MuddSub: Exploring Vehicular Autonomy

Ket Hollingsworth Jordan Carlin Ravago Jones Ryan DeBarger MuddSub Team Members

***Abstract*—This paper discusses the continued development of MuddSub's autonomous underwater vehicles, Alfie and Crush, in preparation for the 2024 International RoboSub Competition, which will mark MuddSub's third in person competition. Alfie was developed with a unique design philosophy that emphasized Simplicity, Stability, and Scalability. Though slightly more complex, Crush was designed with the same philosophy in mind. With the maturation of the MuddSub organization, we aim to underscore the importance of this philosophy and highlight the significance of mentorship within the team. Given that five core members are graduating this year, a crucial objective is to instill strong foundational robotics principles in newer members. These foundations include a deep understanding of sensor integration, precise control algorithms, electrical system design, and robust mechanical design. Sensor integration involves the seamless combination of various sensors, such as depth sensors and Doppler Velocity Logs (DVL), to provide accurate and reliable data for navigation and state estimation. The team also focused on development of the electrical and mechanical systems, building on previous work of the team.**

To help strengthen the foundations of the organization, the previous numerous sub-teams have been reorganized into three generalized teams: Mechanical, Electrical, and Software. A key feature of these teams is the integration of a shared workspace and an emphasis on integration and testing. Through this paper, we aim to illustrate how the principles of our design philosophy serve as guiding tenets for our competition strategy and vehicle design.

By adhering to a clear set of goals, we seek to demonstrate effective competition strategies and innovative systems engineering while avoiding over-engineering. Furthermore, we hope to inspire the RoboSub community by showcasing how a well-defined design philosophy can lead to efficient and impactful contributions to the field of autonomous underwater vehicles.

Fig. 1. Alfie.

I. Design Philosophy

The conception, design, and manufacture of Alfie were initially guided by the principle of simplicity inherent in our Design Philosophy. While this focus on simplicity offered several advantages, it also introduced certain limitations, such as significant blind spots both above and below the robot that hindered the execution of specific tasks. In contrast, the design of our latest robot, Crush, sought to overcome these limitations by embracing a slightly more complex approach. Although the team does not plan to use Crush in this year's competition, it demonstrates our commitment to our Design Philosophy. Crush was engineered to be small, agile, and powerful, reflecting the evolution of our Design Philosophy. Despite these advancements, Crush still abides by our Design Philosophy, and in particular is still constrained by its reliance on vision-based navigation in hopes of preserving simplicity.

The three interconnected principles of our Design Philosophy hold true more than ever now that MuddSub is a maturing organization. Let us define these three principles clearly.

A. Simplicity

We define simplicity as avoiding superfluous complexity. Simplicity often embodies the restrictions inherent to being a new team. These restrictions include development time, human resources, and monetary resources. Even now as a maturing team, these restrictions still apply to new incoming members. In terms of development time, all members of MuddSub are full-time students at a rigorous institution, and over the summer we are all balancing this project with full-time research and internship positions. Maintaining a given level of simplicity is key for the organization to continue to provide an incredibly valuable educational experience.

B. Stability

We define stability as designing for minimal change to a robot's core systems. Stability further serves as our measure of reliability. To exercise our desire for stability, we are prioritizing concepts in a greedy strategy: rather than focusing our efforts on accomplishing many of the complex tasks, we only work on tasks we believe are achievable for the next competition. By doing so, our team can maximize our potential for success in this year, while also laying the foundation for success in future years. In order to achieve stability, we need to prioritize design decisions which would work consistently this year and would require minimal modifications to the robot in the future. Stability is connected to simplicity as robots with high simplicity tend to be stable.

C. Scalability

We define scalability as the possibility of future expansion.

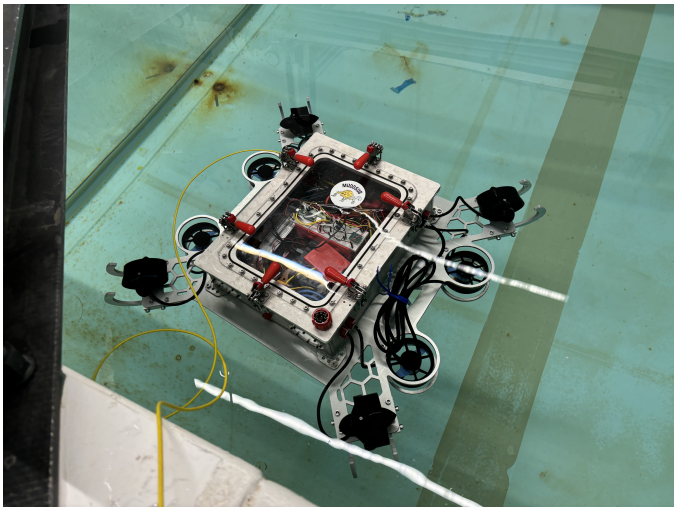

Fig. 2. Testing Alfie controls in a 8 foot square pool.

Scalability is inherently linked to stability. In order to ensure the future growth of the organization, all of our robots must be modifiable without a complete redesign. There is an inherent relationship between scalability and stability because reducing the changes made to existing systems will allocate/divert more time and resources to new systems.

II. Competition Strategy

Our competition begins with the Pre-Qualification attempt, where we will construct the Horizontal Gate and Vertical Marker in our tank room to test Alfie. The process will start by orienting Alfie orthogonally to the gate. Utilizing our state machine, we will enter a gate searching state, moving Alfie slowly forward until our YOLOV6 model detects the individual legs of the gate.

Upon detection, we will transition into the gate approach state, laterally adjusting our position to center our view between the two legs of the gate. Alfie will then continue to advance until a marker is detected. We will ensure our path stays to the right of the marker. Once the marker is out of view, Alfie will move forward and to the left until our IMU indicates an orientation nearly opposite to the initial direction.

Following this, Alfie will perform a sinusoidal swim pattern until the gate is detected again. At this point, we will return to the gate orienting state, making lateral adjustments to re-center Alfie between the legs of the gate.

After passing the gate, MuddSub plans to test its new marker system. To find the marker system, Alfie will leverage its visual mapping, slam, and sensor integration to form a small ongoing map of the pool. Using data pooling with prior trained marker images and YOLO, the visual system will be tuned to this years objects. At deployment, if a marker object is identified with a high confidence, Alfie will use a combination of estimated distance from object, object tracking, and estimated speed of the robot to best predict when Alfie is aligned to the marker. Due to no downward facing camera, this will be a more difficult challenge.

If the marker task is completed successfully, MuddSub's prior torpedo system will be tested. Prior data collection has been used to identify the torpedo board, estimate distance from board, and create a control loop to align Alfie's right camera to the torpedo hole. A spring loaded torpedo will then fire as released by a servo motor.

This strategy is designed to leverage Alfie's sensor capabilities and precise control algorithms, ensuring reliable navigation and successful completion of the Pre-Qualification attempt.

III. Design and Testing

A. Electrical: Crush and Replacing Alfie Components

This year, the electrical subteam continued work on the electronics system for the new robot, Crush. We successfully designed and prototyped implementations for a battery voltage monitor and a software kill switch (for the robot's thrusters). These will allow the software team to have more control over the electical system, so that they can ensure the safe operation the robot. We incorperated these designs into the exising electrical schematics and completed layouts for two new PCBs for Crush: a power regulation board and a signal board. These PCBs will simplify the wiring for Crush and allow for plenty of expansion in the future. Both of these will undergo final design reviews and fabrication/assembly in the upcoming year.

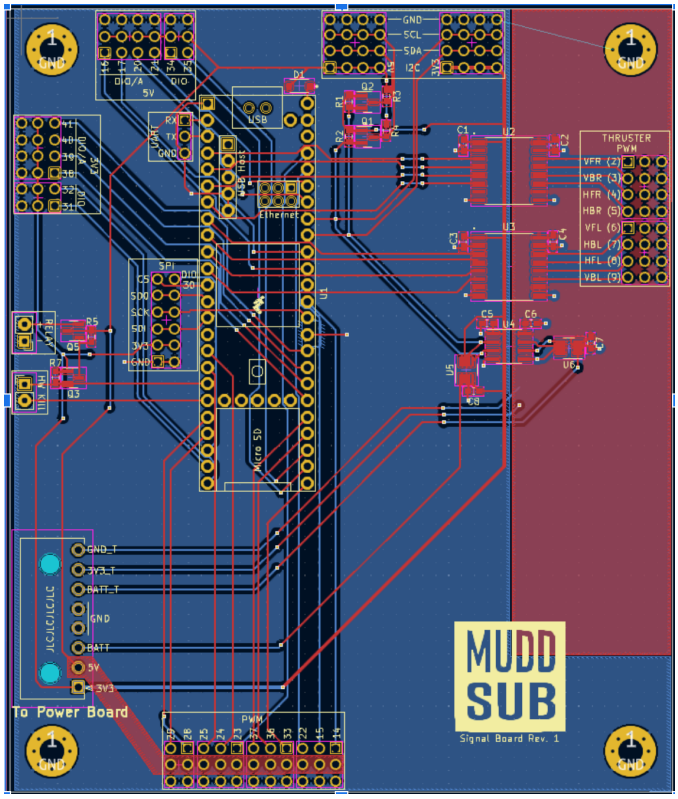

Fig. 3. Signal Board Layout.

Crush is powered by two batteries: one for the thrusters, and one for the control electronics. This is done to allow

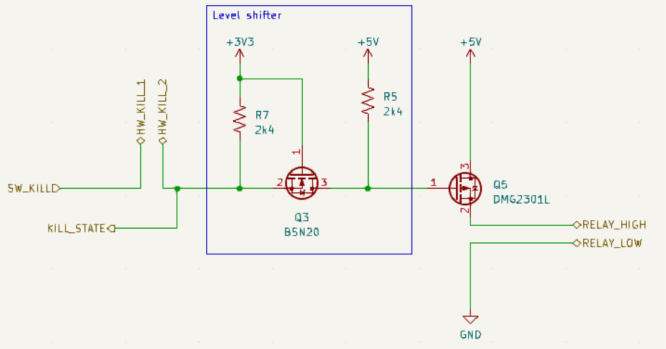

Fig. 4. Software Kill Switch Schematic.

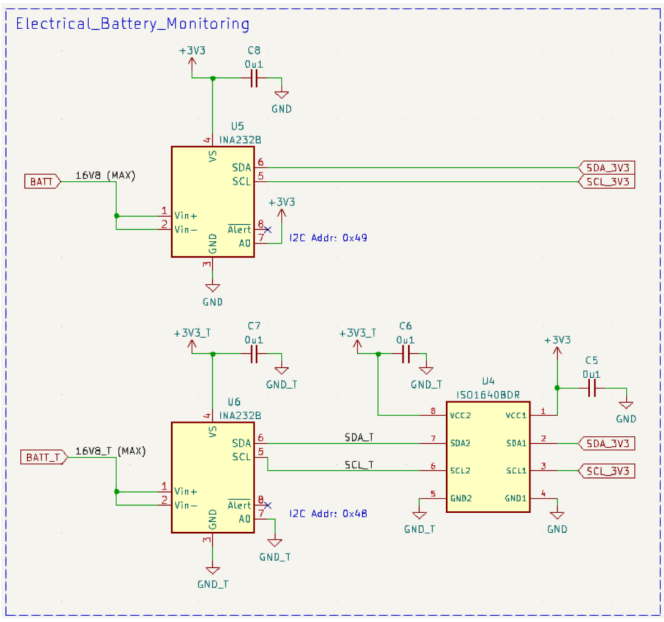

Fig. 5. Battery Monitoring System Schematic.

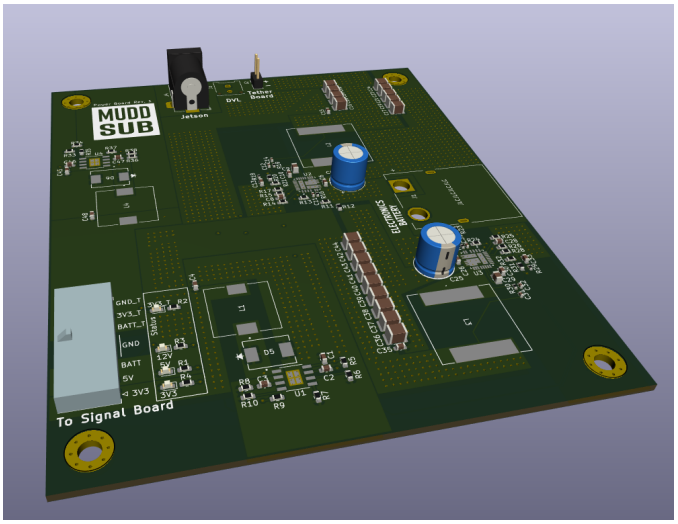

Fig. 6. Power Board Layout.

the thrusters to be powered off while the rest of the robot is running. The power board takes the raw battery voltage and provides a lower, more stable, voltage source for the rest of the robot. This includes a 3.3V source for controlling the thrusters, and 3.3V, 5V, and 12V sources for powering the control electronics. It then has additional connectors which allow it to distribute the appropriate voltages to the other devices on the robot. The signal board is designed to fit in the same footprint as the power board. This will eventually allow the two to be mounted on top of each other, saving space in the robot. It provides connection points for all of the pins on the Teensy microcontroller (used to forward signals from the Jetson, which provides high level robot control, to many of the actual devices on the robot). It also connects these headers to the power board, so that devices can receive both power and control information over the same bus. In addition to this, it will have circuitry to monitor the charge of each battery and disconnect the thruster battery if necessary. (Together, these will allow us to automatically disconnect the battery when it is running low, preserving battery heath.) Finally, it includes multiple optoisolators to allow signals to be passed to the thrusters while maintaining separation between the thruster and signal power supplies.

B. Mechanical: Development of the Marker Subsystem

The mechanical subteam picked up right where it left off at the end of last year: working on the marker subsystem. Last year, the team had settled on a dodecahedron shaped marker based on its movement through the water and its ability to stay where it lands (increasing the odds of points from the rim of the target zone if the AUV was slightly out of alignment). This year, the goal was to take that prototype marker, develop a system for holding and dropping it, and begin work on a final, integrated version. The team performed extensive prototyping to evaluate different ways of holding the marker. Knowing that there were two markers, one big decision was whether to drop them at the same time or separately and whether to stack them or keep them side by side. After observing the way the two markers interfered with each other in water testing, the team decided to keep them separated side by side and drop them one or two seconds apart from each other. There were also severe size constraints, so care had to be taken to ensure the overall device was as small as possible.

The final design places the two dodecahedron shaped markers side by side with a servo between them. The servo has an attachment designed to hold both markers in place until it rotates, at which point one marker is released immediately and the other falls a second or two later after the piece has fully rotated. This prevents them from hitting each other in the water while avoiding the space that two separate servos would need. This final prototype was tested in Harvey Mudd's large tank, and performed very well with both markers dropping into a five gallon bucket positioned underneath nearly every time. Due to a lack of time and people this year, a final version was not machined, but there are plans to produce a higher quality 3D-printed version early next year and attach it to the AUV.

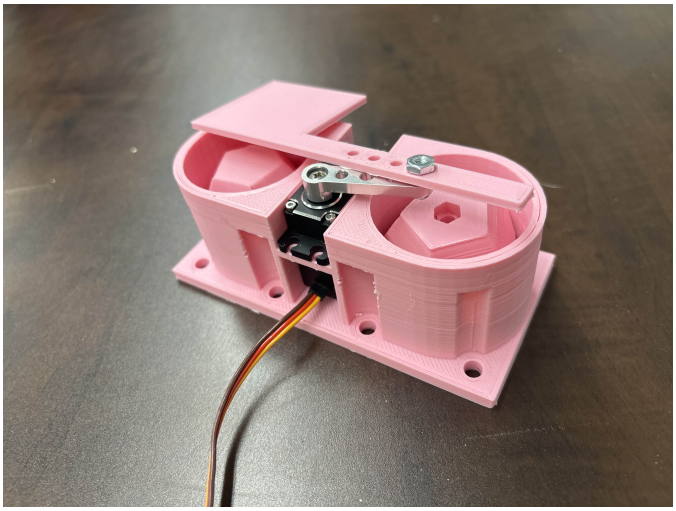

Fig. 7. Marker Mechanism.

C. Software Overview:

Over the years there has been much progress in our Software techniques. Beginning with simple PID controls we are able to control the movement of our robot in any desired fashion. We then built up a set of reference points of the world around us using YOLOv3 and low level CV techniques. Since then we have moved onto the YOLOv6 model due to its improved speed and accuracy. These reference points would then be integrated with our set of sensors (IMU, DVL, Stereo Depth Map) and fed into our FastSLAM 2.0 algorithm in hopes of generating mappings of our surroundings. From this point on we are able to pinpoint our position in relation to other objects, and we can even choose to navigate towards said objects. Navigation takes care of this and uses our SLAM mapping to generate a valid path. With all of this in place we can then decide where we want to go and how we want to behave. This then leaves us at the State Machine the final piece which is currently still in development.

In previous years a majority of the focus was spent on developing these algorithms to aid in the navigation of various environments, but this year we have spent our time on other areas. Despite this, we still retain the progress and plan to incorporate all of these techniques into our final state machine in preparation for the final competition. In this report, we will discuss the subsystems the team primarily focused on for the year, navigation and controls, sensor integration, and simulation development.

D. Navigation and Controls: Alfie

The navigation sub-system is responsible for planning the robot's trajectory. This sub-system works closely with the state machine and SLAM sub-systems. The state machine tells the robot where it should go in order to accomplish its current task, and the SLAM sub-system tells the robot where it currently is in its environment. Navigation combines these pieces of information to produce a series of poses for the robot to follow to get from a starting location to some goal location. The Navigation subteam implemented the A* algorithm for motion planning. It takes as its input a grid map showing what parts of the space are occupied and outputs a path along that grid that the robot can safely take. This past year, we experimented with different heuristic functions. The subteam also wrote code to generate trajectories that have specific shapes. For example, it is useful to have the robot move in a sine wave in order to take measurements of obstacles at different angles to gather increased information for SLAM. Another example is planning a circular path around some obstacle in order to allow the cameras to view it from many different angles. For controls, Alfie uses PID on its thrusters to achieve desired thrust.

E. Sensor Integration

Throughout the year we focused on integrating more of the robot's sensors that we did not utilize at last year's competition. We rewrote old code for many of the sensors that had been used in past years. We now have ROS nodes that publish data from every sensor, including our IMU, gyroscope, depth sensor, DVL, and ZED stereo camera. We also wrote some better controller code that allows us to manually control the robot when tethered that uses an actual joystick instead of a keyboard. This allows us to more easily test new functionality, gather sensor data, and diagnose electrical or mechanical issues.

F. Simulation

We have also explored the possibility of incorporating Gazebo into our suite of tools. Being able to simulate navigation and control algorithms in a simulation greatly speeds up the testing of available navigation algorithms. We plan on continuing the incorporation of Gazebo in the future for easier integration and testing of proposed navigation and control algorithms.

G. Overall System

To ensure that all subsystems work together, many rounds of testing are required. Our initial testing focuses on ensuring that all robot hardware is operational underwater. This includes testing individual thrusters, as well as coordinated multi-thruster movement, and collecting and reviewing sensor data to ensure functionality. The computer vision subsystem is also tested in this initial round of testing. This system is tested by pointing the robot's cameras at a scaled down version of the gate, which is manually moved around by the team to show different angles and distances from the gate. The next round of testing focuses on state estimation. Getting the SLAM subsystem working early is essential as nearly every other subsystem relies on it in some way. Without an estimate of the robot's location and current state, it is impossible to plan what tasks should be performed and impossible to navigate reliably. State estimation testing is conducted by putting the robot in the water and having it localize itself and map its surroundings, then manually moving it around and comparing the estimated map with ground truth data collected through video recording

the robot from outside of the water. After state-estimation comes controls, i.e. the subsystem that enacts the commands given to the robot. Some control system is necessary in order for the navigation subsystem to work. Control testing is done by having the robot track a predefined trajectory, and comparing with ground truth data collected by video recording the robot. Finally, the last subsystem to be integrated and tested is the state machine. This subsystem relies on all other subsystems to varying degrees. However, it is possible to forego some of the more complex navigation algorithms for path planning for the purposes of testing the state machine. For example, the robot could just be commanded to follow a straight line until it reaches the destination for the next task. To test the state machine, the team has the robot follow through the whole state machine and records data on the robot about state information to compare with the anticipated progression.

This testing procedure is done in order to ensure full system integration, so each step is performed at different stages in the design cycle to measure performance of different components when integrated with the full system. For example, after major control development, the team might test all of the subsystems controls is reliant on to ensure those work, then see how the controls subsystem works when fully integrated.

ACKNOWLEDGMENTS

The authors would like to acknowledge the Associated Students of Harvey Mudd College (ASHMC) for providing funding and support, allowing for purchasing of materials and resources for MuddSub. The student budget has played a crucial role in securing a workspace for students to work on MuddSub and explore their passion for robotics and autonomy. Additionally, the authors would like to thank the faculty, staff, and team members at Harvey Mudd College for mentorship and collaborative efforts. In particular, we wish to thank our advisor, Professor Zachary Dodds for budget advising, Xavier Walter, for his assistance in debugging electrical equipment, and Drew Price, for training members in the machine shop and manufacturing efforts. The contributions of all these individuals and organizations have been instrumental in the success and impact of this work. /

REFERENCES

[1] Michael Montemerlo, Sebastian Thrun, Daphne Roller, and Ben Wegbreit. 2003. FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In Proceedings of the 18th international joint conference on Artificial intelligence (IJCAI'03). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1151–1156.

[2] Sebastian Thrun, Wolfram Burgard, Dieter Fox. Probabilistic robotics. Intelligent robotics and autonomous agents, MIT Press 2005, ISBN 978-0-262-20162-9

[3] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement." arXiv, 2018. doi: 10.48550/ARXIV.1804.02767.

[4] Chuyi Li, Lulu Li "YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications" arXic, 2022. arXiv:2209.02976

APPENDIX A

Component	Vendor	Model/Type	Specs	Custom/Purchased	Cost
Buoyancy Control		Dive Weights		Custom	
Frame	In-house machined			Custom	
Waterproof Housing	In-house machined			Custom	
Waterproof Connectors				Custom	
Thrusters	Blue Robotics	T100		Purchased	8x $119
Motor Control	Blue Robotics	Basic ESC		Purchased	4x $27
High Level Control					
Actuators					
Propellers					
Battery	HobbyKing	Turnigy	4S, 10 AH	Purchased	2x $99
Converter				Custom	
Regulator				Custom	
CPU	NVIDIA	Jetson AGX Xavier	30W, 32 TOPS	Purchased	$650
Internal Comm Network					
External Comm Interface					
Compass					
Inertial Measurement Unit (IMU)	VectorNav	VN-100T		Purchased	Donated
Doppler Velocity Log (DVL)	Waterlinked	DVL A50	0.1mm/s Resolution	Purchased	$5200
Manipulator					
Algorithms		Conv Net, Unitary ESPRIT, FastSlam 2.0			
Vision					
Acoustics	Teledyne	Hydrophones		Purchased	4x $1443
Localization & Mapping					
Autonomy					
Open Source Software	Canonical Ltd.	Ubuntu 20.04			Free
Inter-Vehicle Communication					
Programming Language(s)		Python3, C++			Free