Underwater Multi-Robot Simulation and Motion Planning in *Angler*

Akshaya Agrawal[1], Evan Palmer[1*], Zachary Kingston[2], and Geoffrey A. Hollinger[1]

***Abstract*—Deploying multi-robot systems in underwater environments is expensive and lengthy; testing algorithms and software in simulation improves development by decoupling software and hardware. However, this requires a simulation framework that closely resembles the real-world. *Angler* is an open-source framework that simulates low-level communication protocols for an onboard autopilot, such as *ArduSub*, providing a framework that is close to reality, but unfortunately lacking support for simulating multiple robots. We present an extension to *Angler* that supports multi-robot simulation and motion planning. Our extension has a modular architecture that creates non-conflicting communication channels between Gazebo, ArduSub Software-in-the-Loop (SITL), and MAVROS to operate multiple robots simultaneously in the same environment. Our multi-robot motion planning module interfaces with cascaded controllers via a `JointTrajectory` controller in ROS 2. We also provide an integration with the Open Motion Planning Library (OMPL), a collision avoidance module, and tools for procedural environment generation. Our work enables the development and benchmarking of underwater multi-robot motion planning in dynamic environments.**

***Index Terms*—Underwater Robotics, Simulation, Multi-robot Systems, Motion Planning**

I. Introduction

Underwater multi-robot teams are capable of addressing complex, collaborative tasks such as connecting pipes in offshore sub-sea pipeline networks, and can explore areas faster than one robot alone [1], [2], [3]. However, deploying multi-robot systems underwater is risky, difficult, requires significant hardware and labor investment, and is time-intensive. Many components of the system (e.g., integration of different modules such as sensors, controllers, and planners) can be evaluated in simulation to reduce these costs, and high-level integrations can be tested before deployment on real hardware.

In particular, motion planning is a core capability of autonomous underwater robots as it enables collision-free navigation. Fundamental to motion planning is collision detection, a computationally expensive building block which increases in complexity with the team size of robots. In the underwater case, we are also interested in dynamic environments where objects are affected by ocean currents and buoyancy. As objects in the environment are moving, online motion planning is required to maintain safety; this requires the pose of all the robots and obstacles. A fast, dynamic collision detection module is required for online multi-robot motion planning.

[1]Collaborative Robotics and Intelligent Systems (CoRIS) Institute, Oregon State University, {agrawaak, palmeeva, geoff.hollinger}@oregonstate.edu

[2]Department of Computer Science, Purdue University zkingston@purdue.edu

This work was supported in part by ONR awards N0014-21-1-2052 and N00014-22-1-2114.

The work of Evan Palmer was supported by the National Defense Science and Engineering Graduate (NDSEG) Fellowship.

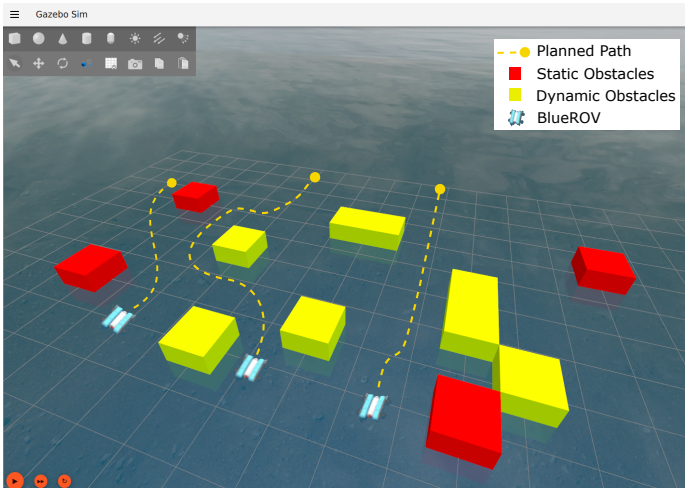

Figure 1: A scene in *Angler* showcasing our multi-robot extension, which simulates a team of robots influenced by ocean currents. These robots are tasked with autonomously navigating around obstacles in a collaborative exploration mission. Our multi-robot extension to *Angler* establishes conflict-free interfaces between Gazebo, ArduSub SITL, and MAVROS, enabling coordinated or individual control of each robot. Additionally, we provide tools for integrating controllers with the motion planning through OMPL and for generating environments that include both static and dynamic obstacles.

Several simulation frameworks address subsets of these requirements for underwater robotics [4], [5], [6], [7]. *Angler*[1,2] is an open source software framework that, unlike other simulators, interfaces with ArduSub software-in-the-loop (SITL) [8], enabling testing of software in simulation while maintaining a close-to-hardware deployment setup, enabling eventual deployment on real hardware. ArduSub SITL provides close to real-world hardware-software integration pipeline by simulating a flight control system of a real underwater robot. *Angler* is based on the latest Gazebo for simulation and is built using ROS 2. Although this framework is specifically designed for BlueROV2 Heavy[3] and BlueROV2 Heavy mounted with Reach Alpha 5 Manipulator[4], it can be adapted to other underwater vehicles. Despite these strengths, *Angler* does not provide support for motion planning out of the box and also lacks support for multi-robot systems due to difficulties in setting up multiple, modular ROS 2, ArduSub SITL, and MAVROS communication channels for every robot in a multi-robot system, while also supporting coordinated control.

This work extends *Angler* to support multi-robot simulation and motion planning, shown in Fig. 1. Our extension provides a platform for both high-level multi-robot coordinated planning approaches and online dynamic replanning. Our code is

[1]https://github.com/Robotic-Decision-Making-Lab/blue
[2]https://github.com/Robotic-Decision-Making-Lab/angler
[3]https://bluerobotics.com/
[4]https://reachrobotics.com/

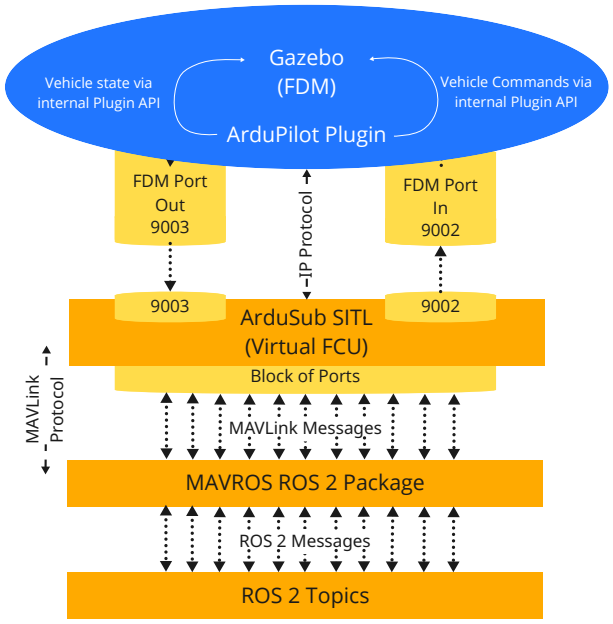

Figure 2: The existing software architecture for simulation in *Angler*.

available in the *Angler*[1] repository. The key contributions of this paper include:

1) A multi-robot simulator that supports virtual autopilot capabilities through ArduSub SITL for underwater vehicles.
2) Integrated tools for motion planning including real-time environment feedback for online planning, collision detection, and an environment generator to support evaluation in diverse environments.
3) Compatibility with any set of controllers that can be integrated below the `JointTrajectory` controller from the `ros2_control` framework.

II. Related Work and Background

There is widespread use of simulation in underwater robotics [9], and there has been focus in developing more accurate simulation to simulate aquatic conditions, e.g., [10]. However, there are few underwater simulators that support multi-robot systems, summarized in Tbl. I. Stonefish [4] is a Bullet-based [11] simulator capable of simulating multi-robot systems. It utilizes the collision detection and multi-body dynamics modeling functionalities of Bullet at its core. However, it is purely a simulator and does not offer any provision for hardware deployment. The UVMS Simulator [5] on the other hand, is designed as a hardware deployment framework. It supports hardware-in-the-loop testing, allowing it to directly interface with a BlueROV Heavy robot and perform tests on hardware. However, both of these simulators lack ArduSub simulation, which is critical for a hardware-software integration pipeline that is closer to real-world systems without requiring actual hardware.

DAVE [12] and the UUV Simulator [13] are other widely used underwater simulators built on top of Gazebo. While DAVE supports multi-robot simulation, it is built with ROS 1. With the release of ROS 2 [14], the older ROS framework will be deprecated by the end of 2025. Moreover, ROS 2 offers

Frameworks	Simulator	ROS 2	ArduSub SITL	Cascade Control
MR Angler (Ours)	Gazebo	✓	✓	✓
UVMS Simulator [5]	Gazebo	✓	×	×
Stonefish [4]	Bullet	✓	×	×
DAVE [12]	Gazebo	✓	×	×
HoloOcean [18]	Unreal	✓	×	×
LRAUV [19]	Gazebo	✓	×	×
UUV Simulator [13]	Gazebo*	×	×	×
ds_sim [20]	Gazebo*	×	×	×
UWSim-NET [21]	Gazebo*	×	✓	×

Table I: Comparison of multi-robot simulation capabilities among existing frameworks that support multi-robot systems. Gazebo* indicates Gazebo Classic in ROS 1.

better support for real-world deployments along with the core functionalities of ROS. To exploit the newer features of ROS 2 and to prevent them from becoming obsolete, it is advisable to build systems using ROS 2. Although efforts are being made to develop DAVE using ROS 2, it has not yet been officially released [12].

Our decision to extend *Angler* [15] rather than other existing simulators stems from several considerations. *Angler* is similar to the Marine Vehicle Packages project [16], which provides a framework for control and high-level autonomy. However MVP is strictly targeted towards single-system deployments. *Angler* has a modular architecture that enabled us to modify and extend individual modules to support multi-robot systems. Our main objective is to provide a platform that allows testing integrated multi-robot motion planning algorithms. *Angler* abstracts the robot-specific controller stack using cascade control architecture, allowing the development of capabilities to support multi-robot motion planning. Additionally, its ability to interface with ArduSub SITL [17] is of utmost importance to us, as it enables us to test multi-robot software architecture in simulation before actual deployment.

A. Angler Software Architecture

Angler is an open-source software framework designed for developing autonomous capabilities for underwater robots. For simulation purposes, it interfaces with Gazebo as the flight dynamic model (FDM) that simulates the robot and onboard sensors. An ArduPilot [17] plugin establishes a two-way bridge between the simulator and the ArduSub SITL to exchange data and actuator commands.

ArduSub is an autopilot system for underwater vehicles from ArduPilot project. Ardusub SITL functions as a virtual flight control unit (FCU) that simulates the behavior of underwater systems FCUs, such as Navigator[5]. The FCU communicates with the Ground Control System (GCS) running on ground computers over UDP/TCP connections following the MAVLink Protocol. While ROS 2 serves as the de facto software framework for developing robotic applications, it lacks native support for MAVLink messages. To address this limitation, MAVROS—a ROS 2 package—acts as a proxy GCS that decodes MAVLink messages and publishes them as ROS

[5]https://bluerobotics.com/store/comm-control-power/control/navigator/

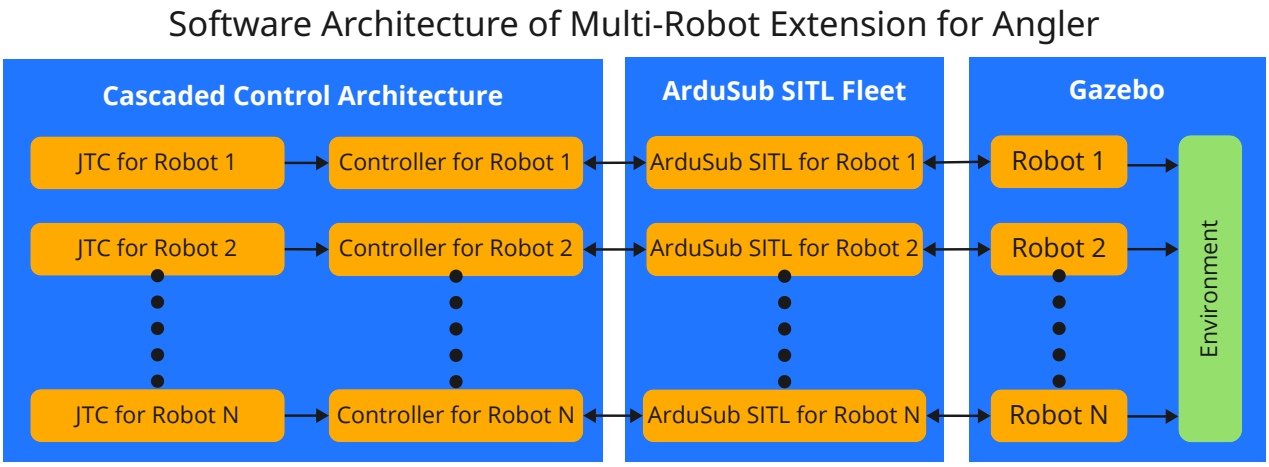

Figure 3: Software architecture illustrating our extension of *Angler* (Fig. 2) to enable multi-robot simulation.

messages on ROS Nodes. To accurately replicate real hardware-software integration, ArduSub SITL interfaces with Gazebo using IP protocol and with ROS 2 through MAVROS. The entire *Angler* architecture is given in a diagram in Fig. 2.

However, initially, *Angler* was developed for a single robot use case, leading to hard-coded configurations for interfacing between subsystems, which constrains multi-robot simulation capabilities.

III. Underwater Multi-Robot Simulation

Our simulation framework is designed for research and development of solutions for multi-robot underwater applications including exploration and transportation tasks. We offer integrated support for autopilot simulation which enables testing the software pipeline along with planning and control algorithms before deployment on hardware. We provide an illustration of our software framework in Fig. 3. Our framework is informed by the following design goals to support multi-robot simulation:

1) *Leverage Existing Features of Angler:* We build our multi-robot simulation capabilities on top of Angler's useful underwater simulation features, including current simulation and vehicle dynamics modeling.
2) *ROS 2 and Gazebo Integration:* We support the latest ROS 2 and Gazebo versions, to enable building on and integrating newer features such as cascade control.
3) *Modularity for Multiple Robots:* We extend Angler with multi-robot simulation such that it retains its modularity. For example, we spawn a separate controller manager for each robot to avoid conflicts. We make the interface between different modules such as Gazebo and ArduSub SITL configurable.

A. Simulation in Gazebo

We use *Angler*'s Gazebo-based simulation framework for modeling underwater environments. It simulates ocean currents and sensor noise, facilitating *in silico* testing of planning and control algorithms under different conditions. Our extension to Angler simulates multiple underwater vehicles concurrently, where robot-specific properties, such as added mass, can be set individually using configuration files, allowing a heterogeneous fleet of robots.

To mimic a setup close to real-world deployment, Gazebo interfaces with the Ardupilot plugin to exchange information with other components of the framework. To achieve modularity for a multi-robot system, we have configurable Flight Dynamics Model (FDM) ports. This plugin serves as an interface that connects the Gazebo sensor simulation data and actuator commands to the Ardusub Software-In-The-Loop (SITL), which virtualizes the Flight Control Unit (FCU). The bidirectional communication enables execution commands from SITL to be accurately reflected in the Gazebo simulation environment, providing a closed-loop testing capability for complex underwater operations.

B. ArduSub SITL Integration

As discussed in Sec. II-A, ArduSub Software-In-The-Loop (SITL) is a firmware that virtualizes the autopilot running on an underwater robot. Integrating ArduSub SITL into our testing framework enables us to test on actual vehicle firmware and validate our software pipeline. The ArduSub SITL module implements sensor fusion using Extended Kalman Filter (EKF), thus providing realistic localization. It is a sandwich module that interacts with the simulator through Flight Dynamics Model (FDM) ports on one side and MAVROS on the other. To establish a multi-robot interface channel with MAVROS, each robot requires access to a unique set of ports; additionally, ArduSub SITL generates MAVLink messages following the MAVLink protocol to mimic a real-world communication network. We provide easy configuration of distinct MAVLink ports for multiple robots by allocating non-conflicting ports to each robot, thus enabling the simultaneous simulation of multiple ArduSub SITLs for multi-robot operations.

C. Controllers

Angler implements a cascaded control architecture. When controlling multiple robots, each robot requires its own set of

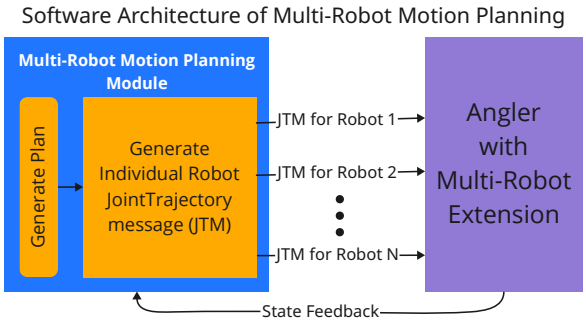

Figure 4: Software architecture for multi-robot motion planning.

cascaded controllers. One of the issues that prevented multi-robot simulation in other frameworks is name collisions, due to hard-coded assumptions in controllers. We have a modular framework that automatically sets the prefixes for `tf2` frames and correctly namespaces the nodes and topics. However, controller stack is managed through controller managers. We spawn robot-specific controller managers to track the controller fleet along with the corresponding hardware interfaces.

Many robotic applications require a combination of autonomous and teleoperated control. For instance, a motion planning algorithm might navigate a robot to the vicinity of a sample collection point, but final precision manipulation may require human oversight. We integrate with the `teleop_twist_keyboard` package to provide simultaneous teleoperation capabilities for multiple robots.

IV. Multi-Robot Motion Planning

We design a multi-robot motion planning framework that interfaces with the cascaded control architecture of *Angler* via a `JointTrajectory` controller, shown in Fig. 4. This enables execution of motion plans within the Gazebo simulation environment. Our planning framework has the ability to receive state feedback from *Angler* which can then be processed and used for updating the motion plans.

A. JointTrajectory Controller Integration

To connect the motion planner and cascaded controller stack from *Angler*, we use `ros2_control`'s `JointTrajectory` controller as the preceding controller. The `JointTrajectory` controller accepts `JointTrajectory` messages from the motion planner, ensuring planner-controller independence through a standard interface. This modularity enables the motion planner to remain agnostic to the underlying control implementation while supporting arbitrary controller stacks that can be integrated with `JointTrajectory` controller.

B. Collision Avoidance Module

Collision checking is empirically one of the most computationally expensive operations in motion planning [22], with complexity increasing proportionally to the number of bodies in the environment. Although Gazebo offers a plugin for collision detection, we need a fast method to check for collisions of any potential state of the system to generate plans. FCL [23], a popular C++ library that provides fast collision checking, is used within numerous frameworks, e.g., MoveIt, Trimesh, and Pinocchio [24], [25], [26]. In particular, we are interested in exploring cooperative construction activities underwater, which can utilize manifolds for defining constraints [27], which is well supported by Pinocchio. Hence, we chose Pinocchio to develop a multi-robot collision checking module. This module allows us to check for inter-robot collisions, along with external collisions with the environment.

C. OMPL Integration

The Open Motion Planning Library (OMPL) [28] offers a rich collection of state-of-the-art sampling-based motion planners. We integrate OMPL with our multi-robot collision checking and modeling to generate trajectories for the entire multi-robot system Fig. 4. Here, we model the multi-robot system as a composition of the individual configuration spaces, planning for the motions of the robots simultaneously. We plan to develop custom OMPL-based motion planners for more complex use cases like collaborative underwater transportation [27].

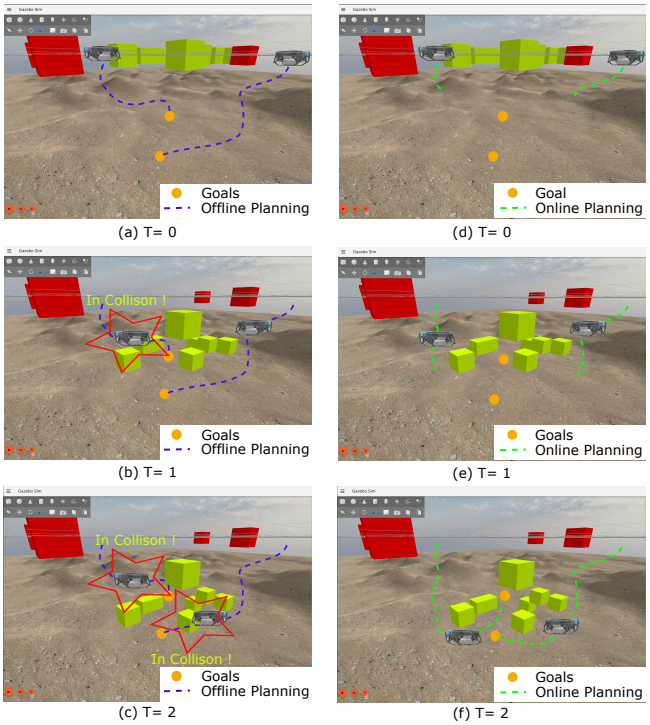

Figure 5: A scene in *Angler* showcasing online and offline motion planning in presence of static and dynamic obstacles for two robots. $T = 0, 1$, and 2 represents three distinct timesteps.

D. Environment Generation Tool

To validate the robustness of a motion planning algorithm or to identify potential failure cases, we must perform rigorous testing in diverse environments. For learning-based planners that require datasets to train their models, manually generating environment configurations is impractical and time-consuming. We provide a procedural generation approach based on a

cellular automata [29] to populate obstacles in the environment, creating a combination of fixed and dynamic obstacles, an example of which is shown in Fig. 5. Dynamic obstacles represent objects set in motion by ocean currents and the effects of buoyancy, introducing temporal complexity to the navigation problem. Our tool can generate environments with varying levels of complexity. Environment complexity is defined by obstacle density and the number of dynamic obstacles.

E. Benchmarking Planning for Underwater Applications

Our simulation framework and motion planning tools enables multi-robot motion planning and benchmarking of novel motion planners against existing ones, particularly those supported by OMPL. We enable the comparison of different algorithms under identical environments and simulation setups using metrics such as computation time, execution time, and success rate. Our procedural environment generation tool enables us to benchmark the performance of different algorithms in various settings, including environments with static and moving obstacles. We can identify failure points of unsuccessful plans by utilizing Gazebo's collision detection capabilities. The simulator enables us to find exact parameters that optimizes firmware performance. Our framework also supports developing integrated motion planning algorithms that incorporate real-time feedback about robot and obstacle states, thus enabling us to develop and validate multi-robot motion planning algorithms for underwater.

V. Discussion and Conclusion

We have presented an extension to *Angler* to support simulation of multiple robots, a framework that enables the development and validation of solutions for autonomous multi-robot operations, including sensing, control, and motion planning. The core capabilities that our extension provides over the existing *Angler* framework include establishing non-conflicting communication channels between Gazebo, ArduSub SITL, and MAVROS for each individual robot in a multi-robot setup, where multiple robots can operate in the same environment simultaneously.

We also developed a multi-robot motion planning tool by providing a high-levl `JointTrajectory` controller that interfaces with robot-specific controller stacks. Our ROS 2 package supports any controller stack that can be interfaced with the `JointTrajectory` controller. Our OMPL integration and environment generation tool enable rapid development and testing against existing algorithms.

One challenge with the underwater domain is testing for scalability with an increasing number of robots, as real-world testing is expensive. Our framework can be used to estimate the number of robots required for a task. We are also interested to evaluate whether our framework can be used for generating datasets or simulating such applications for learning-based methods.

Our ongoing goal is to perform underwater cooperative construction and transportation tasks. Future work will focus on multi-robot mobile manipulation, underwater localization, and task allocation mechanisms for underwater multi-robot operations in marine environments.

References

[1] R. Pi, P. Cieślak, P. Ridao, and P. J. Sanz, "Twinbot: Autonomous underwater cooperative transportation," *IEEE Access*, vol. 9, pp. 37 668–37 684, 2021.

[2] G. Marani, S. K. Choi, and J. Yuh, "Underwater autonomous manipulation for intervention missions auvs," *Ocean Engineering*, vol. 36, no. 1, pp. 15–23, 2009, autonomous Underwater Vehicles. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002980180800173X

[3] P. J. Sanz, P. Ridao, G. Oliver, C. Melchiorri, G. Casalino, C. Silvestre, Y. Petillot, and A. Turetta, "Trident: A framework for autonomous underwater intervention missions with dexterous manipulation capabilities," *IFAC Proceedings Volumes*, vol. 43, no. 16, pp. 187–192, 2010, 7th IFAC Symposium on Intelligent Autonomous Vehicles. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667016350546

[4] M. Grimaldi, P. Cieslak, E. Ochoa, V. Bharti, H. Rajani, I. Carlucho, M. Koskinopoulou, Y. R. Petillot, and N. Gracias. (2025) Stonefish: Supporting machine learning research in marine robotics. [Online]. Available: https://arxiv.org/abs/2502.11887

[5] E. Morgan and C. Barbalata. (2025, Mar.) Underwater vehicle & manipulator simulator and hardware framework. [Online]. Available: https://doi.org/10.5281/zenodo.15085171

[6] E. C. Gezer, M. Zhou, L. Zhao, and W. McConnell, "Working toward the development of a generic marine vehicle framework: Ros-mvp," in *OCEANS 2022, Hampton Roads*, pp. 1–5.

[7] N. Palomeras, A. El-Fakdi, M. Carreras, and P. Ridao, "Cola2: A control architecture for auvs," *IEEE Journal of Oceanic Engineering*, vol. 37, no. 4, pp. 695–716, 2012.

[8] ArduSub Dev Team, "Ardusub and the ardupilot project," 2025, accessed: 2025-03-27. [Online]. Available: https://www.ardusub.com/

[9] Z. Huang, M. Buchholz, M. Grimaldi, H. Yu, I. Carlucho, and Y. R. Petillot, "Urobench: Comparative analyses of underwater robotics simulators from reinforcement learning perspective," in *OCEANS 2024 - Singapore*, 2024, pp. 1–8.

[10] B. Bingham, C. Agüero, M. McCarrin, J. Klamo, J. Malia, K. Allen, T. Lum, M. Rawson, and R. Waqar, "Toward maritime robotic simulation in gazebo," in *OCEANS 2019 MTS/IEEE SEATTLE*. IEEE, 2019, pp. 1–10.

[11] E. Coumans, "Bullet physics simulation," in *ACM SIGGRAPH 2015 Courses*, 2015, p. 1.

[12] M. M. Zhang, W.-S. Choi, J. Herman, D. Davis, C. Vogt, M. McCarrin, Y. Vijay, D. Dutia, W. Lew, S. Peters, and B. Bingham, "Dave aquatic virtual environment: Toward a general underwater robotics simulator," in *2022 IEEE/OES Autonomous Underwater Vehicles Symposium (AUV)*, pp. 1–8.

[13] M. M. M. Manhães, S. A. Scherer, M. Voss, L. R. Douat, and T. Rauschenbach, "Uuv simulator: A gazebo-based package for underwater intervention and multi-robot simulation," in *OCEANS 2016 MTS/IEEE Monterey*, pp. 1–8.

[14] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science robotics*, vol. 7, no. 66, p. eabm6074, 2022.

[15] E. Palmer, C. Holm, and G. Hollinger, "Angler: An autonomy framework for intervention tasks with lightweight underwater vehicle manipulator systems," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6126–6132.

[16] E. C. Gezer, M. Zhou, L. Zhao, and W. McConnell, "Working toward the development of a generic marine vehicle framework: Ros-mvp," in *OCEANS 2022, Hampton Roads*, pp. 1–5.

[17] ArduPilot Dev Team, "Ardupilot," 2025, accessed: 2025-03-27. [Online]. Available: https://ardupilot.org/

[18] E. Potokar, S. Ashford, M. Kaess, and J. G. Mangelson, "Holoocean: An underwater robotics simulator," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 3040–3046.

[19] T. R. Player, A. Chakravarty, M. M. Zhang, B. Y. Raanan, B. Kieft, Y. Zhang, and B. Hobson, "From concept to field tests: Accelerated development of multi-auv missions using a high-fidelity faster-than-real-time simulator," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3102–3108.

[20] I. Vaughn and S. Suman, "ds_sim," Woods Hole Oceanographic Institution, 2025, code repository. [Online]. Available: https://bitbucket.org/whoidsl/ds_sim/src/master/

[21] D. Centelles, A. Soriano, J. V. Martí, R. Marin, and P. J. Sanz, "Uwsim-net: An open-source framework for experimentation in communications for underwater robotics," in *OCEANS 2019 - Marseille*, pp. 1–8.

[22] J. Bialkowski, S. Karaman, and E. Frazzoli, "Massively parallelizing the rrt and the rrt," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3513–3518.

[23] J. Pan, S. Chitta, and D. Manocha, "Fcl: A general purpose library for collision and proximity queries," in *2012 IEEE international conference on robotics and automation*. IEEE, 2012, pp. 3859–3866.

[24] S. Chitta, I. Sucan, and S. Cousins, "Moveit!" *IEEE robotics & automation magazine*, vol. 19, no. 1, pp. 18–19, 2012.

[25] Dawson-Haggerty et al., "trimesh." [Online]. Available: https://trimesh.org/

[26] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, "The pinocchio c++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *2019 IEEE/SICE International Symposium on System Integration (SII)*. IEEE, 2019, pp. 614–619.

[27] A. Agrawal, P. Mayer, Z. Kingston, and G. A. Hollinger, "Constrained nonlinear kaczmarz projection on intersections of manifolds for coordinated multi-robot mobile manipulation," in *IEEE International Conference on Robotics and Automation*, to appear.

[28] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012, https://ompl.kavrakilab.org.

[29] V. Poupet, "Simulating 3d cellular automata with 2d cellular automata," in *Mathematical Foundations of Computer Science 2004*, J. Fiala, V. Koubek, and J. Kratochvíl, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 439–450.