2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

# From Concept to Field Tests: Accelerated Development of Multi-AUV Missions Using a High-Fidelity Faster-than-Real-Time Simulator

Timothy R. Player*, Arjo Chakravarty†⋆, Mabel M. Zhang†, Ben Yair Raanan‡,
Brian Kieft‡, Yanwu Zhang‡, and Brett Hobson‡

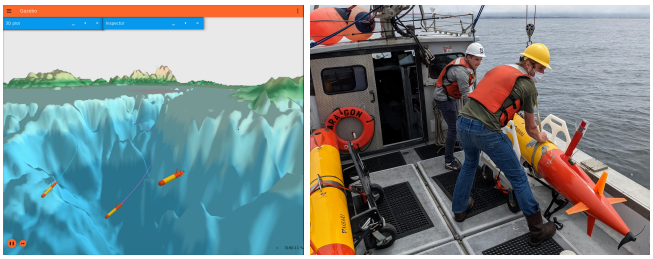
***Abstract*— We designed and validated a novel simulator for efficient development of multi-robot marine missions. To accelerate development of cooperative behaviors, the simulator models the robots' operating conditions with moderately high fidelity and runs significantly faster than real time, including acoustic communications, dynamic environmental data, and high-resolution bathymetry in large worlds. The simulator's ability to exceed a real-time factor (RTF) of 100 has been stress-tested with a robust continuous integration suite and was used to develop a multi-robot field experiment.**

## I. Introduction

Autonomous robots are a mainstay of modern ocean exploration. Robots collect measurements in situ at larger scales, with greater precision, and at significantly lower cost than traditional ship operations. Further, multi-robot systems in long-duration deployments can collect larger-scale data more efficiently than single Autonomous Underwater Vehicles (AUVs) [1], [2], [3]. However, complex underwater multi-robot systems require rigorous validation in simulation and in the field to function reliably.

Developing long-duration multi-AUV missions is challenging because many failure modes could jeopardize success in week or month-long deployments. Underwater platforms must operate reliably with limited communication, constrained power, and uncertain localization. Failures can result in the loss of expensive payloads and data. The risk increases with multiple agents with more points of failure. Simulation plays a key role by allowing code to be tested before high-stake deployments. However, existing simulators are too slow or do not support multiple vehicles.

We designed a simulation stack, LRAUV Sim[1], for developing complex multi-AUV missions. The simulator can be extended to incorporate arbitrary propeller-driven underwater vehicles, but presently models Long-Range Autonomous Underwater Vehicles (LRAUVs), a slender AUV that is regularly deployed in the real world by two institutions. LRAUV Sim extends the new Gazebo simulator [4] and models hydrodynamics, acoustic communications, and marine sensors at what we believe to be greater speed than previous simulators, while allowing scientific data to be visualized from a user-provided scalar field. LRAUV Sim enables a development paradigm for field robotics in which multiple action sequences can be quickly simulated to test failure cases and mission logic in complex systems.



(a) (b)

Fig. 1: (a) Multi-vehicle simulation. (b) LRAUVs during field deployment from R/V *Paragon* in Monterey Bay.

While real-world validation is still necessary for mission development, the ability to rapidly simulate diverse scenarios allows practitioners to focus time in the field on fine-tuning missions to account for elements that are imperfectly modeled in simulation, such as subtle hydrodynamic behaviors and sensor and actuator characteristics, instead of discovering software bugs in mission logic or control flow in the field.

We validated the simulator through continuous integration (CI) tests and by developing, from simulation to successful field trials, a complex behavior to maintain observation by a multi-robot team. In the behavior, an AUV uses acoustic localization and communication to precisely replace another AUV, allowing the relieved vehicle to be recharged or redispatched: a monitoring technique relevant to ocean researchers.

Our contributions are as follows:

- Multi-robot faster-than-real-time (FTRT) marine simulation, with fastest RTF we know of
- Continuous controller integration for hydrodynamics validation based on solid theoretical foundations
- Synchronization of physics simulation time step with iterations of black-box controller
- Simulation-accelerated mission development and validation demonstrated in the real world
- In-simulator visualization of large-scale dense data, including dynamically interpolated science data, and high-resolution real-world bathymetry
- Software contribution accepted as native built-in features in a general simulator (the new Gazebo [4])

* This author is with *Oregon State University*, Corvallis, Oregon, USA. This work was conducted as a summer intern at *Monterey Bay Aquarium Research Institute*. playert@lifetime.oregonstate.edu.

† These authors are with *Open Robotics*, Mountain View, California, USA {arjo, mabel}@openrobotics.org.

⋆ This author is also with *Singapore University of Technology and Design*, Singapore 1007419@mymail.sutd.edu.sg.

‡ These authors are with *Monterey Bay Aquarium Research Institute*, Moss Landing, California, USA. {byraanan, bkieft, yzhang, hobson}@mbari.org.

[1] https://github.com/osrf/lrauv

## II. Related Work

As with most underwater simulators, LRAUV Sim provides typical building blocks, such as hydrodynamics, acoustic sensors, and marine-specific actuators. Different from existing simulators, LRAUV Sim is unique in its stress-tested FTRT speedup, field-tested multi-vehicle mission simulation, meticulously verified synchronization of physics time steps with the real control loop, continuous integration with the controller, and visualization tools to assist development.

In short, LRAUV Sim is a mission-driven validation tool that has been rigorously tested against a real controller and mission-validated at sea. Further, it is not only an underwater simulator, but also a demonstration of software modules that have been accepted as built-in features to a widely used open source general robotics simulator (the new Gazebo).

FTRT and multi-vehicle simulation are not always a high priority, because underwater simulation is often used for brief missions such as manipulation and inspection. FTRT is necessary for complex missions with long-term autonomy, which require validating event sequences that lead to failures. We measure FTRT by the RTF, which is 1 at real time, $< 1$ for slower, and $> 1$ for faster. Multi-vehicle simulation is necessary for tasks such as collaborative seafloor mapping [14] and environmental monitoring. To validate LRAUV Sim for multi-vehicle use cases, we tested for stable physics performance at RTF 100 for a single vehicle, and we measured the RTF while increasing the number of vehicles.

Table I compares several modern underwater simulations on features relevant to our contributions. UWSim [32] was among the first to extend the popular Gazebo-classic [33], [13] simulator with underwater capabilities, by implementing hydrodynamics for manipulation with intervention AUVs. UWSim was extended and studied in [17], [16], [34], [35].

In a newer generation, Unmanned Underwater Vehicle (UUV) Simulator [14] was developed beyond the basic hydrodynamics, sensors, and actuators for Gazebo-classic. Most related to our work, it simulated multi-vehicle seabed mapping, chemical plumes, and a particle concentration sensor. DAVE [12] extended UUV Simulator by adding a CUDA-enabled physics-based multi-beam sonar [36], water tracking for the Doppler Velocity Logger (DVL), dynamic tile loading for large-scale worlds, dual-arm manipulation, and other features. AUG Sim [37] simulated an FTRT glider. However, it did not demonstrate multiple vehicles.

| Underwater Extensions | Simulator Platforms | Multi-Agent | Acoustic Comms | Large Worlds |
|---|---|---|---|---|
| LRAUV Sim | (new) Gazebo [4] | ✓ | ✓ | ✓ |
| HoloOcean [5] | Unreal Engine [6] | ✓ | ✓ | ✗ |
| Aqua [7] | Unreal Engine [6] | ✗ | ✗ | ✗ |
| Stonefish [8] | Bullet Physics [9] | ✗ | ✗ | ✗ |
| URSim [10] | Unity [11] | ✗ | ✗ | ✗ |
| DAVE [12] | Gazebo-classic [13] | ✓ | ✗ | ✓ |
| UUV Sim [14] | Gazebo-classic [13] | ✓ | ✗ | ✗ |
| `ds_sim` [15] | Gazebo-classic [13] | ✓ | ✓ | ✓ |
| UWSim-NET [16] | Gazebo-classic [13] | ✓ | ✓ | ✗ |
| Freefloating [17] | Gazebo-classic [13] | ✗ | ✗ | ✗ |
| Suzuki et al. [18] | Choreonoid [19] | ✗ | ✗ | ✗ |
| MARS [20] | Bullet Physics [9] | ✓ | ✓ | ✗ |
| UW MORSE [21] | MORSE [22] | ✗ | ✓ | ✗ |
| Rock [23] | Gazebo-classic [13] | ✗ | ✗ | ✗ |
| Melman et al. [24] | | ✗ | ✓ | ✗ |
| uSimMarine [25] | MOOS-IvP [26] | ✗ | ✗ | ✗ |
| Sehgal et al. [27] | USARSim [28] | ✓ | ✓ | ✗ |
| Song et al. [29] | POSIX | ✓ | ✓ | ✗ |
| DVECS [30] | Yuh et al. [31] | ✓ | ✗ | ✗ |

TABLE I: Comparison of underwater simulation packages

LRAUV Sim is based on the new Gazebo [4], which is entirely rewritten and completely different from Gazebo-classic [13]. Similar to [14] and [12], state-of-the-art holistic underwater simulators, we implemented hydrodynamics using Fossen's equations [38]. However, though FTRT was possible in Gazebo-classic, these works did not rigorously measure the extent of speedup possible, especially for multiple vehicles. To do so, we vary the physics timestep while using an automated test suite to measure navigation trajectory errors. For instance, we numerically verify that the modeled vehicle can execute motions (both open-loop and trajectories commanded by a real controller) within deltas from a reference trajectory as the real-time factor increases. This test suite is explained further in Section V. UUV Simulator and DAVE did not have continuous integration test suites validated with a real controller.

Further, their worlds and science data were not demonstrated at as large a scale as LRAUV Sim (150 miles). Large worlds are inherent to marine environments and pose a simulation performance challenge. Note that the home-brewed dynamic tiling for large worlds, which DAVE [12] adapted from `ds_sim` [15], is a built-in feature (Levels) in the new Gazebo, which LRAUV Sim demonstrated through high-definition bathymetry.

HoloOcean [5] was based on Unreal Engine [6], a 3D graphics game engine. Targeting the visuals for SLAM applications, it did not emphasize accurate hydrodynamics. Gazebo is known to have more accurate physics than game engines, as it prioritizes high-fidelity physics engines for robotics. We modeled all the actuators of a robot deployed at sea more than 300 days a year. Thus, hydrodynamics' impact on vehicle behavior was important. Using an automated test suite, we rigorously tested the interactions between actuators and hydrodynamics.

Conventionally, simulation data are either plotted offline from log files, or visualized dynamically in a separate tool. We developed native GUI components to overlay point clouds directly in the simulation, making visualization and debugging more efficient.

We emphasize that though our test bed was a specific vehicle, our software contributions have been accepted as native features in the open source Gazebo simulator and are available as individual modules to general robotics. Though we synchronized the physics time step to a specific controller, we demonstrated that low-level coupling enables simulation to be used in a high-fidelity manner to quickly validate long-term mission success using real-world controllers.

## III. Dynamics Formulation

Basic dynamics and common actuators are modeled using standard methods. We summarize here for completeness.

### A. General Hydrodynamics

We model basic hydrodynamics using Fossen's equations [38]:

$$M\ddot{x} + D(\dot{x})\dot{x} + C(\dot{x})\dot{x} = F \tag{1}$$

where $M$ is the added mass matrix, $D(\dot{x})$ is the damping matrix, $C(\dot{x})$ is the Coriolis matrix, and $\dot{x}$ is the velocity six-vector. Ocean current is approximated by adding a term $v_c$ to the velocity:

$$M\ddot{x} + D(\dot{x} - v_c)(\dot{x} - v_c) + C(\dot{x} - v_c)(\dot{x} - v_c) = F \tag{2}$$

In our case, $C(\dot{x})$ is small, so $D(\dot{x})$ is the main contribution. We only simulate the diagonal terms:

$$\begin{aligned} D(\dot{x}) = &- \operatorname{diag}\{X_u, Y_x, Z_w, K_p, M_q, N_r\} \\ &- \operatorname{diag}\{X_{u|u|}|u|, Y_{v|v|}|v|, Z_{w|w|}|w|, \\ &K_{p|p|}|p|, M_{q|q|}|q|, N_{r|r|}|r|\} \end{aligned} \tag{3}$$

### B. Thruster

We model the thruster force $F$ after [38] as well:

$$F = \rho D^4 K_T(J_0) n|n| \tag{4}$$

where $D$ is the fin's diameter. $n$ is rotations per second, and $K_T(J_0)$ is the propeller's coefficient, empirically determined.

### C. Lift on Fins

Fins are common on gliders. When the fins are angled, a lifting force is generated, which causes the vehicle to rotate. These are modeled using the classic lift equation:

$$F = \frac{1}{2} C_l \rho v^2 A \tag{5}$$

where $C_l$ is the lift coefficient, $v$ is the velocity, and $A$ is the area. We model the value of $C_l$ as a piece-wise linear function. $C_l$ varies linearly with the angle of attack until the stall angle is reached.

## IV. Simulator Features

Various aspects relevant to ocean science mission development are modeled. We use the DART physics engine [39], but others can be used. Data such as temperature and salinity can be queried and visualized, high-resolution bathymetry can be dynamically loaded, and inter-vehicle communication is modeled. Code from black-box controllers is integrated with the simulator through inter-process communication. These features provide a framework for multi-robot mission development.

### A. Volumetric Data

Historic data, such as ocean current, salinity, temperature, and chlorophyll, that vary spatially with arbitrary resolution can be loaded and queried. For roboticists, changes in current, temperature, and density affect vehicle dynamics and communications. For biologists, they affect ocean ecology.

*1) Environmental Data Interpolation:* Environmental data like current, temperature, or chlorophyll are often generated by predictive models at a spatial scale of kilometers or degrees, while the spatial scale relevant to simulating AUV dynamics and sensors is local. To achieve local observations, we store environmental data in a sparse grid and trilinearly interpolate among the nearest points on the grid.

The main challenge is to efficiently query grid data. The grid is not uniformly spaced, due to conversion errors between spherical coordinates and Mercator projections. Often, the depth axis has logarithmic spacing between data points. This makes modeling it as a simple 3D array precarious.

To handle these inconsistencies in resolution, we use a special data structure. The axes are indexed as red-black trees in memory and queried using binary search. Binary search on this axis returns the closest points in $O(\log n)$ time, where $n$ is the number of points on the axis. If the value is found, it is returned; otherwise, it returns the closest 2, 4, or 8 points, depending on whether the query lies on a plane or in between points in the grid. We then utilize linear, bilinear or trilinear interpolation to estimate the value of the current point.

Linear interpolation is further used between values that change in time, to estimate values between time steps. We extend this to time-varying 3D data by loading and interpolating between the previous and next data time step. Searching a 1 GB data file takes on the order of 10–100 ns per query on a laptop with a AMD Ryzen 5900HX CPU.

### B. GUI and Visualization Tools

*1) Point Cloud Visualization of Environmental Data:* Data visualization aids in development of oceanographic sampling missions. Fig. 2(a) shows temperature distribution in Monterey Bay from historical data. We provide real data samples for chlorophyll, temperature, salinity, and ocean current. Multiple types of data can be overlaid simultaneously in configurable colors. The implementation allows the user to easily extend to more data types.

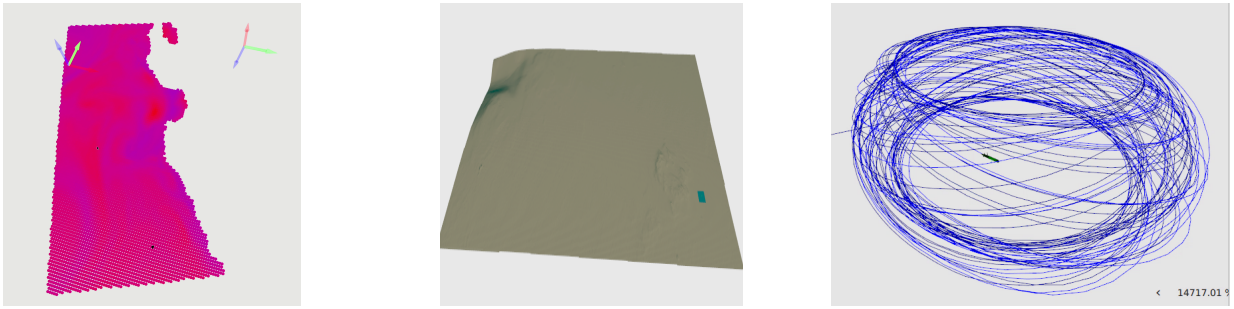
*2) 3D Trajectory Plot:* We perform 3D visualization of the robot's trajectory natively in the simulator, useful for debugging control and navigation algorithms. Fig. 2(c) shows the robot's trajectory in a Circling Yo-Yo mission.

### C. High-Resolution Bathymetry

Real-world high-resolution (1 m) bathymetry (Fig. 2(b)) collected by vehicles can be loaded while maintaining simulation performance. We split the bathymetry into tiles and only load tiles directly below the vehicle. This allows for modeling large-scale terrain while maintaining low latency, useful for developing algorithms that use bathymetry to navigate or to determine sampling locations.

### D. Acoustic Communication

In a modular architecture, we use a delay-based protocol with a range limited drop-off model. A simple model meets our needs, but the modular architecture allows for more complex acoustic models. The acoustic communication model also enables an acoustic localization plugin, allowing robots to query the relative position of an acoustic beacon.



(a) (b) (c)

Fig. 2: Features relevant to robot missions for oceanography, navigation, or controls. (a) Visualization of large-scale dense data on sea temperature in coastal California, from San Francisco Bay at the top, Monterey Bay in top middle, and near San Simeon at the bottom, spanning ∼150 miles. Resolution was variable, finest being 5 m in z depth. Red: high temperature; blue: low. (b) Detailed bathymetry, with 1 m resolution, of Portuguese Ledge in Monterey Bay. (c) 3D plot of robot trajectory in the Circling Yo-Yo mission.

### E. Integration with Controller

The vehicle control loop is synchronized with the simulated physics loop by reading a simulation timestamp from the simulator state. As the physics time step, or simulation speed, goes faster in real-world time, the control loop issues commands faster.

To achieve FTRT, large time steps are used, e.g., 20 ms for navigation as opposed to the Gazebo default of 1 ms. Lower physics fidelity that accompanies larger time steps is compensated by a closed-loop controller [40] and verified to be within acceptable delta bounds using automated tests.

Fig. 3 shows the RTF performance while increasing the number of vehicles, up to a physics time step of 30 ms. The RTF is inversely proportional to the number of vehicles. On a single vehicle, we achieved RTF $> 100$, or 100 times FTRT, at a time step of 30 ms. The robot was still able to perform the Circling Yo-Yo mission shown in Fig. 2(c), indicating sufficient physics fidelity for mission-level verification.

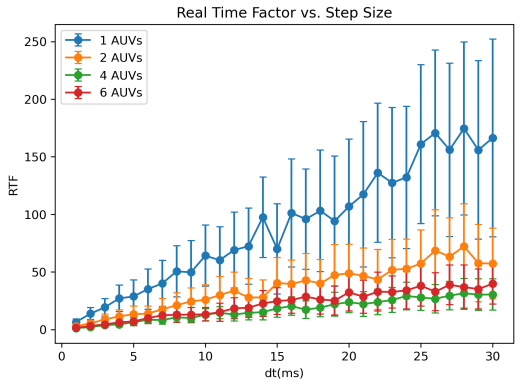



Fig. 3: RTF increased with physics step size and was inversely proportional to the number of vehicles during a Circling Yo-Yo mission.

## V. Validation through Continuous Integration

In implementing the hydrodynamics, lift-drag, buoyancy, and other fundamental factors that affect the physics of underwater vehicles, it was necessary to validate each actuator and its physics effect individually, before combining all of them. Otherwise, it was impossible to track down which part of the theoretical formulation was incorrectly implemented.

This was carried out by CI testing. Each time the simulator is changed, such as through updating the source code, an automated test suite is rerun to verify correctness, to make sure new additions do not break existing behaviors. The suite consists of several types of tests:

- Unit tests of basic physics to verify individual forces (e.g., buoyancy on a geometric primitive)
- Unit tests of physics with simple actuator commands (e.g., a translation on the battery prismatic joint), for vehicle model correctness
- Integration tests of the simulated vehicle with real-world controller on individual actuators (e.g., holding a constant value on the rudder to go in a circle), to verify synchronized control loop time steps
- Integration tests at the mission level (e.g., Circling Yo-Yo)

*1) Tests of the Vehicle Model without the Controller:* These tests explore a minimum set of invariants that a simulated underwater vehicle should satisfy. We test these invariants and compare them to real world constants. Though the test tolerances are tuned for one vehicle, the components tested, such as hydrostatics and hydrodynamics, and the actuators tested, such as propellers and fins, are common among underwater gliders.

*a) Hydrostatics:* We ensure the vehicle model is neutrally buoyant and has form stability (i.e., it tends to right itself). For neutral buoyancy, we test that the vehicle does not move when starting at rest and external forces are absent.

For stability, we enable buoyancy and gravity without hydrodynamic drag and start the vehicle at some offset angle $\theta_0$. The off-center buoyancy results in a restoring moment given by

$$\tau(\theta) = \rho g V C_b \sin\theta \tag{6}$$

where $V$ is volume, $C_b$ is the vertical displacement of the center of buoyancy from the center of mass, and $g$ is the gravitational acceleration. We check that this results in an oscillation between $-\theta_0$ and $\theta_0$, which is expected by conservation of energy.

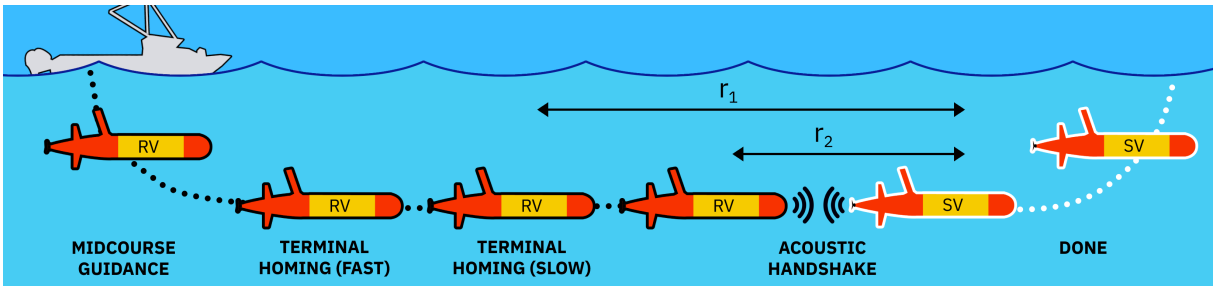



Fig. 4: Hot-bunking mission phases. Once deployed, the relief vehicle (RV) begins a three-stage homing sequence toward the drifting sampling vehicle (SV): after achieving a GPS waypoint during *Midcourse Guidance*, the RV switches to an acoustic *Terminal Homing* stage with fast speed before slowing down once range is reduced to $r_1$. Once a range of $r_2$ is attained, a two-way *Acoustic Handshake* occurs where the RV commands the SV to surface and the SV acknowledges the command and surfaces. In the *Done* phase, the RV begins sampling.

*b) Hydrodynamics:* To verify hydrodynamics, we check that the vehicle velocity is realistically damped. Since the hydrodynamics equations (Section III-A) consist of one acceleration- and two velocity-dependent terms, it is necessary to test the terms separately.

$$\underbrace{M\ddot{x}}_{\text{Acceleration-dependent term}} + \underbrace{D(\dot{x})\dot{x} + C(\dot{x})\dot{x}}_{\text{Velocity-dependent terms}} \tag{7}$$

For the acceleration-dependent term, we monitor response to oscillations. For a naturally stable vehicle, the acceleration term should damp oscillations. We check this by introducing a position-dependent force to induce oscillations. This should result in a scenario similar to simple harmonic oscillation; an unstable oscillation would fail the automated test.

We check the velocity-dependent terms by simulating the vehicle at maximum thrust. Given a constant force, the equations of motion should converge [38] to a certain terminal velocity. Experimentally, we know this value to be 1 m/s. We verify that a similar terminal velocity is reached.

*c) Vertical Fins (Rudder):* Fins introduce forces that cause the vehicle to pitch or yaw. We test the horizontal and vertical fins separately. For the vertical fins, we perform a unit test by moving the vehicle in a circular trajectory. The force exerted by the fins is given by Eqn. (5), which we equate to the expression for circular motion:

$$\frac{1}{2}C_l\rho v^2 A = \frac{mv^2}{r} \tag{8}$$

This further simplifies to:

$$r = \frac{2m}{C_l\rho A} \tag{9}$$

While we could check the radius of curvature, floating point errors may accumulate. Hence, we check that the yaw rate during circular motion is a constant value of $\frac{2\pi r}{v}$.

*d) Horizontal Fins (Elevator):* When pitched, they exert a force that pitches the vehicle. The force is counteracted by the restoring moment in Eqn. (6). We model the torque $\tau(\theta)$ and the fin's displacement $d$ from the center of buoyancy:

$$\tau(\theta) = \rho g V \sin\theta - \frac{1}{2}C_l\rho v^2 A d \tag{10}$$

Solving Eqn. (10) for $\tau(\theta) = 0$ gives the min/max pitch the vehicle can have when traveling at speed $v$. At 1 m/s, the vehicle pitches a maximum of $\pm 20°$. This is another invariant used to validate the model.

*e) Battery Mass Shifter:* A vehicle-specific actuator, the mass shifter, trims the center of mass by moving the battery to pitch the vehicle [3]. To test it, we translate the mass to a fixed location. This introduces a torque, which is countered by the buoyancy force. This is similar to horizontal fins (Eqn. (10)), except here $d$ is variable, and $m_{\text{shifter}}$ represents the mass shifter's weight:

$$\tau(\theta) = \rho g V C_b \sin\theta - m_{\text{shifter}}\, g\, d\, \cos(\theta) \tag{11}$$

Solving for $\tau(\theta) = 0$ gives the equilibrium pitch angle for a given $d$. This value is verified against an empirical reference.

*2) Integration Tests with the Controller:* We developed integration tests with the real vehicle's controller, testing that each component works correctly. At the unit integration level, for instance, we check that shifting the mass causes the vehicle to pitch. At the mission integration level, for instance, the vehicle performs a Circling Yo-Yo mission. This involves controlling the vehicle to circle downward toward the sea floor and then resurface multiple times. We check that the vehicle's direction of travel and yaw rate are sufficiently close to the reference trajectory.

## VI. Experimental Results

We validated the simulator by developing and testing a real multi-robot mission called "acoustic hot-bunking," where an AUV uses acoustic localization to replace another AUV at its precise location. Hot-bunking extends the duration that a specific water mass can be monitored, which is needed to study drifting microbial populations [2], phytoplankton patch formation [41], and the impact of floating sediment on cellular life [42].

This mission is challenging to develop because failures in either robot could occur at each of the five mission phases shown in Fig. 4, requiring simulated tests to ensure that the two-robot system can recover from errors in each phase. The recovery behaviors would be impractical to test in a real-time simulated deployment of over two hours. We used the simulator's capability for multi-robot simulation, acoustic communications, and integration with the real-world controller to test these scenarios in FTRT.

### A. Mission Definition

In the mission, the sampling vehicle (SV) is replaced by the relief vehicle (RV) using the sequence of phases shown in Fig. 4. To ensure that the SV is replaced in its exact location despite odometry drift in a GPS-denied environment [43], we perform closed-loop homing using relative localization from a directional acoustic transponder (DAT). Homing is done using a pure pursuit controller, such that

$$\theta_{des} \Leftarrow \theta_{SV \leftarrow RV,\, meas} \tag{12}$$

where $\theta_{des}$ is the commanded yaw and $\theta_{SV \leftarrow RV,\, meas}$ is the azimuth from the RV to the SV, in a global frame, measured by the RV's acoustic transponder.

### B. Mission Validation

Validation involved testing the nominal event sequence, where each mission phase worked as intended, and the off-nominal event sequences where failures must be accommodated. To verify the nominal event sequence, we developed unit tests for each phase to ensure successful operation.

However, in field deployments, vehicle capability could be compromised by environmental factors (e.g., seaweed entangling a propeller) or software factors (e.g., a bug disabling one vehicle's acoustic modem). Consequently, we also simulated component failures and verified that appropriate timeout behaviors were triggered if components failed in any of the phases. FTRT simulation allowed each test to complete in minutes rather than over two hours in real time.

### C. Results

The hot-bunking mission was run on two vehicles: *Brizo*, the SV, and *Tethys*, the RV (Fig. 1(b)). Both were equipped with Teledyne Benthos DATs which provided relative acoustic localization and underwater communications. The vehicles were deployed from the R/V *Paragon* for a two-day saltwater deployment above Portuguese Ledge in Monterey Bay, during which three hot-bunking trials were executed via Iridium satellite connection.

The field trials enabled high-level parameters to be tuned from field data that was not possible in simulation. The first trial inspired an adjustment of the slow terminal homing speed from $0.5$ m/s to $0.8$ m/s, after it was observed that a low speed was insufficient to maintain control authority in the real system.

In the second trial, the two-way acoustic handshake (Fig. 4) was only partially successful: the RV did not receive an acknowledgement signal from the SV because the acoustic bandwidth was overshadowed by localization requests from the DAT. In the next trial, the frequency of acoustic localization was modified to prevent this issue, which was not present in simulation. The third trial successfully demonstrated all phases of the hot-bunking sequence.

*1) Acoustic Homing:* Fig. 5 shows the trajectory of the DAT fix in the RV frame during the third trial, advancing from a range of $500$ m to within the success radius of $20$ m, while keeping the SV approximately straight ahead of the RV in both simulated and real trials. The decreasing trend

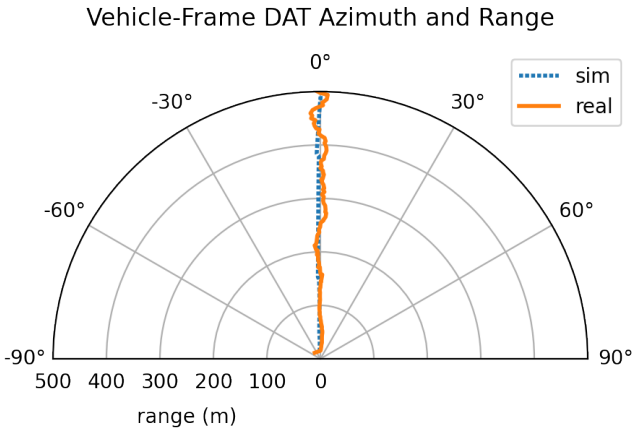



Fig. 5: Polar trace of the $xy$-plane DAT vector in the relief vehicle's frame, in simulation and reality. The range to the sampling vehicle decreased until the $20$ m success criteria.

in range in both simulated and real trajectories indicates the successful operation of the control law.

The real trajectory (orange) displays more oscillation than the simulated trajectory (dotted blue). The oscillation in DAT azimuth can be attributed to hydrodynamic forces in the ocean inducing low-amplitude yawing motions to a greater extent than in simulation.

*2) Acoustic Handshake:* In the third trial, the acoustic handshake was successfully carried out: the RV sent a "relieved-of-duty" transmission; the SV received and acknowledged it; the RV registered the acknowledgement and began scientific sampling, and the SV ascended to the surface. The acoustic communications behaviors developed in simulation were successful on the real system.

## VII. Discussion

LRAUV Sim enabled fast iteration during mission development, to quickly bring a novel and impactful AUV behavior to fruition. While two robots constitute the minimum of a multi-robot system, the benefits of FTRT simulation for our field trials should scale to greatly benefit larger fleets. The simulations preceded the field trials, so that development time in the field was spent where it should be: tuning mission parameters based on field data rather than troubleshooting software components and mission control flow.

By supporting development with simulations of many event sequences, we caught multiple bugs in control flow, including ones infeasible to identify without the broad test coverage enabled by FTRT simulation – a crucial benefit to larger fleets with more failure cases.

## VIII. Acknowledgements

We are grateful for support from the David and Lucile Packard Foundation. This work would not have been possible without the team at Open Robotics and Ekumen Labs.

## References


[1] N. D. Zarokanellos, D. L. Rudnick, M. Garcia-Jove, B. Mourre, S. Ruiz, A. Pascual, and J. Tintoré, "Frontal Dynamics in the Alboran Sea: 1. Coherent 3D Pathways at the Almeria-Oran Front Using Underwater Glider Observations," *Journal of Geophysical Research: Oceans*, vol. 127, no. 3, 2022.

[2] Y. Zhang, J. P. Ryan, B. W. Hobson, B. Kieft, A. Romano, B. Barone, C. M. Preston, B. Roman, B.-Y. Raanan, D. Pargett, *et al.*, "A system of coordinated autonomous robots for Lagrangian studies of microbes in the oceanic deep chlorophyll maximum," *Science Robotics*, vol. 6, no. 50, 2021.

[3] B. W. Hobson, J. G. Bellingham, B. Kieft, R. McEwen, M. Godin, and Y. Zhang, "Tethys-class long range AUVs - extending the endurance of propeller-driven cruising AUVs from days to weeks," in *2012 IEEE/OES Autonomous Underwater Vehicles (AUV)*, 2012, pp. 1–8.

[4] Open Source Robotics Foundation, "Gazebo." [Online]. Available: https://gazebosim.org

[5] E. Potokar, S. Ashford, M. Kaess, and J. G. Mangelson, "HoloOcean: An Underwater Robotics Simulator," in *ICRA*, 2022.

[6] Epic Games, "Unreal engine." [Online]. Available: https://www.unrealengine.com

[7] T. Manderson, I. Karp, and G. Dudek, "Aqua Underwater Simulator," in *IROS workshop New Horizons for Underwater Intervention Missions: from Current Technologies to Future Applications*, 2018.

[8] P. Cieślak, "Stonefish: An advanced open-source simulation tool designed for marine robotics, with a ROS interface," in *OCEANS*, 2019.

[9] E. Coumans, "Bullet physics simulation," in *ACM SIGGRAPH*, New York, NY, USA, 2015.

[10] P. Katara, M. Khanna, H. Nagar, and A. Panaiyappan, "Open Source Simulator for Unmanned Underwater Vehicles using ROS and Unity3D," in *IEEE Underwater Technology*, 2019.

[11] "Unity3d." [Online]. Available: https://www.unity.com

[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 *Symposium on Autonomous Underwater Vehicle Technology (AUV)*. IEEE, 2022.

[13] Open Source Robotics Foundation, "Gazebo-classic." [Online]. Available: https://classic.gazebosim.org

[14] 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*. IEEE, sep 2016.

[15] I. Vaughn and S. Suman, "Deep Submergence `ds_sim`." [Online]. Available: https://bitbucket.org/whoidsl/ds_sim/src/master/

[16] D. Centelles, A. Soriano-Asensi, J. V. Martí, R. Marín, and P. J. Sanz, "Underwater Wireless Communications for Cooperative Robotics with UWSim-NET," *Applied Sciences*, vol. 9, no. 17, 2019.

[17] O. Kermorgant, "A Dynamic Simulator for Underwater Vehicle-Manipulators," in *International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*. Springer, 2014, pp. 25–36.

[18] K. Suzuki and K. Kawabata, "Development of a Simulator for Underwater Reconnaissance Tasks by Utilizing Remotely Operated Robots," in *IEEE/SICE International Symposium on System Integration (SII)*, 2020.

[19] S. Nakaoka, "Choreonoid: Extensible Virtual Robot Environment Built on an Integrated GUI Framework," in *IEEE/SICE International Symposium on System Integration (SII)*, Dec. 2012, pp. 79–85.

[20] T. Tosik, J. Schwinghammer, M. J. Feldvoß, J. P. Jonte, A. Brech, and E. Maehle, "MARS: A simulation environment for marine swarm robotics and environmental monitoring," in *OCEANS*, 2016.

[21] E. H. Henriksen, I. Schjølberg, and T. B. Gjersvik, "UW MORSE: The underwater Modular Open Robot Simulation Engine," in *AUV*, 2016.

[22] G. Echeverria, N. Lassabe, A. Degroote, and S. Lemaignan, "Modular open robots simulation engine: MORSE," in *ICRA*, 2011.

[23] T. Watanabe, G. Neves, R. Cerqueira, T. Trocoli, M. Reis, S. Joyeux, and J. Albiez, "The Rock-Gazebo Integration and a Real-Time AUV Simulation," in *IEEE Latin American Robotics Symposium and Brazilian Symposium on Robotics (LARS-SBR)*, 2015.

[24] S. Melman, A. Pavin, V. Bobkov, and A. Inzartsev, "Distributed simulation framework for investigation of autonomous underwater vehicles' real-time behavior," in *OCEANS*, 2015.

[25] M. Benjamin, "uSimMarine: Basic vehicle simulation," 2018. [Online]. Available: https://oceanai.mit.edu/ivpman/pmwiki/pmwiki.php?n=IvPTools.USimMarine

[26] "MOOS-IvP." [Online]. Available: http://moos-ivp.org

[27] A. Sehgal, D. Cernea, and A. Birk, "Modeling Underwater Acoustic Communications for Multi-Robot Missions in a Robotics Simulator," in *OCEANS*, 2010.

[28] S. Carpin, M. Lewis, J. Wang, S. Balakirsky, and C. Scrapper, "USARSim: a robot simulator for research and education," in *ICRA*, 2007.

[29] F. Song, P. E. An, and A. Folleco, "Modeling and simulation of autonomous underwater vehicles: design and implementation," *IEEE Journal of Oceanic Engineering (JOE)*, 2003.

[30] S. Choi, S. Menor, and J. Yuh, "Distributed virtual environment collaborative simulator for underwater robots," in *IROS*, 2000.

[31] J. Yuh, V. Adivi, and S. Choi, "Development Of A 3D Graphic Test Platform For Underwater Robotic Vehicles," in *The Second International Offshore and Polar Engineering Conference*, no. ISOPE-I-92-169, San Francisco, California, USA, June 1992.

[32] M. Prats, J. Pérez, J. J. Fernández, and P. J. Sanz, "An open source tool for simulation and supervision of underwater intervention missions," in *IROS*, 2012.

[33] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. IEEE, 2004, pp. 2149–2154.

[34] J. J. Fernández, J. Pérez, A. Peñalver, J. Sales, D. Fornas, and P. J. Sanz, "Benchmarking using UWSim, Simurv and ROS: An autonomous free floating dredging intervention case study," in *OCEANS*, 2015.

[35] D.-H. Gwon, J. Kim, M. H. Kim, H. G. Park, T. Y. Kim, and A. Kim, "Development of a side scan sonar module for the underwater simulator," in *IEEE Ubiquitous Robots and Ambient Intelligence (URAI)*, 2017.

[36] W.-S. Choi, D. R. Olson, D. Davis, M. Zhang, A. Racson, B. Bingham, M. McCarrin, C. Vogt, and J. Herman, "Physics-Based Modelling and Simulation of Multibeam Echosounder Perception for Autonomous Underwater Manipulation," *Frontiers in Robotics and AI*, vol. 8, 2021.

[37] W.-S. Choi, B. Bingham, and R. Camilli, "Faster-than-real-time Hybrid Automotive Underwater Glider Simulation for Ocean Mapping," *Journal of the Korean Society of Marine Environment and Safety*, 2022.

[38] T. I. Fossen, *Guidance and Control of Ocean Vehicles*. Chichester: Wiley, 1994.

[39] J. Lee, M. X. Grey, S. Ha, T. Kunz, S. Jain, Y. Ye, S. S. Srinivasa, M. Stilman, and C. K. Liu, "DART: Dynamic Animation and Robotics Toolkit," *Journal of Open Source Software*, vol. 3, no. 22, p. 500, 2018.

[40] A. S. Willsky and I. T. Young, *Signals and Systems*. Prentice-Hall International, 1997.

[41] J. S. Jaffe, P. J. Franks, P. L. Roberts, D. Mirza, C. Schurgers, R. Kastner, and A. Boch, "A swarm of autonomous miniature underwater robot drifters for exploring submesoscale ocean dynamics," *Nature Communications*, vol. 8, no. 1, pp. 1–8, 2017.

[42] J. Ryan, M. McManus, R. Kudela, M. L. Artigas, J. Bellingham, F. Chavez, G. Doucette, D. Foley, M. Godin, J. Harvey, *et al.*, "Boundary influences on HAB phytoplankton ecology in a stratification-enhanced upwelling shadow," *Deep Sea Research Part II: Topical Studies in Oceanography*, vol. 101, pp. 63–79, 2014.

[43] O. J. Woodman, "An introduction to inertial navigation," University of Cambridge, Computer Laboratory, Tech. Rep., 2007.