are printed vertically with 0% infill and six walls using a 0.6 mm nozzle. The large number of walls gives the parts more than enough structural strength (in combination with the larger nozzle), and the 0% infill allows for drainage holes to be drilled later. Generally, the use of 3D printed parts for aquatic use is avoided because of the inherent issue of waterproofing a semi-hollow part with hundreds of individual cells within the infill for water to get trapped in. By using a thick-walled hollow part, this issue is avoided entirely by accounting for water ingress and allowing the water to drain when pulled out. The assembly of the frame requires the enclosure to clamp around using six M4 bolts. The advantage to a design like this is the ease of switching out the frame with an entirely new one. In the future, the frame will be larger to accommodate more powerful thrusters and larger batteries - the enclosure can be

removed from the current frame and simply dropped into the new one with the same functionality.



Figure 5.1: CAD model of entire ROV with numbered thrusters

All non-waterproof electronics are housed inside a 4" acrylic enclosure manufactured by Blue Robotics. The front end is sealed by their domed end cap, which allows for the large 140° field-of-view to be fully taken advantage of. Both of these components are rated up to 500m depth, which are the limiting factors to the theoretical operational depth of the entire vehicle (without the GoPro). The other end of the acrylic tube is sealed by an aluminum end cap with 18 M10 holes. Sealing the interface between motor wires, tether, and external sensors are the Blue Robotics WetLink Cable Penetrators. A total of 9 of these are used for the motor wires and tether, with the remaining 9 M10 holes populated with a pressure relief valve, vacuum port, depth sensor, power switch, and plugged penetrators. The five plugged penetrators could later be replaced with WetLink ones to be used for lights, acoustic positioning sensors, an external battery enclosure, etc.

5.2 Electronics and Hardware

The electronics integration of this ROV follows closely to the documentation provided by the open-source project Ardusub, using a Pixhawk for running the firmware and a Raspberry Pi companion computer running BlueOS. Manual joystick controls are sent using a standard Xbox controller to a surface computer using either QGroundControl, Mission Planner, or Cockpit. These ground station softwares decode the joystick inputs into MAVLink control messages which are sent directly to the Raspberry Pi over a Cat5e ethernet cable capable of full gigabit transfer speeds. It is worth noting that this system does not utilize signal extenders such as the Fathom boards from Blue Robotics - these would be necessary for tether lengths greater than 100m, but since this revision of the platform is intended only for pool use, they are not needed. The Raspberry Pi handles video processing from the 1080p USB camera and sends the video feed right back to the surface computer with a measured delay of about 100 ms.

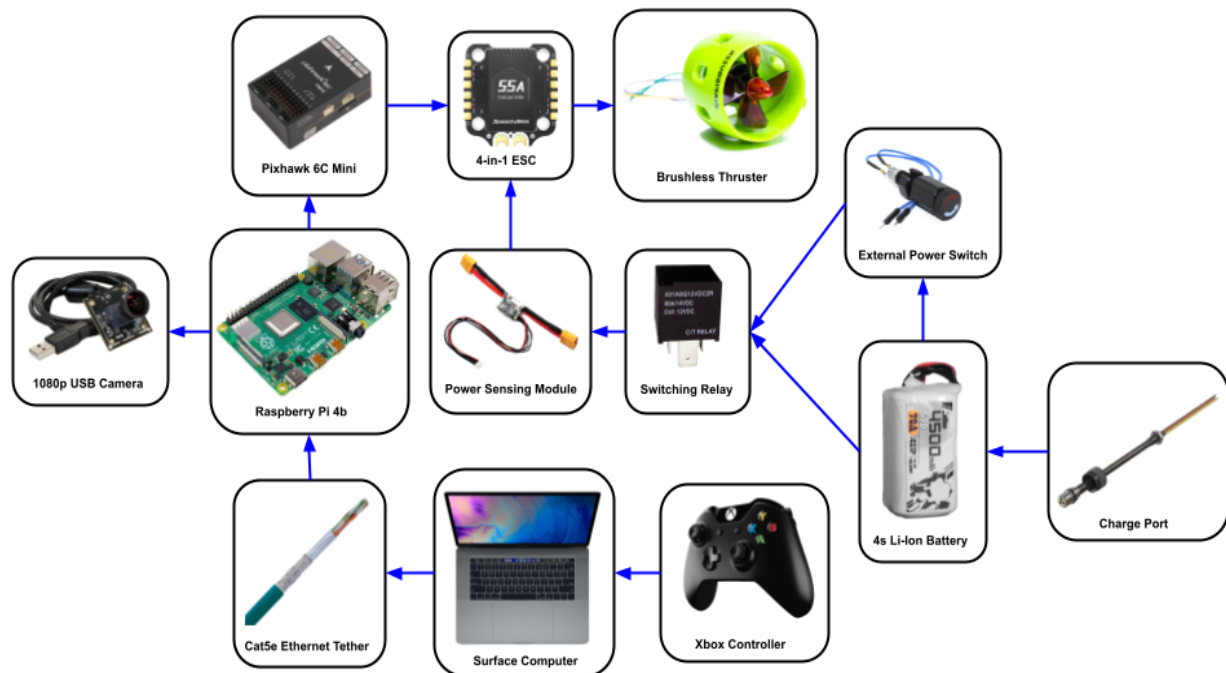


Figure 5.2: Vehicle electronics ecosystem

The Raspberry Pi communicates with the Pixhawk autopilot board over USB, which runs the actual ArduSub firmware and dictates motor mixing, rates, and compiles sensor data. Motor PWM signals are sent to two 4-in-1 55A 4-in-1 electronic speed controllers designed for aerial drones - these allow for a much more compact system than using 8 individual speed controllers. Each ESC connects to the 12 wires corresponding to their respective half of the brushless thrusters through 12-pin Molex connectors. The thrusters themselves are Apisqueen U01s, a cheap brushless motor/enclosure system with plenty of thrust for this version of the ROV. Current draw headroom was left for these to be upgraded in the future to more powerful thrusters such as the Blue Robotics T200s.

Power for the entire system is supplied by a 4s 4500 mAh lithium-ion battery. An external power switch combined with a high current automotive relay is used to control power delivery. The relay is switched using battery voltage by completing a circuit using a Blue Robotics switch that also acts as a plugged penetrator. With the relay in the off position, the battery is able to accept charge through a Blue Trail Engineering Cobalt series connector. Both the external switch and charging port exist in order to maintain full functionality of the platform without opening the enclosure to plug in the battery, resulting in fast deployment and redeployment times.

5.3 Software Architecture

As stated in the previous section, there are two components to the software architecture: the operating system running on the Raspberry Pi and the firmware running on the Pixhawk. ArduSub is a low-level control firmware built on the Ardupilot stack. It handles sensor fusion (IMU, depth sensor), motor mixing (translating joystick commands into individual thruster commands), PID control loops, and MAVLink communication with the higher-level systems.

The entire Ardusub firmware is completely open source, allowing for modification to any part of the original directory. Ardusub offers a number of different pre-programmed motor configurations that tell the firmware which motors need to actuate in order to achieve movement in a desired DOF. Since a custom design is used for the frame here, it was necessary to construct a new motor mixing matrix for this particular design. After adding the custom motor configuration and recompiling the firmware, the new firmware can be flashed back onto the Pixhawk using Mission Planner. In the future, larger changes could be made to the firmware such as adding custom flight modes for the preprogrammed flight paths that will be discussed later.

The Raspberry Pi runs Blue Robotics' own open-source operating system called BlueOS. This system creates a web address so that communication can be established between the surface computer and this companion computer. It also provides a MAVLink router for communication between the Pixhawk, ground control software, and any other add-ons. Finally, it decodes the live camera feed and sends it back to the surface computer for real time viewing in addition to any sensor data provided by the Pixhawk.

The surface computer will be running the user's preferred ground control software - most commonly Cockpit, Mission Planner, or QGroundControl. For this platform, QGroundControl was used mostly for its simplicity but any of the three will serve the same purpose. This software is responsible for receiving and displaying telemetry data (attitude, depth, battery voltage, etc.) as well as the camera feed. It also sends control commands from the joystick via MAVLink packets and is used to configure Ardusub parameters. Some parameters include the PID values for tuning control behavior, selection of sensors used, frame configuration, and many more. ([link](#)

all parameters). The following is an example of how a command gets sent, received, and executed with the above infrastructure:

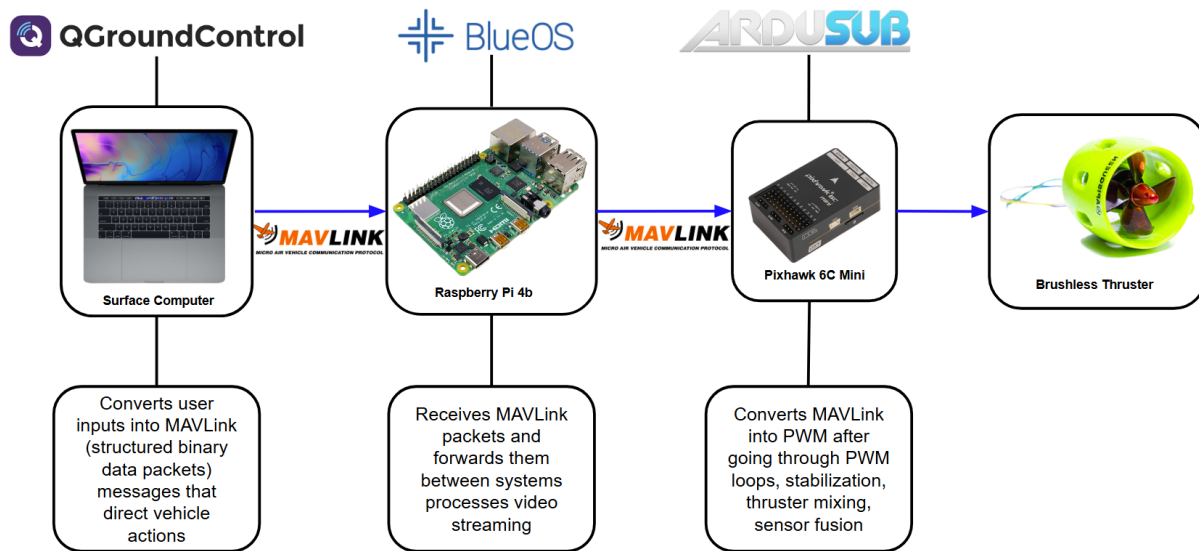


Figure 5.3: Data path from user input to actuation

Chapter 6 AUTONOMOUS OPERATION FRAMEWORK

This chapter outlines the development of the control scripts and automated flight modes that enable the vehicle to perform precise underwater maneuvers. Using the pymavlink library and custom Python scripts, the framework allows for the repeatability and accuracy required for precise photogrammetric data collection.

6.1 Flight Mode Customization

There are a number of different ways to add programmed flight parameters to an ROV platform with this software architecture. The most straightforward method which was used for this platform is pymavlink. Similar to how the ground control software transfers data, pymavlink outputs MAVLink commands via a User Datagram Protocol (UDP) port which is read by the Raspberry Pi (specified by its IP address). Pymavlink can be used to send essentially all of the same commands that the ground control software can provide, but here they can be automated with high precision.

Pymavlink can be run on either the surface computer or on the companion computer itself. Using pymavlink on the surface computer is useful for pushing messages at the user's discretion, while running pymavlink on the companion computer would be better suited to systems that need to reference sensor data with low latency or that are capable of onboard autonomy. The platform built for this thesis is focused more on a proof of concept than a mission ready machine and serves to demonstrate that the fully autonomous operation will be possible in the future for photogrammetric scanning. Given this use case, a simpler control system of running pymavlink on the surface computer to send pre-programmed movement commands is implemented.

Since the main objective of this platform is to perform photogrammetric scans, the focus for the pymavlink scripts was to command the ROV in a variety of movement patterns. An example script looks like this:

1. Establish master connection to the ROV over UDP
2. Set flight mode
3. Arm the vehicle
4. Dive to specified depth
5. Run a control loop a specified number of times
6. Move forward for 10 seconds
7. Turn +90 degrees from initial heading
8. Move forward for 3 seconds
9. Turn +180 degrees from initial heading
10. Move forward for 10 seconds
11. Turn -90 degrees from initial heading
12. Stop movement
13. Disarm the vehicle

The script is executable through the terminal on the ground control computer. All scripts are stored in a folder, and once inside that directory simply typing “python {script name}” will start the communication between the surface computer and the Raspberry Pi. The MAVLink messages must be sent over a different UDP port than the one that the ground control software uses to communicate with the vehicle, otherwise issues arise with which source is able to

maintain control of the vehicle. Operating on the same port will cause dangerously inconsistent thrust commands to the vehicle, resulting in violently twitchy thrusters that offer less precise control and are likely prone to mechanical failure.

Because the current version of this platform does not have more advanced positioning sensors, the autonomous operation of the vehicle is limited to an open-loop system. The ROV is able to continuously reference its attitude and depth and use the PID control to maintain given values, but it is not able to move to specific waypoints or record exact positional data. This functionality will be necessary for future revisions; photogrammetry data needs to be aligned with global positioning in order to contextualize the scans. It will also be necessary for higher-level autonomy and more accurate lawnmower patterns as well as more complicated patterns such as circling a focal point.

6.2 Photogrammetry Workflow

All photogrammetry data is recorded from a GoPro Hero 13 Black mounted to the bottom of the ROV in timelapse mode. The GoPro takes pictures in 1 second intervals at 5.3k resolution with the Wide FOV setting and a fixed white balance. The GoPro can be angled straight down for a traditional lawnmower pattern, angled slightly forward, or angled completely forward for hull scanning or orbiting an object. Once the ROV mission is completed, the images are offloaded from the onboard SD card uploaded to a much more powerful computer running the program Agisoft Metashape, a dedicated commercial platform for photogrammetry reconstructions (Agisoft LLC 2025).

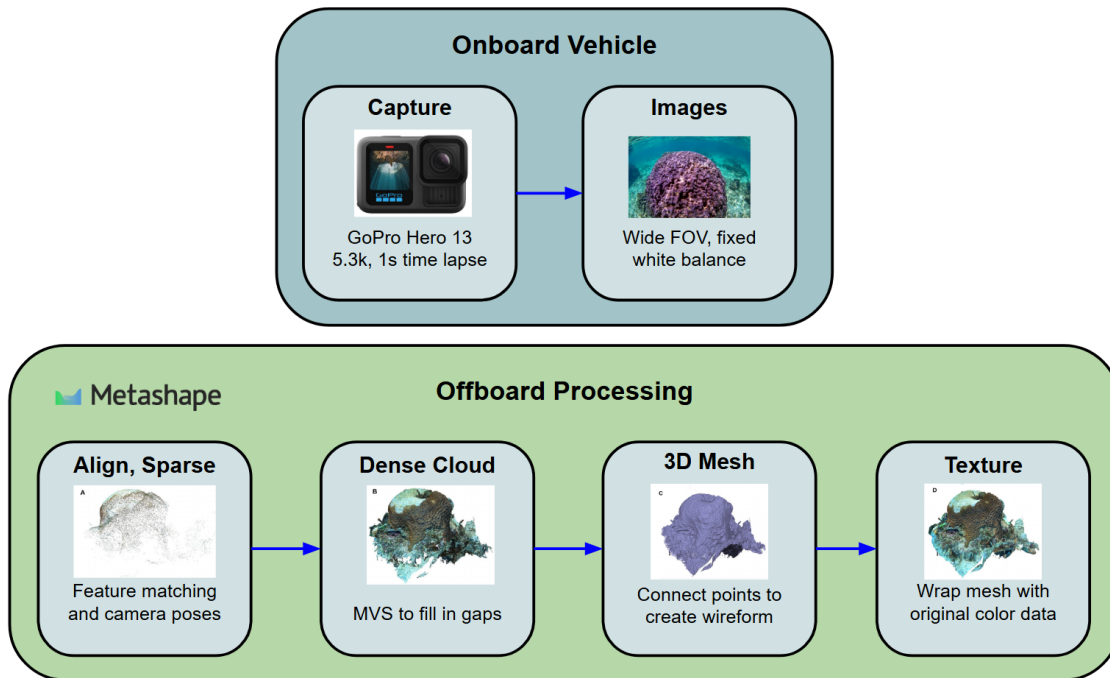


Figure 6.1: Onboard and offboard photogrammetry workflow

Chapter 7 EXPERIMENTAL SETUP AND TESTING

This chapter describes the procedures used to validate the ROV's performance against the initial design goals. By conducting a series of controlled tests in a pool environment, the platform's stability, response to autonomous commands, and effectiveness in capturing multi-angle photogrammetric data were evaluated.

7.1 Final Assembly

The ROV was assembled according to the previous parts list and CAD. The frame was printed, drainage holes were drilled, and the thrusters were bolted onto their mounting locations. Because the WetLink Cable Penetrators are designed for use with cables of a standard diameter, it was necessary to extend the thruster motor wires with Blue Robotics' PUR subsea cables to ensure proper fit with the cable penetrators. Those splice locations were the source of leaks and required a more robust waterproofing - two-piece 3D-printed shells were bolted around the splice locations and flooded with epoxy.

The rest of the assembly was as simple as wiring the components according to their connection diagrams, making sure thrusters are spinning in the correct directions, and tuning the buoyancy to be slightly positive using small dive weights and foam. Lastly, a GoPro mount allowing for straight-down and forward camera angles was designed, 3D-printed, and fastened to the underside of the ROV as the primary photogrammetry sensor. In the future, the camera could be integrated with the onboard computers to store and send images to the topside computer in real time but for this proof of concept design a GoPro taking pictures at regular intervals is more than enough to obtain full photogrammetry reconstructions.

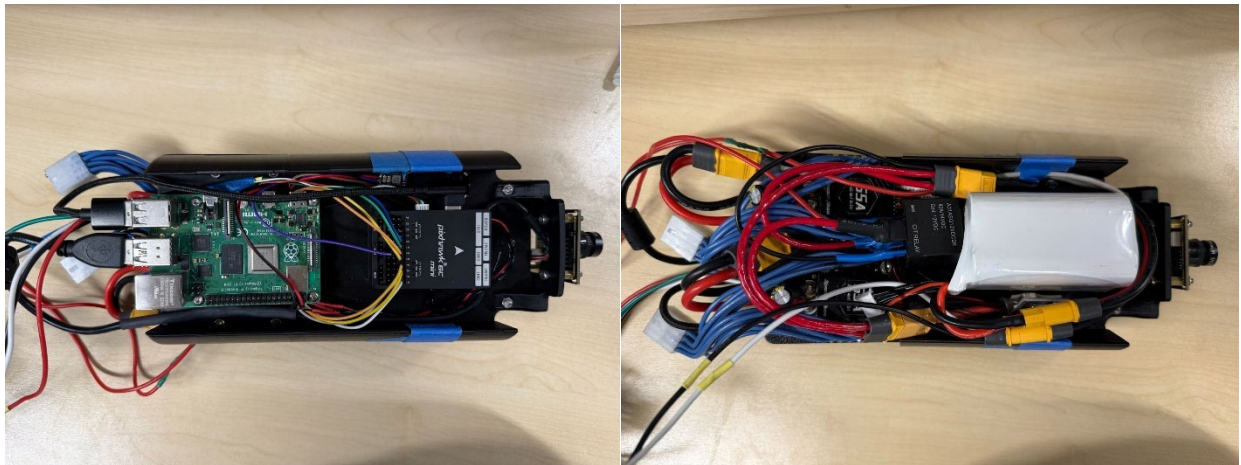
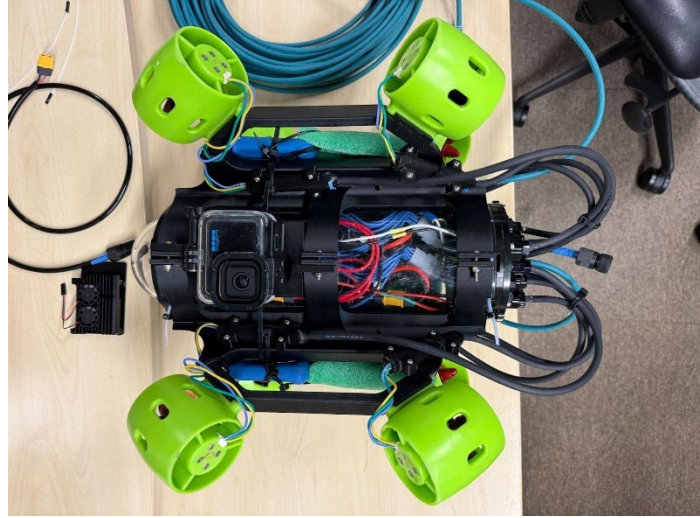


Figure 7.1: Final assemblies of the ROV frame and electronics sled

7.2 Test Setup

First tests were conducted in Ardusub's manual mode, verifying that the intended DOFs are fully actuatable and correspond to the correct joysticks on the controller. Next, stabilize and depth hold flight modes were verified by pushing the ROV at different locations using a long implement and observing its reactions. As expected, the ROV was able to hold its depth reliably and maintain level flight with ease.

The main tests conducted for this study rely on the python scripts previously generated in order to verify accurate control in six DOFs, as well as autonomous photogrammetry capability.

A simple python script was written to actuate each DOF individually - for the translational DOFs, the vehicle is commanded for a set number of seconds in the positive and negative directions. For the rotational DOFs, the SET_ATTITUDE function in pymavlink was utilized to set specific angles for yaw, pitch, and roll. The accelerometer data can be compared to these set angles to determine rate of convergence and accuracy at achieving those angles.

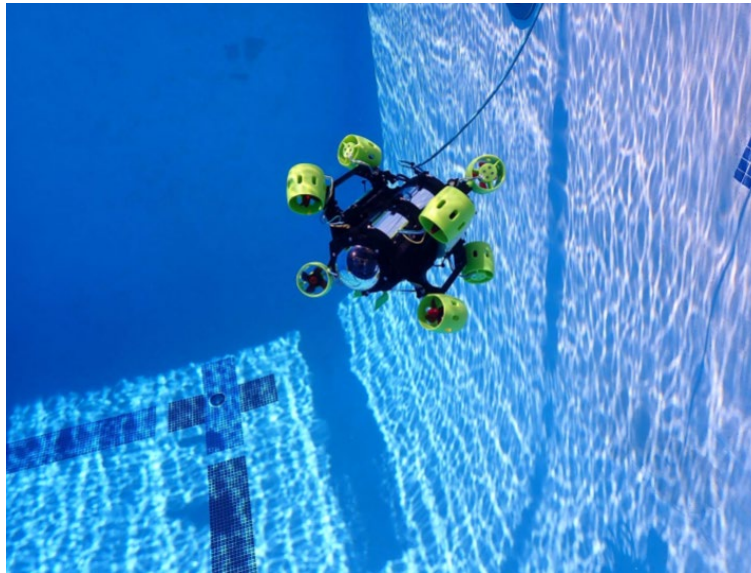


Figure 7.2: Initial testing of the platform verifying proper control in multiple DOFs

For the photogrammetry section of the testing, multiple scripts were written to test functionality for different mapping scenarios. Firstly, a script that commands the ROV forward for a user-inputted number of seconds (row length), rotates by 90 degrees to its initial heading, another forward command (row width), rotates 90 degrees, forward again by row length, etc. In this script, the ROV always moves in the direction it faces. Another script was devised that works in a similar fashion, but instead of yawing to a heading between rows it simply moves laterally between rows. With this script, the ROV always faces in the same direction, moving forward, sideways, backward, etc. Finally, a third photogrammetry script was written to do scans of a vertical surface - descending to a set depth, moving laterally, ascending to a set depth, etc.

Another test would be conducted manually without a script to test a theoretical orbit path for scanning cylindrical or smaller objects.

In order to determine the effectiveness of these scripts, various props were staged in a pool as features for the photogrammetry scans to capture. This included a plastic lawn chair and three large boards with various definable features attached. Using a photogrammetry post-processing software called AGI Metashape, the images captured by the GoPro can be reconstructed into full 3D models and we will be able to view the spatial locations of each picture, allowing us to determine the accuracy of the pre-programmed patterns.

Chapter 8 RESULTS

This chapter presents the quantitative and qualitative findings gathered during the experimental phase of the study. By analyzing the data from thruster characterization, stability tests, and the final 3D reconstructions, the performance of the vehicle is evaluated against the design constraints to determine the success of the six-DOF implementation.

8.1 Angle Convergence

Using log data generated by Ardusub, we can observe the attitude of the vehicle in real time and compare it to the PWM signals being sent in order to check the operational effectiveness of each individual DOF. Surge, sway, and heave were verified visually - this version of the vehicle is not equipped with the more advanced positional sensors that are required to determine real time velocity. Roll, pitch, and yaw, however, are accurately measured by the Pixhawk and recorded as .bin files. From the .bin files we can extract raw data such as battery voltage, attitude, and PWM signal which are all necessary for determining angle convergence and thrust generation. Using a thrust curve provided by the thruster manufacturer, it is possible to determine the thrust produced at any given moment based on the PWM signal and the battery voltage. By running individual thrusts through the previously described thruster mixing matrix, we calculate net forces in all DOFs.

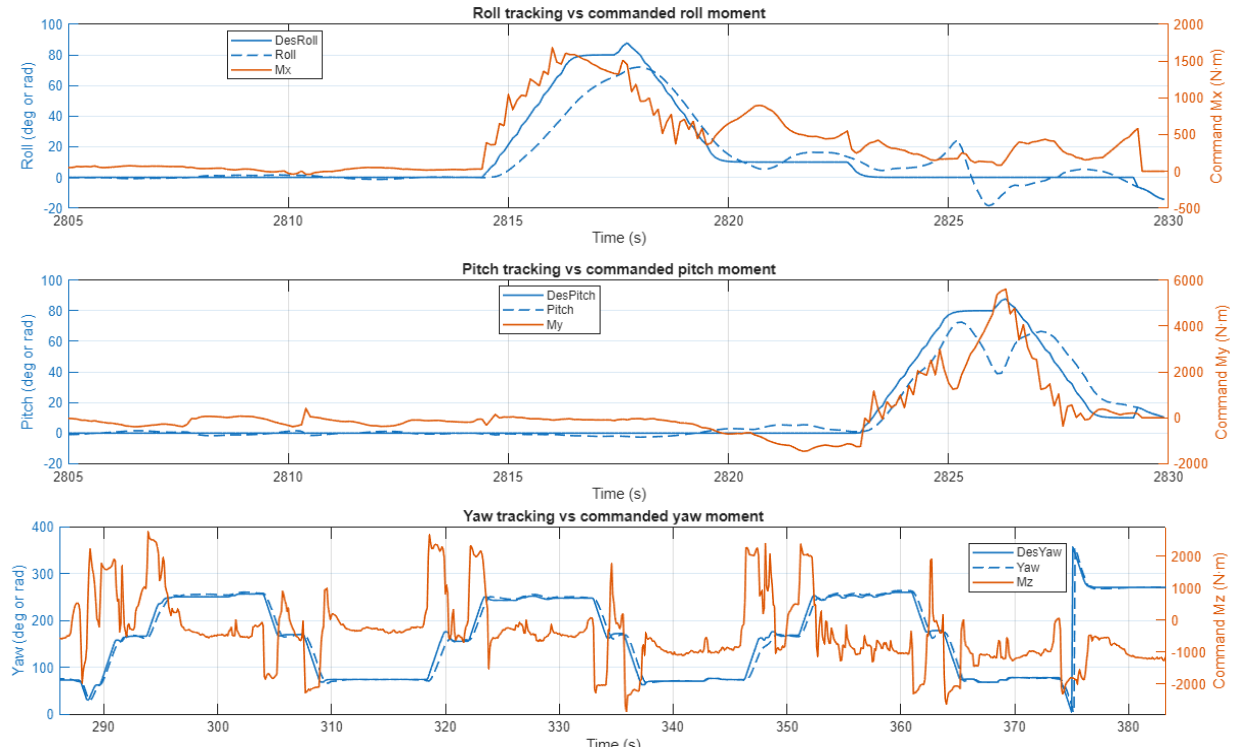


Figure 8.1: Attitude and commanded moments compared over time for rotational DOFs

By running the python script commanding the ROV to rotate exactly by a certain number of degrees, the resultant moments can be plotted with the actual observed attitude of the vehicle to get a better understanding of how the vehicle is utilizing its authority to reach the desired attitudes. The first command rolls the ROV to 90° , holds for a moment, then returns to level flight. The first plot of Figure 7.1 shows the ROV rotating to 88° , very close to the intended attitude. The second plot also shows good convergence to the intended 90° angle, reaching 89° before returning to 0° . The last plot from Figure 7.1 captures the yaw command and response for the lawnmower pattern that utilizes yaw to move between rows. The ROV yaws exactly 90° , moves forward for a short or long period of time depending on if it is running down the length of a row or moving between rows, and yaws by another 90° , alternating directions depending on if the ROV is positioned at the beginning or end of a row. The jagged behavior visible in the

commanded moments for the three plots is as a result of the default PID loop in ArduSub continuously correcting the vehicle towards its commanded attitude.

The plots from Figure 7.1 prove sufficient convergence to commanded angles in any of the rotational DOFs which will allow the ROV to maintain specific orientations for different photogrammetry programs (e.g. gradually pitch the platform down as it orbits an object on the sea floor, resulting in a dome-shaped pattern). Pitch and roll are measured using the gyroscope and accelerometer on the Pixhawk, requiring the integration of angular rate measured by the gyro and corrected by the accelerometer. These are measured relative to the gravitational force on the ROV, which can be less accurate than using a magnetometer as an absolute heading reference for yaw. As a result, the angle measurements are not exact - the expectation is for them to be $\pm 3^\circ$ from the intended value to get full functionality from the platform. Because yaw relies on the magnetometer, the ROV is able to reach the exact commanded angles (provided that the compass has been properly calibrated and there is limited interference from the environment).

In addition to the verification of angle convergence in the three rotational DOFs,

8.2 Pattern Thrust Verification

After confirming that the vehicle can accurately reach the angles it is commanded to, the next step is to verify that the rotations can be combined with translation movements to create functional photogrammetry patterns. After retrieving the PWM data being sent from the Pixhawk to the thrusters as well as the battery voltage data, the PWM signal can be converted to individual motor thrusts, and finally into net thrusts after passing the values through the thruster position/orientation matrix. The result is net forces in any of the six DOFs at any given time frame during the data logging. This will be used to prove that the onboard computers are commanding the intended flight paths to the vehicle. The first pattern and likely the most useful

in a standard photogrammetry application is the “yaw-to-angle” horizontal lawnmower pattern. Figure 7.2 clearly depicts the longer forward movement, followed by a yaw command to rotate 90 degrees, etc. The pattern shows the yaw command flipping directions when moving between rows on opposite sides of the pattern.

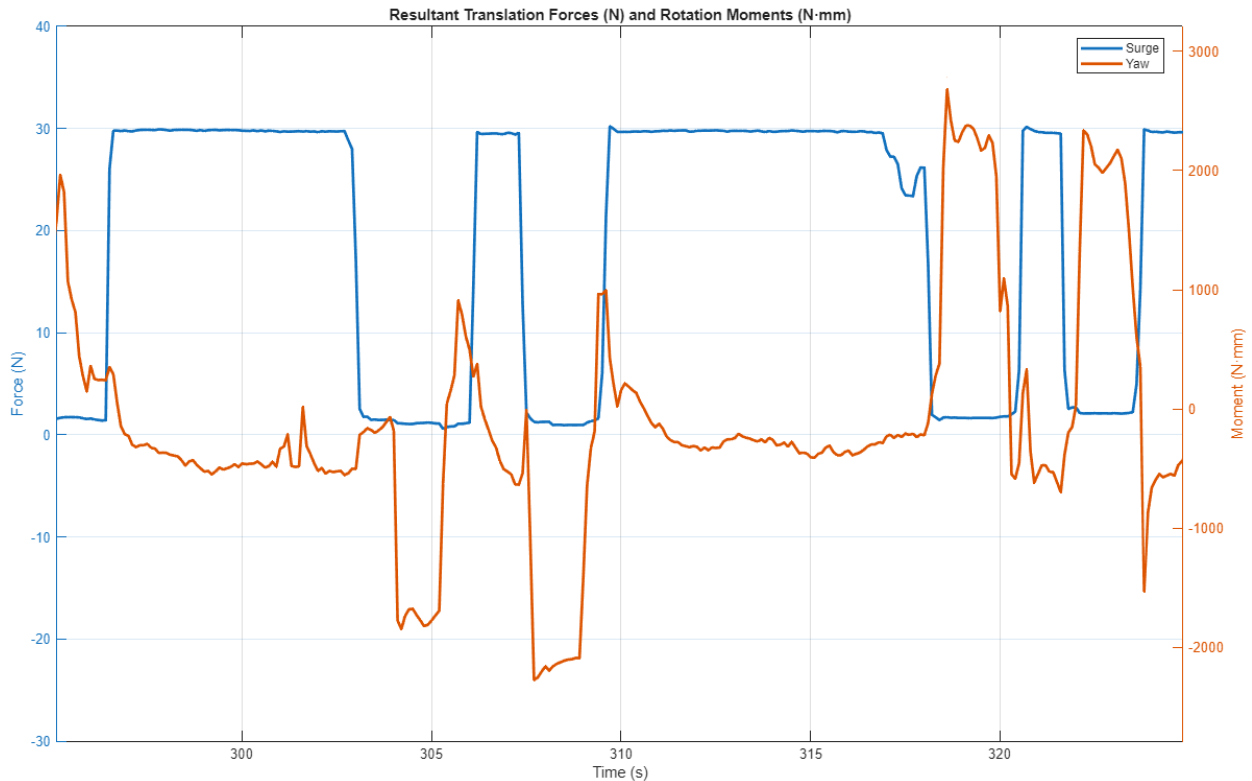


Figure 8.2: Resultant forces on vehicle during horizontal lawnmower pattern (yawing to heading between rows)

The next set of data comes from the horizontal lawnmower pattern that commands the vehicle laterally between rows as opposed to turning to specific headings. Clearly visible in Figure 7.3 are the clean and steady thruster outputs commanding the vehicle in the intended directions. As expected, surge has almost double the available thrust as sway. Because there are no other positional sensors located on the vehicle, visual inspection verifies that the vehicle performed the intended movements: forward, lateral right, backward, lateral right, etc. Of note:

although the same thrust is applied in reverse as forwards, the vehicle tended to move backward roughly 15% farther. This is likely due to the vehicle pulling on the tether as it moves forwards and encounters less resistance in reverse.

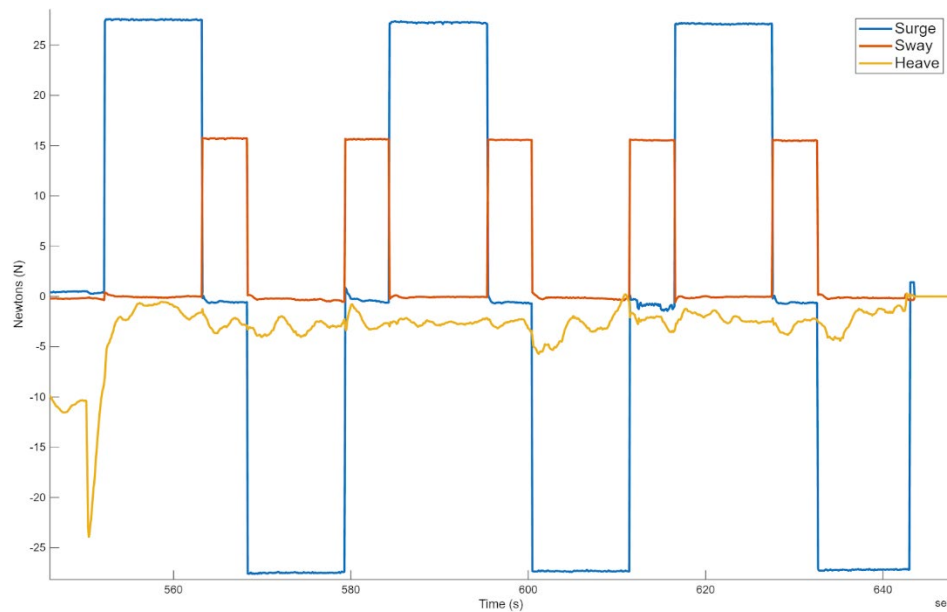


Figure 8.3: Resultant forces on vehicle during horizontal lawnmower pattern (vehicle moving laterally between rows)

The final lawnmower pattern is the vertical pattern, which commands the vehicle in vertical columns moving laterally between them. Again, the thrust data in Figure 7.4 shows exactly the net forces that were intended by the script - the vehicle moves downward, holds for a moment, moves upward, moves laterally to the next column, etc. In testing, the tether also caused some disturbances - weight from the tether caused the vehicle to gradually drift backward as it traveled up and down (this is also clearly visible in the photogrammetry section). It is worth noting that the shape of the heave force curves appears more angular than the flat sections for the other translational DOFs because of the script actually setting a depth and relying on the PID loop to send motor controls (as opposed to sending purely PWM signal for surge and sway).

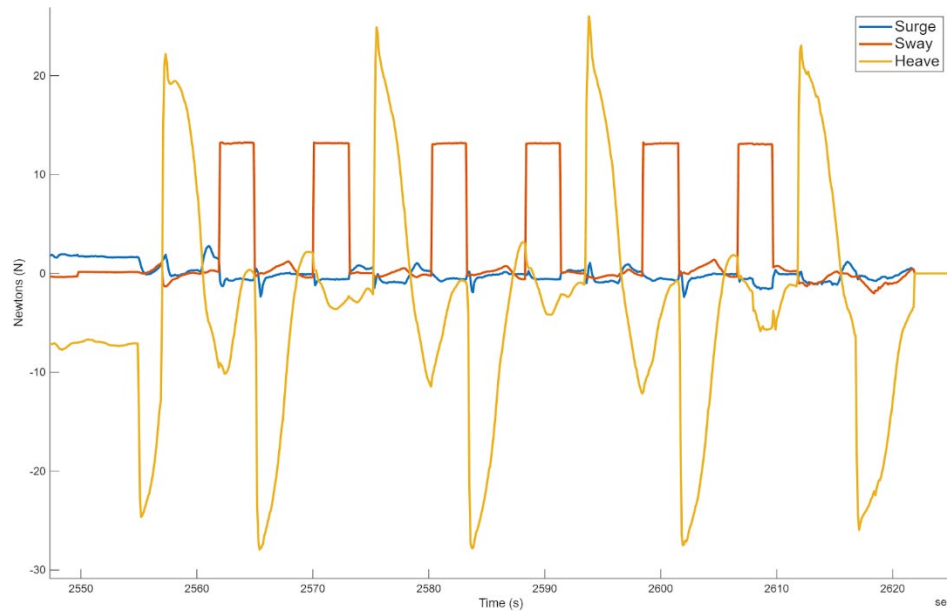


Figure 8.4: Resultant forces on vehicle during vertical lawnmower pattern (vehicle moving laterally between rows)

8.3 Photogrammetry

The final pool testing for the vehicle was conducted with a GoPro Hero 13 Black mounted to the underside on a swivel mount allowing for both forward- and downward-facing images. Using a fixed white balance and 5568 x 4872-pixel resolution, images were taken one second intervals for the duration of the testing. A combination of vertical and horizontal lawnmower patterns was used to capture the majority of the images, as well as a manually-controlled orbit maneuver around a prop chair for the purpose of assessing finer detail capture.



Figure 8.5: Comprehensive photogrammetry scan of pool testing area

A total of 3,120 images across multiple dives in a pool were aligned to construct a final textured 3D model in Agisoft Metashape Professional. Medium to high settings for the point cloud, depth maps, mesh construction, and texturing were used to obtain a usable, detailed model of a large corner section of the pool.

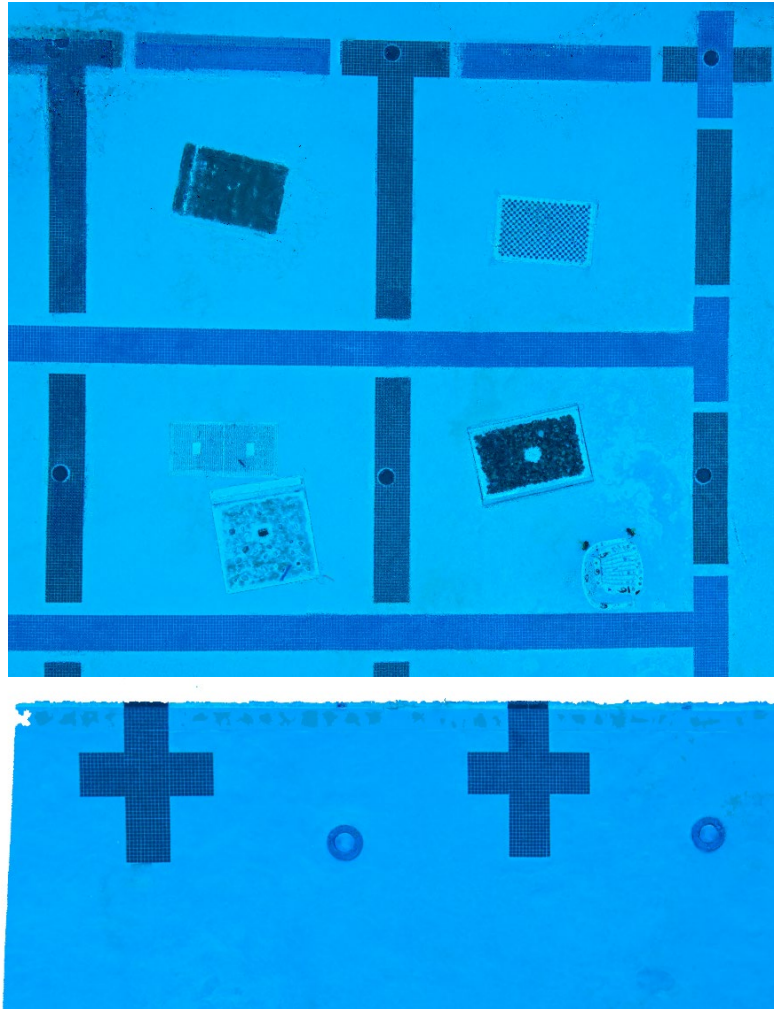


Figure 8.6: (From top to bottom) Model sections reconstructed using horizontal and vertical lawnmower patterns

The distance between the vehicle and the pool floor averaged between 5 and 10 feet, and distance to the captured wall was 5 feet or closer. As a result, finer details are visible in the side facing scans than the downward ones. The accuracy of the model tended to be higher near the surfaces that the vehicle was closer to, which was apparent when filtering out lower accuracy points in the point cloud during reconstruction.

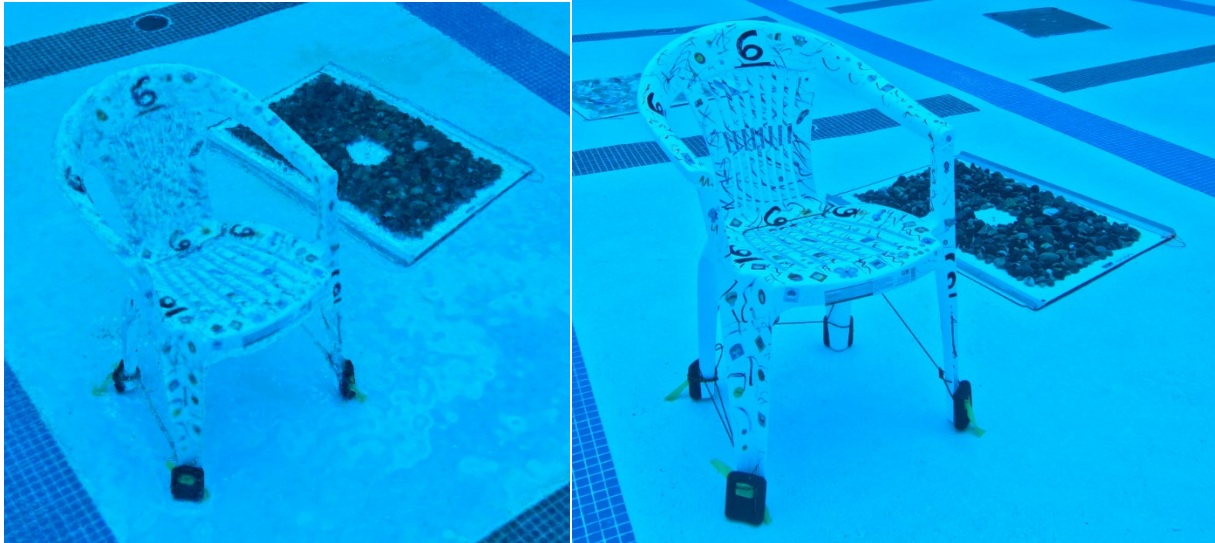


Figure 8.7: (From left to right) Close-up of prop chair in reconstruction compared to raw image from the onboard GoPro used to construct the model

To test the capability of the vehicle performing a more complicated maneuver, the ROV was manually piloted around a plastic lawn chair with various defining features - starting horizontal, and gradually pitching the forward-facing GoPro downwards as it circled higher above the chair. Figure 7.7 shows a screenshot of the 3D model itself next to a GoPro still frame. The resolution of the chair is high enough to clearly read the number written in multiple places, as well as capturing even finer details such as the cables fastening the weights to the chair.

Chapter 9 CONCLUSION AND FUTURE WORK

The final chapter summarizes the contributions of this research in the field of underwater photogrammetry and robotic maneuverability. It reflects on the effectiveness of the developed platform, discusses the limitations encountered during testing, and proposes future improvements for enhancing the vehicle's autonomous capabilities and depth-rating.

9.1 Summary of Contributions and Key Findings

The goal of achieving a submersible vehicle with six DOFs and semi-autonomous control was achieved. The final product is a reliable platform capable of being deployed in less than five minutes and piloted manually using intuitive controls while also having the ability to execute simple photogrammetry patterns that make use of the limited sensors available. The entire system is fabricated using off-the-shelf components at a significant discount compared to similarly-capable commercial options, and with proper guidance could be assembled and operated by individuals with little robotics and marine engineering experience. It was proven that the vehicle can accurately rotate to commanded attitudes, a crucial characteristic for a platform whose use case is mainly to replace divers that would be conducting a similar task. As a platform that can follow predetermined paths more accurately than a human diver, underwater scans can be completed more efficiently and with fewer camera positions, reducing processing time without subtracting from the accuracy and detail of the resultant photogrammetric reconstruction.

9.2 Limitations

While the vehicle designed and tested for this study successfully achieved its purpose, there are many areas that could benefit from further improvements. The first and most foremost limitation is the sensing capability of the platform. Currently, the ROV relies strictly on the

gyroscope, accelerometer, magnetometer, and depth sensor to determine its positioning. In order to produce accurate and exact lawnmower patterns and to spatially contextualize the models they provide, additional positioning sensors would be required.

Implementation of a Doppler Velocity Log (DVL) would enable measurement of translational velocity relative to the seafloor. A DVL uses multiple acoustic beams pointed downwards at known angles which reflect off of the bottom surface, recording the Doppler shift of the returned signal (Alexandris et al. 2024). Combining the beams results in a 3-axis velocity vector in the vehicle frame. This would allow for further optimization of the image capturing required for photogrammetry reconstruction - tuning the camera capture rate and the exact velocity of the vehicle in order to hit a sweet spot of image overlap.

A second sensing system can be combined with DVL to achieve absolute positioning. Ultra-Short Baseline (USBL) positioning provides absolute position fixes of the ROV relative to a surface reference point whose position is known via GNSS (Luo et al. 2020). The system relies on a surface transceiver with a known GNSS position, and an acoustic transponder mounted on the ROV itself. The surface unit measures the time of flight and phase difference across a small hydroponic array of an acoustic signal heard by the transponder to determine the range and bearing of the ROV. Using USBL, the vehicle would be able to move to exact waypoints instead of the current dead reckoning method used, allowing for precise patterns that would easily surpass the capabilities of human divers, provided the equipment is calibrated and functioning properly.

Another area for improvement of this platform would be the quality of components used - in particular, the thrusters. With an emphasis on prototyping and creating a fully functional proof of concept, cheap brushless thrusters were used and could be replaced by the more powerful Blue

Robotics T200s or even newer and more novel solutions such as the Hydromea Diskdrive hubless thrusters. With a more powerful thrust system, the frame could be expanded to carry a higher resolution DSLR-style camera for much greater model precision as well as combat stronger ocean currents. With the above improvements, the ROV system would be fully capable of producing accurate, contextualized models of almost any shallow to mid-depth underwater structure.

9.3 Next Steps

After applying the above hardware changes to the system, further progress can be made to the software side - specifically, improving autonomous functions. With the more capable sensors, the rudimentary lawnmower pattern scripts demonstrated here can be expanded to follow waypoints, orbit objects, and record location data for multiple models in different locations to be stitched together. Further long-term goals exist for the system, most notably the combination with an autonomous surface vehicle that can carry significantly more battery charge than what is available on the ROV. The surface vehicle, responsible for the GNSS position, will carry the ROV to specified locations and deploy the ROV, which will carry out its intended tasks. Once completed, the ROV will dock with the surface vehicle, charging its onboard battery before being taken to a new location. Two autonomous systems, one surface and one sub-surface vehicle, will allow for a drastic improvement in underwater mapping accuracy and efficiency and significantly lower cost and risk when compared to using human divers.

REFERENCES

- Agisoft LLC. (2025). Agisoft Metashape Professional (Version 2.1.0) [Computer software]. <https://www.agisoft.com>
- Alexandris, C., Papageorgas, P., & Piromalis, D. (2024). Positioning systems for unmanned underwater vehicles: A comprehensive review of navigation technologies including Doppler velocity logs. *Applied Sciences*, 14(21), 9671. <https://www.mdpi.com/2076-3417/14/21/9671>
- Ansys, Inc. (2023). Ansys Fluent (Release 2023 R1) [Computer software]. <https://www.ansys.com>
- Ardupilot. Ardupilot documentation. (Accessed September 2025). <https://ardupilot.org/ardupilot/>
- Bayley, D. T., & Mogg, A. O. (2020). A protocol for the large-scale analysis of reefs using structure from motion photogrammetry. *Methods in Ecology and Evolution*, 11(11), 1410–1420. <https://doi.org/10.1111/2041-210x.13476>
- Blue Robotics. BLUEROV2. (2025, December 16). <https://bluerobotics.com/store/rov/bluerov2/>
- Burns, J. H. R., Weyenberg, G., Mandel, T., Ferreira, S. B., Gotshalk, D., Kinoshita, C. K., Marshal, M. J., De Moral, N. A., Murphy, S. J., Pascoe, K. H., Runyan, A., Spengler, A. J., Wells, B. J., Wilde, D. K., Pelayo, R. (2020). A comparison of the diagnostic accuracy of in-situ and digital image-based assessments of coral health and disease. *Frontiers in Marine Science*, 7(May). <https://doi.org/10.3389/fmars.2020.00304>
- Garcia-Nava, S., García-Rangel, M. A., Zamora-Suárez, Á. E., Manzanilla-Magallanes, A., Muñoz, F., Lozano, R., & Serrano-Almeida, A. (2023). Development of a 6 degree of Freedom Unmanned Underwater Vehicle: Design, construction and real-time experiments. *Journal of Marine Science and Engineering*, 11(9), 1744. <https://doi.org/10.3390/jmse11091744>
- Johansen, T. A., & Fossen, T. I. (2013). Control Allocation—A survey. *Automatica*, 49(5), 1087–1103. <https://doi.org/10.1016/j.automatica.2013.01.035>
- Luo, Q., Yan, X., Ju, C., Chen, Y., & Luo, Z. (2020). An ultra-short baseline underwater positioning system with Kalman filtering. *Sensors*, 20(Publication number). <https://pmc.ncbi.nlm.nih.gov/articles/PMC7796008/>
- The MathWorks, Inc. (2025). MATLAB (Version R2025b) [Computer software]. <https://www.mathworks.com>
- MAVLink. MAVLink Developer Guide. (Accessed September 2025). <https://mavlink.io/en/>

MAVLink. (Accessed September 2025). Using Pymavlink libraries (mavgen).
https://mavlink.io/en/mavgen_python/

McCarthy, J., & Benjamin, J. (2014). Multi-image photogrammetry for underwater archaeological site recording: An accessible, diver-based approach. *Journal of Maritime Archaeology*, 9(1), 95–114. <https://doi.org/10.1007/s11457-014-9127-7>

Moberg, F., & Folke, C. (1999). Ecological Goods and services of Coral Reef Ecosystems. *Ecological Economics*, 29(2), 215–233. [https://doi.org/10.1016/s0921-8009\(99\)00009-9](https://doi.org/10.1016/s0921-8009(99)00009-9)

Nocerino, E., Stathopoulou, E. K., Rigon, S., & Remondino, F. (2020). Surface Reconstruction Assessment in photogrammetric applications. *Sensors*, 20(20), 5863.
<https://doi.org/10.3390/s20205863>

Oceanbotics. SRV-8: Professional remotely operated vehicle. (2025, August 19).
<https://oceanbotics.com/srv-8/>

Pizarro, O., Friedman, A., Bryson, M., Williams, S. B., & Madin, J. (2017). A simple, fast, and repeatable survey method for underwater visual 3D benthic mapping and monitoring. *Ecology and Evolution*, 7(6), 1770–1782. <https://doi.org/10.1002/ece3.2701>

Teague, J., & Scott, T. (2017). Underwater Photogrammetry and 3D Reconstruction of Submerged Objects in Shallow Environments by ROV and Underwater GPS. *Journal of Marine Science Research and Technology*.
https://www.researchgate.net/publication/329611845_Underwater_photogrammetry_and_3D_reconstruction_of_submerged_objects_in_shallow_environments_by_ROV_and_underwater_GPS