# Aqua Underwater Simulator

Travis Manderson, Ian Karp, and Gregory Dudek

## I. Introduction

We have been considering the need for marine simulation of robotic vessels with a particular focus on vision-based sensing for shallow-water environments such as coral reefs.

The conditions required for deployment and testing of autonomous underwater vehicles are limited by geography, weather, and hardware to a much higher extent than vehicles designed for land, which limits research in the field. In order to more rapidly test new algorithms in a robust way, reliable and realistic simulators are necessary. The use of simulation systems in both terrestrial and marine robotics is widespread [1], [2], [3], but relatively few simulators are targeted to marine environments and provide the combination of ROS compatibility, extreme photo-realism, hydrodynamics and high performance that we seek to deliver.

In this paper we present our own underwater simulator which was created to run high fidelity and risk free experiments, tightly coupled with ROS, on a virtual version of the Aqua robot designed at McGill University. Our tools exploit a pre-existing gaming "engine" (simulator), allowing us to take advantage of it's photo-realistic rendering capabilities and built-in physics modeling for simulating experiments and evaluating algorithms with perfect ground truths. Additionally, the use of ROS2 [4] for inter-process communication within our simulated sensors and controllers allow for the replication of vision and control algorithms that run on real-world robots.

## II. Aqua Robot

The Aqua Robot [5] is a fully autonomous amphibious robot developed at McGill University used primarily to explore underwater environments and gather data with minimal disturbance of indigenous marine life. Using six independently controlled flippers for locomotion, the Aqua has incredible maneuverability in the water allowing it to safely pass over coral in shallow waters or explore a sandy bottom without physical contact. Aqua is typically equipped with multiple cameras and uses computer vision as a primary sensing modality for many kinds of deployment, such as coral reef monitoring [6].

### A. Sensors

Aqua relies on two front cameras and one rear camera for navigation, which requires robust visual sensing for collision avoidance, navigation, and gaze selection. Aqua also contains an inertial measurement unit (IMU) and depth sensor.

School of Computer Science, McGill University, Rm 318, 3480 Rue University, Montreal QC, H3A 0E9, Canada

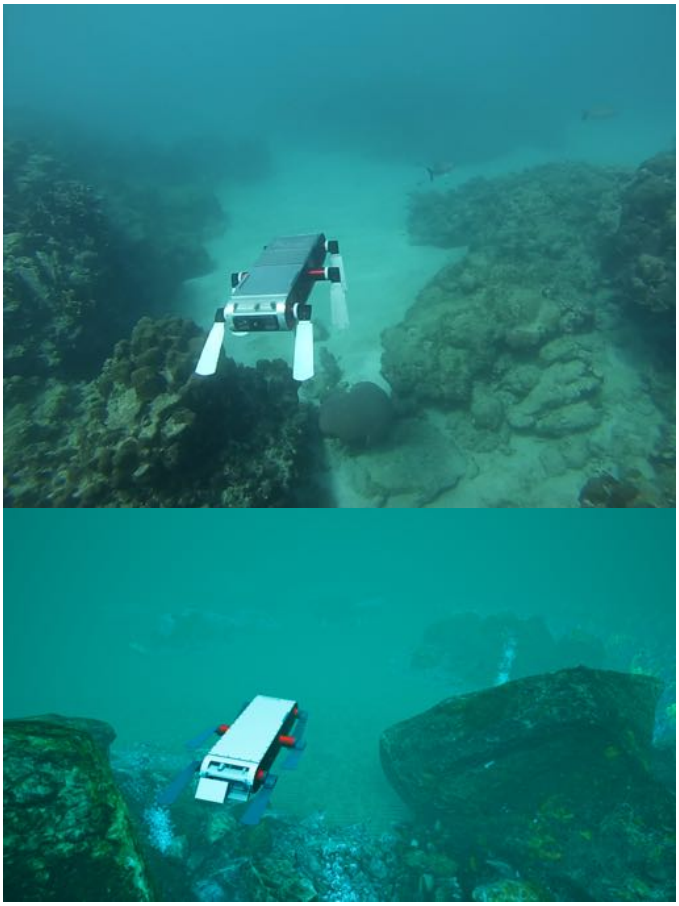

Fig. 1. Aqua in Barbados (top) vs simulated Aqua in synthetic environment (bottom)

## III. Aqua Simulation in Unreal Engine

Our simulator is built on Unreal Engine, from Epic Games, Inc., arguably the most advanced commercially available 3D simulator available today. Unreal Engine is widely used for "first person shooter" games, as well as for research on terrestrial robots [7] and is noted for the realism of its visual models, as demonstrated in Fig 1 and Fig 2.

With a vehicle like Aqua that relies so heavily on interpretation of visual data, the development of reliable algorithms for visual odometry is essential, though these algorithms are often difficult to test and debug.

### A. Architecture

Our simulator uses separate dependencies for OpenCV [8] and ROS2 to publish sensor data and receive inputs from external systems. Currently, we have an Aqua robot

Fig. 2. Sample images from the Aqua Underwater Simulator

in a simulated environment that can be controlled with a gamepad or through a script that interfaces with ROS2 or ROS1, allowing us to evaluate and perfect algorithms that can be later used on the physical robot. Since the simulator uses ROS2, use of the ros1_bridge is required when interfacing with ROS1 code, as seen in the *Development Option* configuration in Fig 3.

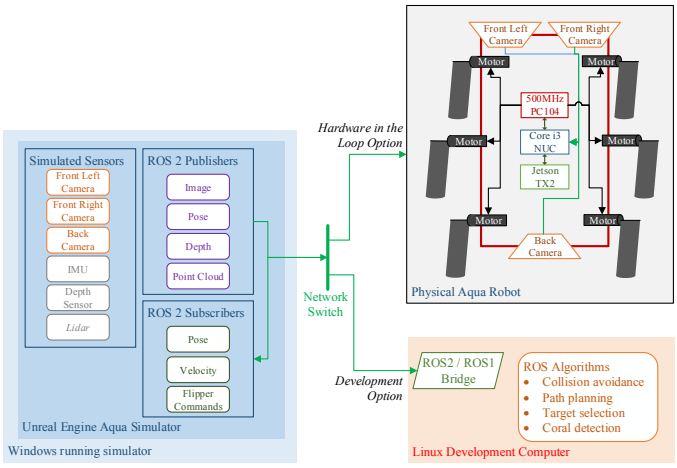


Fig. 3. Flowchart of the Aqua Simulator

## B. Simulated Sensors

In order to create a realistic simulator, several sensors were implemented that use ROS2 to allow interprocess communication between the vehicle and an external controller or system. These sensors include cameras with customized lens parameters, Inertial Measurement Unit (IMU), and depth. LiDAR has also been implemented in our simulator to replicate existing underwater laser scanners [9], though it is not yet being used on the physical Aqua. Additionally, the available ground truth makes it possible to approximate sonar sensors if needed.

## C. Physics

Our underwater simulator takes advantage of Unreal Engine's built-in physics engine for realistic control and interaction with the synthetic environment. Buoyancy and drag

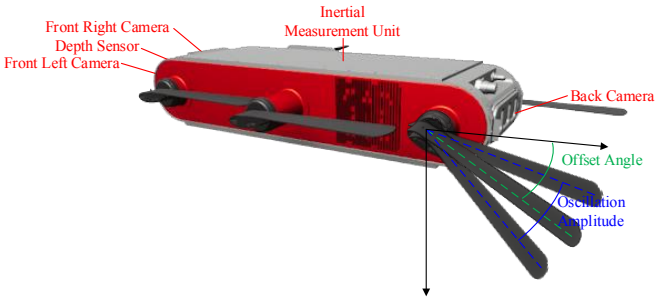


Fig. 4. Aqua robot showing sensor locations and the flipper thrust

forces are computed and applied to the Aqua at each physics tick as per [10], while Unreal Engine's highly optimized collision detection is leveraged for determining external forces. By using and improving established methods found in [11] we determine the thrust vectors of each independent flipper based on oscillation amplitude and offset angles demonstrated in Fig 4. These thrust vectors are applied to the simulated Aqua for navigation.

## D. Hardware in the Loop

With our simulator, it is possible to replace the real Aqua's cameras with the simulated sensors, as seen in the *Hardware in the Loop Option* configuration in Fig 3. Using ROS1 and ROS2 hooks in both the real-world Aqua and simulator control code respectively, the real Aqua robot's internal computers can be used to subscribe to the simulated images and send control information to the simulated Aqua robot.

## E. Dataset Creation

The realistic visuals of Unreal Engine also allow for the creation of photo-realistic datasets with perfect ground truth to supplement or replace real-world data. This is especially important for underwater scenarios that can be extremely diverse and difficult to capture due to the risk of injury for divers and local marine life.

The simulator was used in our recently accepted work [12] to pre-train a deep neural network and quantitatively validate our system which enhances the visual odometry pose estimate by predicting good navigation paths.

## F. Simulator Evaluation

TABLE I
EVALUATION RESULTS FOR PUBLISHING 600X800 IMAGES

| Subscriber | # of Cameras | Images/second |
|---|---|---|
| Virtual Machine | 1 | 64.12 |
| Virtual Machine | 2 | 32.12 |
| Virtual Machine | 3 | 21.43 |
| External Computer | 1 | 19.17 |
| External Computer | 2 | 9.61 |
| External Computer | 3 | 6.51 |

Table I shows the image throughput between the simulator and either an external computer and virtual machine on the same computer (one of the most common use cases). The

simulator was run on a Windows 10 machine with an NVidia Titan X GPU with both the virtual machine and external system using Ubuntu 16.04. Images were published using ROS2 and then interpreted by the ros1_bridge on either the virtual machine or external computer before being processed in ROS1.

### G. Conclusion and Discussion

In this paper we have outlined our work on the development of a high-performance simulator for marine environments that provides photo-realistic simulated image capture, marine dynamics modeling and a ROS1 and ROS2 interface. We are using this for testing and evaluation of sensor-based robotics algorithms and machine learning techniques and plan to make the simulator publicly available prior to presenting this work (if accepted).

A subtopic of rapidly growing importance is transfer learning, and in our specific case, the ability to transfer learning results from the simulator to real-world systems. Doing this successfully entails several challenges outside the scope of this paper, but faithful and realistic simulation is indisputable a key requirement. In future work we seek to examine mechanisms for what we call *inverse transfer learning* whereby discrepancies between the simulator and the real environment can be identified and incorporated into the simulator to enhance its performance.

## REFERENCES


[1] S. B. Williams, P. Newman, J. Rosenblatt, G. Dissanayake, and H. Durrant-Whyte, "Autonomous underwater navigation and control," *Robotica*, vol. 19, no. 5, pp. 481–496, 2001.

[2] I. Zamora, N. G. Lopez, V. M. Vilches, and A. H. Cordero, "Extending the openai gym for robotics: a toolkit for reinforcement learning using ros and gazebo," *arXiv preprint arXiv:1608.05742*, 2016.

[3] N. Koenig and A. Howard, "Gazebo-3d multiple robot simulator with dynamics," 2006.

[4] O. S. R. F. (OSRF), "Ros2," https://github.com/ros2, 2018.

[5] C. Georgiades, A. German, A. Hogue, H. Liu, C. Prahacs, A. Ripsman, R. Sim, L. A. Torres, P. Zhang, M. Buehler, G. Dudek, M. Jenkin, and E. Milios, "Aqua: an aquatic walking robot," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 4, Sept 2004, pp. 3525–3531 vol.4.

[6] P. Giguere, G. Dudek, C. Prahacs, N. Plamondon, and K. Turgeon, "Unsupervised learning of terrain appearance for automated coral reef exploration," in *Canadian Conference on Computer and Robot Vision*, Kelowna, British Columbia, May 2009.

[7] S. Carpin, J. Wang, M. Lewis, A. Birk, and A. Jacoff, "High fidelity tools for rescue robotics: results and perspectives," in *Robot Soccer World Cup*. Springer, 2005, pp. 301–311.

[8] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[9] G. Robotics, "Dynamic underwater laser scanner," https://www.2grobotics.com/products/underwater-laser-scanner-uls-500/, 2018.

[10] S. WeiBssmann and U. Pinkall, "Underwater rigid body dynamics," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 104:1–104:7, July 2012. [Online]. Available: http://doi.acm.org/10.1145/2185520.2185600

[11] N. Plamondon and M. Nahon, "Adaptive controller for a biomimetic underwater vehicle," *Journal of Unmanned Vehicle Systems*, vol. 01, no. 01, pp. 1–13, 2013. [Online]. Available: https://doi.org/10.1139/juvs-2013-0008

[12] T. Manderson, J. Gamboa Higuera, R. Cheng, and G. Dudek, "Navigation in the service of enhanced pose estimation," 2018, accepted.