Design and Implementation of an Autonomous Surface Vehicle Platform for Water Quality Monitoring

Miyah D. Queliste, Joseph Anthony C. Hermocilla and Jaderick P. Pabico
Institute of Computer Science
University of the Philippines Los Baños
(mdqueliste,jchermocilla,jppabico)@up.edu.ph

ABSTRACT

Autonomous surface vehicles (ASV) are water surface robots that move independently without manual operation. They are becoming popular for various applications including water quality monitoring (WQM). The Philippines, though heavily dependent on its water resources, has limited adaptation of ASV technologies for automated WQM. Manual WQM, currently practiced by experts via manned vessel operations or fixed sensors, can be labor-intensive and time-consuming. It also does not provide real-time information to stakeholders. This work presents the design and implementation of an ASV platform for WQM. The platform includes a 0.8 m catamaran-style boat and a software system consisting of a web application and mission scripts. The boat was built with an on-board computer equipped with navigation and control sensors. Water quality sensors like temperature and dissolved oxygen are integrated on the boat for *in situ* data collection. The web application is used to create autonomous missions. Mobile applications were also developed to facilitate the transmission of water quality data to a database. Simulation and field tests showed successful autonomous navigation through predefined mission waypoints and sensing of sound water quality data. Loiter mode was found effective in enabling the ASV to hold position for up to one meter radius.

CCS CONCEPTS

• **Applied computing → Computers in other domains**;

KEYWORDS

autonomous surface vehicles, water quality monitoring, navigation, sensors

ACM Reference Format:
Miyah D. Queliste, Joseph Anthony C. Hermocilla and Jaderick P. Pabico. 2020. Design and Implementation of an Autonomous Surface Vehicle Platform for Water Quality Monitoring. In *Proceedings of Philippine Computing Science Congress (PCSC2020)*. Baguio City, Philippines, 12 pages.

1 INTRODUCTION

Advancements in remote sensing technologies, wireless networks, navigation, and localization systems paved the way towards rapid developments in the area of robotics. Global positioning devices and other primary components like sensors have become more compact and inexpensive. Wireless data systems have also supported longer range and higher bandwidth of communications [14]. Coupling these with autonomous vehicles have offered innovative and vast capabilities that are currently explored worldwide [5, 8, 28].

Autonomous Surface Vehicles (ASV), also called Unmanned Surface Vessels (USV), are water surface robots that move independently without continuous human intervention. These vehicles, without an operator on board, allow other modes of control such as semiautonomous control, tele-operated control, or remote control [9]. They come in various shapes and sizes and are becoming popular for a wide range of research and military applications such as security, defense, bathymetric mapping, environmental sampling, flood response, water monitoring, and agriculture [6, 15].

Apart from bathymetric surveys where ASVs were initially applied in the 1990s, water quality monitoring is also prevalent among available studies in the field. Sensing surface water quality is of increasing interest to researchers as it encompasses several environmental and economic concerns. Data collected from these activities give insights to the suitability of water for domestic use, biodiversity, and the water's ability in supporting aquatic life [11]. Water quality monitoring, for instance, helps in understanding and modeling water contaminants like chemical spills and growth of harmful algal blooms (HABs) [8].

The Philippines, one of the largest fish producers in the world, is heavily dependent on its water resources for economic sustainability. More than 800 thousand Filipinos are involved in the fishing industry in various production levels—municipal, commercial, and aquaculture [23]. The country's lakes are of great interest to water quality research because of the fish kill occurrences in the area. In 2011, more than PHP 57M worth of 700 metric tons of farmed fish were killed in Lake Taal. One identified cause was the sudden rise in temperature followed by rain which resulted in inadequate oxygen in the water surface that suffocated the fish [22, 27]. Other researchers found that it was due to "lake overturn" that pushed up the bottom stratum of the lake and pushed down the upper stratum caused by the changing of the general wind direction. Being a caldera of past volcanic eruptions, the bottom stratum is warm with sulfuric emissions from the lake floor's volcanic vents bringing along with it water layers with practically zero dissolved oxygen. Had there been real-time WQM, the Taal fisher folks would have been given early warning to avoid heavy financial losses [12, 16–18]. Thus, water quality monitoring is of scientific, economic, and governance importance for the country.

PCSC2020, March 2020, Baguio City, Philippines

Despite the urgency of innovating WQM processes, there is still limited utilization and adoption of ASV technologies in the Philippines both in the government and in the private sectors. WQM in the country is either performed (1) by experts who manually submerge sensors while on a manned vessel or (2) using fixed sensors attached to buoys whose readings are read periodically by experts or through data loggers. These manual operations are labor-intensive, time-consuming and experts-dependent. Fixed sensors also suffer from limited sensing coverage, can be hardly transferable, and require on-site installation and maintenance [25]. The data collected through these methods are commonly made available through reports after monthly or quarterly monitoring [4]. Thus, they are not generally useful for the daily management of the industries, so much so as an early warning system for the residents of the communities that depend on these water bodies for livelihood.

This study aims to leverage the current advancements in ASV research to ease and automate water quality monitoring in the Philippines. It also aims to make the water quality data readily available to stakeholders such as officials, fish farmers, and related industries. This study describes the development of an ASV platform that enables the automated collection and transmission of water quality information.

2 RELATED WORK

Since the 1990s, research has progressed in the development of ASVs. Among the pioneers is the ARTEMIS platform by MIT Sea Grant which was first used for collecting bathymetric data in a river in Boston, Massachusetts [28]. Further improvements of the platform resulted to the creation of Autonomous Coastal Exploration System (ACES) which was used for hydrographic survey in Boston Harbor [13]. ACES was further upgraded as AutoCat which reduced the mechanical and electronic shortcomings of the previous ones [15].

Various vehicle designs were explored through the years and catamaran styles showed to be popular on ASVs for research purposes. This is due to their lower speed and higher stability as compared to monohull ASVs which are relatively faster [5]. Examples of catamaran ASVs are SESAMO and ROAZ which were used for oceanographic research [2] and bathymetric surveys [6]. A more recent 16-foot long catamaran ASV is capable of navigating through complex areas and measuring water quality parameters and greenhouse gas emissions. Its novel feature is its integration with a distributed floating sensor network which offered more advanced communications [5].

As attempts to improve environmental data gathering and cover larger areas of interest, research works on multi-robot/cooperative watercrafts have emerged. Such is the Multilevel Autonomy Robot Telesupervision Architecture (MARTA), an architecture for supervising and controlling a fleet of ASVs with environmental sensors. It was originally designed for supervisory control of multiple robots for lunar surface exploration but was later expanded and used in Telesupervised Adaptive Ocean Sensor Fleet (TAOSF) for control of multiple robots for marine and freshwater research. It was applied on the OASIS platform for oceanic boats and on the Robotic Sensor Boats (RSBs) for freshwater applications [11].

The need for longer duration of deployment of ASVs has driven researchers towards renewable energy sources such as wind, solar and wave energy for ASV propulsion. Solar-powered boats are more common [5, 14]. However, solar panels cannot always be relied on and the effect of its additional weight can be a downside in some situations. Wave power, on the other hand, is inevitable which is why some works focused on taking advantage of it. WaveGlider, a wave-powered ASV in Hawaii, had its first prototypes in 2005 for ocean observation. WaveGliders were good listening platforms and were originally developed to monitor whale sound but have boundless application opportunities [7].

In research works on water quality monitoring, temperature, pH, conductivity, salinity, and dissolved oxygen (DO) are some of the frequently identified primary parameters of interest [11, 19, 26, 29]. A study in Taal lake, for example, assessed the effect of aquaculture activities on selected water quality parameters [21]. The study compared water quality parameters between aquaculture and non-aquaculture sites in the lake. The parameters include temperature, transparency, pH, nitrates, phosphates, total dissolved solids (TDS), salinity, and DO. The study showed no significant differences in the water quality parameters between the aquaculture and non-aquaculture sites in the lake. However, from the standard 5mg/L for a Class C water [3], DO dipped to critical level of below 4mg/L in the months of January and February. During this time, phytoplankton died-off followed by a fish kill event.

Nowadays, these water parameters can already be collected through off-the-shelf sensors that provide readings every second. Therefore, research efforts began incorporating these sensors into ASVs for water quality monitoring purposes. Most of these sensors require ample time to get stabilized readings. Meanwhile, ASVs are commonly confronted with disturbances during operation. Wind and current move the vehicle away from desired GPS location especially when deploying on non-still waters such as lakes, rivers, and open ocean. Thus, the need for an ASV to hold a desired position is studied for WQM applications. Station-keeping is a significant challenge in ASV development and is usually done by using controllers. Underactuated ASVs previously developed have explored Proportional-Integral-Derivative (PID), wind feed-forward, line-of-sight guidance controllers and their variants for station-keeping [1, 20, 24].

In the Philippines, WQM is done manually. In Taal, for instance, the National Fisheries Research and Development Institute (NFRDI) officials conduct WQM through boat operations by experts using a multi-parameter water quality analyzer. Limited references were published about the utilization of ASV technology to ease and automate the monitoring process in the country. There are a few works which were tested in Philippine lakes such as Taal and Laguna de Bay [25, 27]. In these research studies, they described the development of a low cost multi-robot autonomous marine surface platform. This Cooperative Robotic Watercraft (CRW) platform was tested through different real world environments including irrigation ponds, fish farms, and lakes. In their field tests, they measured the surface temperature and electrical conductivity in Taal Lake. They also deployed their CRW in a fish farming pond in Dagupan where they gathered electrical conductivity readings. Electrical conductivity approximates the TDS in the water [25].

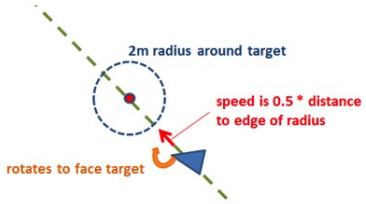


Figure 1: ArduRover loiter mode mechanism[2].

Autonomous vehicles are dependent on autopilot systems for navigation, control, and guidance. Hardware, firmware, and software components make up an autopilot system. ArduPilot[1] is a powerful and open source autopilot system capable of controlling any vehicle system including airplanes, rovers, and helicopters, boats and submarines. The firmware depends on the desired vehicle type. Currently, ArduPilot supports four vehicle types: Copter, Plane, Rover, and Antenna Tracker. ArduRover is a popular choice for building ASVs. Though originally developed for land robots, it is effectively applied to boats. ArduRover runs on Pixhawk and a number of other supported autopilot boards like Erle-Brain, a Linux-based artificial robotic brain for building robots and drones produced by Erle Robotics.

Starting from ArduRover 3.4, Loiter mode was introduced to allow boats to hold position. For water quality monitoring purposes of ASVs, this mode is useful since water quality sensors require ample time to get stabilized readings. Having the ability to hold at a specified spot will allow a more accurate collection of water quality data. Figure 1 shows an illustration of the Loiter mode. Loiter mode works by calculating a station upon switching to the mode. A waypoint radius parameter represented as WP_RADIUS is set. While the boat is within the waypoint radius of the station, the boat will drift. When the boat strays away from the target, the boat will correct its heading and drive towards the nearest circle edge.

3 METHODOLOGY

3.1 ASV Platform Overview

The ASV platform developed is composed of two major components: an ASV and a software system consisting of a web application and mission scripts. The platform architecture is illustrated in Figure 2. The ASV is a fabricated boat that navigates through a water body and collects water quality information. The web application frontend is the primary interface for creating a mission while the RESTful API backend serves HTTP requests. The API receives logs from the ASV via SMS, storing data on a MySQL database. The mission scripts are programs that manage and execute the entire autonomous mission workflow. Each component is discussed in detail in the succeeding sections.

[1]http://ardupilot.org/
[2]https://ardupilot.org/rover/docs/loiter-mode.html

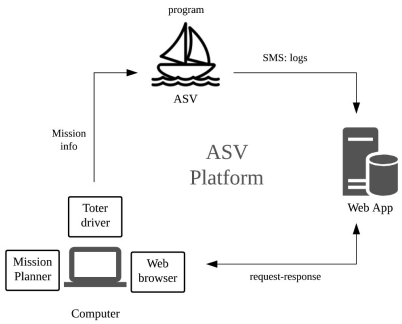


Figure 2: ASV platform architecture.

3.2 Design and Fabrication of Surface Vehicle Prototype

The ASV is a custom-built catamaran boat. The catamaran style provides stability to the boat as it traverses water surfaces. The boat is about 0.8 meters long and 0.5 meters wide. This size is appropriate for inland water deployments as it is easy to control alongside fish pens and against low to medium wave intensity in lakes. Though small, there is enough room to house all the electronic components of the system.

3.2.1 Hull Design. The hulls were made up of foam board, a strong and lightweight material. The foam board used consists of three layers, an inner layer of polystyrene foam that is covered with clay coated paper on the outside. The boards were joined using glue stick and shaped into two hulls and a center platform. The edges were covered with fiberglass tape and the entire boat was coated with waterproofing, sanded, and painted.

Two underwater thrusters were used for propulsion and skid-steering. The main consideration for this design is the requirement of the research that the ASV stays or holds on a specific position while it is collecting water quality information. The thrusters were mounted astern, parallel to each other at the sides of the boat. The electronics compartments were installed at the center of the boat.

3.2.2 Computing and Electronics. The onboard electronics are divided into two compartments: the Erle-Brain 2 and the Raspberry Pi 3 assemblies. Erle-Brain 2 is responsible for the navigation and control mechanisms while Raspberry Pi 3 is in charge of the sensing and transmission component of the ASV.

Navigation and control compartment. Erle-Brain 2 sensors such as WiFi dongle, gyroscope, compass, and GPS are used for ASV navigation and control. The underwater thrusters are also connected to Erle-Brain via an electronic speed controller (ESC). A standard servo motor is also connected to Erle-Brain. This servo turns the water quality sensors up and down during missions. The power source for this component is a 3s 5200mah lithium polymer battery, distributing power through a power regulator. The regulator provides power to Erle-Brain and the thruster ESCs. The vehicle receives input from a transmitter off-shore through a receiver connected to Erle-Brain. The assembly is housed in a box at the center platform of the boat. The layout of the box is shown in Figure 3.

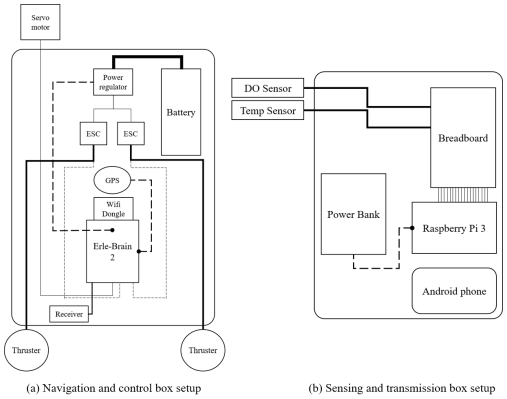


Figure 3: The layout of the ASV compartments.

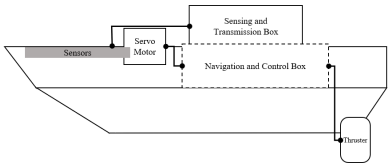


Figure 4: The hull, navigation and control box, and sensing and transmission box assembled into an ASV.

Sensing and transmission compartment. At the core of the sensing assembly is the Raspberry Pi 3 running Raspbian, the official supported operating system for RPi. A 10050mah power bank is used to power the Raspberry Pi. All payload sensors are connected to the RPi I/O through a T-Cobbler GPIO breakout. The sensors are mounted on a PVC pipe which is turned 90 degrees down by Erle-Brain's servo motor whenever a waypoint is reached to submerge the sensors approximately 15cm below the water surface.

Since one of the considerations of this research is the lack of Internet connectivity during missions, an old Android phone for the transmission of sensor readings/logs was added to the assembly. It serves as a substitute for a GSM modem. This was done due to limited resources at the time that the research was conducted. All these electronic devices are enclosed in a sealed box. This box, the hull, and the navigation and control box are assembled into an ASV as shown in Figure 4.

Weight test. The design of the surface vehicle prototype was evaluated through a weight test. The test is to check the maximum weight capacity that the prototype can handle without sinking. The test is done by incrementally adding weight to the boat while on water until the maximum allowable draft is reached—a draft of 14cm and a freeboard of 7cm. Dumbbells and ankle weights weighing were used for the test.

3.3 ASV Software System Development

A software system for managing autonomous navigation and for collecting, storing, and transmitting water quality data was developed.

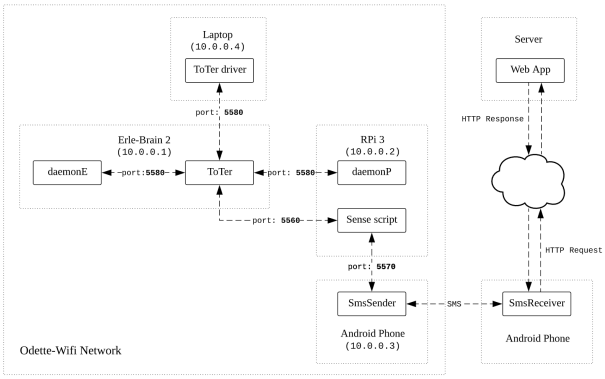


Figure 5: Conceptual design of the ASV software system.

The system is composed of Python scripts, mobile applications, and a web application. These system components communicate via socket streams, SMS, and HTTP. The conceptual design of the system is shown in Figure 5. Six machines are involved in the entire system workflow: the user's computer, Erle-Brain 2, Raspberry Pi 3, two Android phones, and a server machine. ToTer driver script runs on the user's laptop. daemonE and ToTer scripts run on Erle-Brain. daemonP and sense scripts run on Raspberry Pi. SmsSender mobile app runs in the Android phone inside the ASV. SmsReceiver app runs in the user's Android phone. The web app is hosted in a server machine.

3.3.1 Managing Autonomous Navigation. Autonomous missions are already supported in ArduRover through its Auto mode. However, to achieve the goal of this research to collect stabilized water quality readings, the ASV should be capable of autonomously holding its position while collecting water quality information. This was done by developing a set of scripts for a customized mission. These scripts communicate over the Odette-Wifi network. Erle-Brain serves as the wifi access point for the Odette-Wifi network with 10.0.0.1 as its IP address. It was configured to accept three connections: RPi, Android phone, and ground station laptop. RPi is assigned with 10.0.0.2, the Android phone, 10.0.0.3, and the laptop, 10.0.0.4. Dronekit-Python was installed on the user's laptop and in Erle-Brain. This library enables program communication to Ardupilot via MAVLink messages. It was used to write the scripts for managing autonomous navigation.

Mission Planner version 1.3.6 was installed on the user's laptop as the ground control station. Figure 6 shows the software interface of the GCS. Through Mission Planner, the user can load waypoints from the vehicle and view the ongoing mission as navigated by the ASV. Before deployment, calibrations were done using the GCS such as Radio Control Calibration, Accelerometer Calibration, Compass Calibration, and Flight Mode Configuration.

MAVProxy was installed on Raspberry Pi to forward received MAVlink packets to 10.0.0.4:14550, 10.0.0.1:14551, and 10.0.0.1:14551. Port 14550 is dedicated for Mission Planner while port 14551 is for the autonomous navigation scripts.

Figure 6: Flight plan view of Mission Planner.

Figure 7: Laguna de Bay study site.

3.3.2 Collecting, Storing, and Transmitting Water Quality Data. A sense script was developed to run in Raspberry Pi for the collection of water quality data. Two mobile applications were created for data transmission while a web application was developed for data storage. The temperature and dissolved oxygen sensors were connected to Raspberry Pi through serial and I2C communications. Both were tested using the Python codes from the sensors' documentation. The codes were ported into the sense script, the main script of RPi. The sensors were calibrated before deployment. To send collected data as SMS, an Android application was developed and installed in the mobile phone of the RPi compartment. This Android application was written in Java for Android using Android Studio version 3.1.3.

The web application runs a frontend and an API backend. The API was developed using Express - NodeJS web application framework. It is connected to a MySQL database. The web app frontend was developed using ReactJS.
System Requirements:
- Node.js version 10.4.1
- NPM version 6.1.0
- MySQL version 5.6.27
- ReactJS version 16.4.1

To receive SMS data from the ASV, from SmsSender specifically, the web app API needs an SMS gateway. Most SMS gateways require subscription fees for full functionality use. For this reason, another mobile application was developed to serve as a substitute to an SMS gateway for the API. This mobile app, called SmsReceiver, works as a middle man between the ASV and the web app API for data transmission.

3.4 Evaluation of the ASV Platform

The developed ASV platform was tested through simulation tests and field deployments in two pool areas and in Laguna de Bay.

3.4.1 Simulation Environment. A simulation environment was set up to simulate the platform workflow. The simulator is called SITL (Software in the Loop). Though originally for Linux, SITL can also be built and run natively on Windows.
System Requirements:
- Cygwin
- GCC
- MAVProxy

The complete platform workflow was tested through the SITL simulator. For the simulation test, the scripts that are supposed to run on Erle Brain (daemon and ToTer) were executed on the laptop where the simulator is running.

3.4.2 Research Area. The pilot deployment site for the ASV is at Laguna de Bay (14.3935° N, 121.1939° E). It is the largest lake in the country with a surface area of 900 km^2. Around 100 tributary rivers and streams drain into the lake. The lake is source of food, irrigation water, industrial and domestic water supply, area for recreation for about 15 million people residing around it [10]. The south edge of the lake is accessible from Brgy. Baybayin, Los Baños.

3.4.3 Field Tests. Field tests were done in two pool sites and in Laguna de Bay to evaluate the performance of the ASV platform. The first was done in City of Springs Resort in Los Baños. The second test was done in UPLB Baker Hall pool. For both pool sites, two waypoints were selected. For the lake, three missions were tested: a one-waypoint, a two-waypoint, and a rectangular mission. A WP_RADIUS of 2 meters was set. The lake test was done in November early in the morning.

The complete ASV platform workflow was evaluated in smaller parts. Manual control of the ASV was evaluated through manual tests. For missions, waypoint, loiter, and transmission performance were tested.

Manual test. Manual tests were done prior to every mission. The ASV was placed in the water and controlled via the remote controller through different throttle speeds and turns. The thrusters were observed if they followed the remote control commands properly.

Waypoint test. To test if the ASV reaches the waypoints, for each waypoint, the GPS location when the ASV switched to Loiter was obtained. This is the actual location of the ASV when ArduRover believed that it has reached the waypoint which prompted the switch to Loiter mode. This was obtained from the DataFlash log of the mission accessible through Mission Planner. This was then compared with each target waypoint in the mission file. The target and the actual points were plotted on a map and the distances were computed.

Loiter test. Loiter was tested in two ways. First, for varying durations of 30 seconds to three minutes, the ASV was switched from Manual to Loiter mode using the remote controller. Afterwards, the DataFlash log was analyzed in Mission Planner. Furthermore, after a mission, each target waypoint in the mission file was compared with the GPS location in the logs sent to the database. The target and the actual points were plotted on a map and the distances were

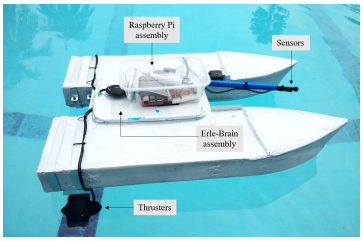
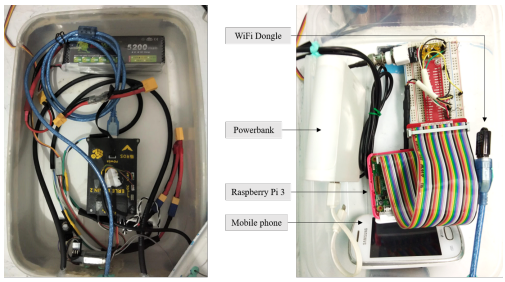


Figure 8: The ASV prototype.



(a) Navigation and control box (b) Sensing and Transmission box

Figure 9: Compartments of the ASV prototype.

computed. The Mean Squared Error (MSE) for each waypoint is also computed as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2, \quad (1)$$

where $(Y_i - \hat{Y}_i)$ is the distance of the target point from the ith actual point obtained.

Transmission test. Prior to deployment, the system time in Raspberry Pi and Erle-Brain were synchronized with the laptop. The laptop system time and the time of the user's phone were also synchronized. The transmission of data was evaluated by comparing the time in the log sent to the database with the time when the SMS was received in the user's phone.

4 RESULTS AND DISCUSSION

4.1 ASV Prototype

The fabricated catamaran ASV is shown in Figure 8 following the design and dimensions discussed in the previous section. Two Blue Robotics T100[3] thrusters were used. These are powerful underwater thrusters with forward and reverse motion. The actual view of the navigation and control box setup and the the sensing and transmission box setup are shown in Figures 9. The specifications of the water quality sensors used in the ASV are listed in Table 1. A DS18B20 temperature sensor and an Atlas Scientific Dissolved Oxygen Kit were used. The mobile phone used is an old Samsung Galaxy Star phone. It runs on Android 4.1 Jelly Bean.

The boat hull alone weighs about 2.7 kg. Together with all the hardware components onboard, the whole ASV weighs about 4.7 kg. The weight test showed that, from its current load, the ASV can handle additional 11.1 kg of load. With this additional load, the water line is right in the middle of the target freeboard and draft of the boat as shown in Figure 10. This shows that additional sensors can still be added to the prototype in the future.

[3]https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t100-thruster/

Table 1: Specifications of the water quality sensors attached to the ASV

Specification	Temperature Sensor	DO Sensor
Model	Waterproof Temperature Sensor DS18B20	Atlas Scientific Dissolved Oxygen Kit
Range	-55 °C to +125 °C	0.01 - 100+ mg/L
Accuracy	±0.5 °C	±0.05 mg/L
Reading rate	<750 msec	1 sec
Operating Voltage	3.3V - 5V	3.3V - 5V
Data Output	String in °C and °F	String in mg/L

Figure 10: The ASV carrying an additional 11.1 kg load on its deck.

4.2 ASV Software System

The sequence diagram showing the complete process workflow of the developed ASV software system is shown in Figure 11.

4.2.1 Managing Autonomous Navigation. Erle-Brain runs the daemon script on startup. This daemon script spawns the central script for navigation, ToTer, which is mainly in charge of the mission. The ToTer script is a combined Auto and Loiter mission. ToTer makes the ASV navigate the predefined waypoints in Auto mode and switches to Loiter mode whenever a waypoint is reached. This causes the ASV to hold its position to allow sensing to take place.

The ToTer script waits for client connections from the driver script running on the user's laptop. This driver script is the entry point of the platform. It is executed through a Windows batch file on the user's laptop. The driver script connects to ArduRover to get the vehicle's home location, forwards the home location to the web app API backend through a POST request, and waits for a new mission to be created.

The web app frontend is the primary interface for creating a new mission. The user interface of the application is shown in Figure 12. It includes a Place search box where the user can search or enter the area where the mission will be deployed, a drop-down menu for all the areas in the previous missions, and a Google Map. The frontend is connected to the API backend.

4.2.2 Collecting, Storing, and Transmitting Water Quality Data. Raspberry Pi also runs a daemon script on startup. This daemon script spawns the central script for sensing through the water quality sensors. ToTer connects to this sense script. For every waypoint reached and Loiter invocation, ToTer sends a "SENSE" signal to RPi.

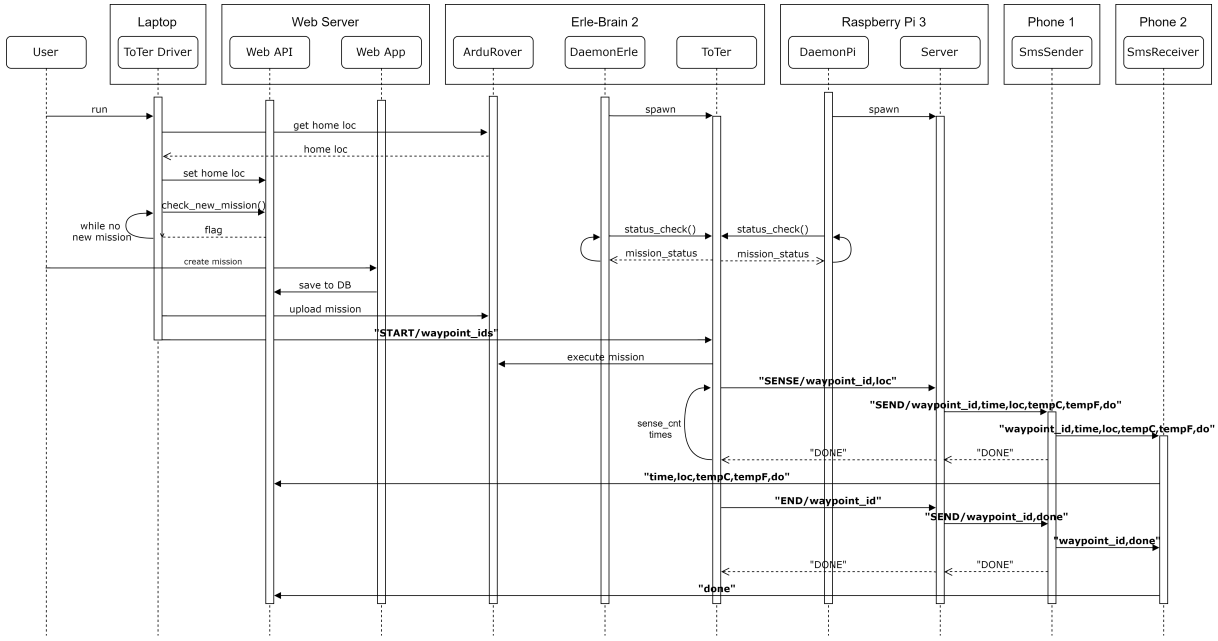


Figure 11: Sequence diagram of the entire platform workflow.

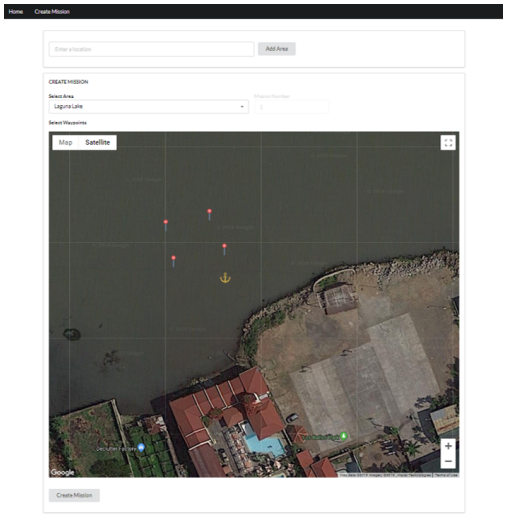


Figure 12: User interface of the web application frontend.

When the signal is received, the sense script starts reading water quality values. RPi's sense script in turn connects to the server socket of the mobile app, SmsSender. The app waits for client messages and forwards them as SMS to the web app database via the phone number specified in the app.

The database schema of the API is shown in Figure 13. The main entity of the schema is the mission which refers to a single ASV sail. A mission has an area bearing the name of the water body

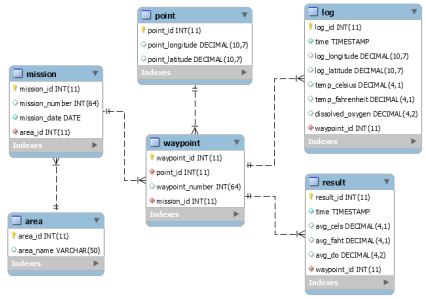


Figure 13: Database schema for the storage of collected water quality data.

to monitor, a mission date, and a mission number. A mission has a set of waypoints. A waypoint refers to a certain point (latitude and longitude coordinates) in the map that the ASV navigates to. A waypoint has a waypoint number which specifies its order in the mission execution. Logs are the significant data collected by the ASV platform tied with a specific waypoint. A log has a timestamp, a point location, and the water sensor values collected. Processed logs are stored as results in the result table which has similar attributes as log.

Three components connect to the API: the ToTer driver script discussed above, the web application frontend, and the SmsReceiver app. Connections are only GET and POST requests to retrieve and insert values in the database respectively. Table 2 shows the functionalities of the API.

Table 2: Features and routes of the web app API.

App Features	Method	Route	Function Name
Retrieve all areas	GET	/areas	findAllAreas()
Retrieve all missions of an area	GET	/areas/:area_id/missions	findAllMissions()
Retrieve all points in an area	GET	/areas/:area_id/points	findPointsByArea()
Retrieve last mission in an area	GET	/areas/:area_id/last_mission	findLastMission()
Retrieve waypoints in a mission	GET	/missions/:mission_id/waypoints	findWaypointsByMission()
Retrieve all results from a mission	GET	/missions/:mission_id/results	findMission()
Retrieve new mission flag	GET	/newmission	checkNewMission()
Insert logs of a point or process results	POST	/waypoints/:waypoint_id/logs	save()
Insert area	POST	/areas	addArea()
Insert mission	POST	/areas/:area_id/missions	addMission()
Insert point	POST	/points	addPoint()
Insert waypoint	POST	/missions/:mission_id/waypoints	addWaypoint()

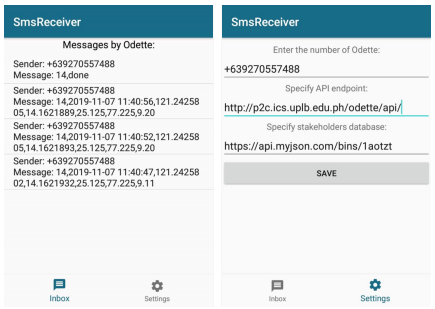


Figure 14: User interface of the SmsReceiver android application.

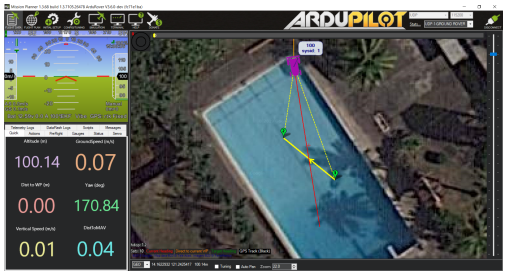


Figure 15: Mission Planner view of the mission performed by the ASV in UPLB Baker Hall pool.

SmsReceiver was designed to run on the platform user's mobile phone with Android 4.4 Kitkat OS and above. The user interface of the app is shown in Figure 14. The app contains two tabs, the Inbox tab where the received logs are displayed and the Settings tab where the user can set the number on the ASV, the API endpoint and the stakeholders' numbers. The SmsReceiver app requires Internet connectivity to work. It waits for SMS logs from the ASV. When an SMS is received, a post request is made through the route `/waypoints/:waypoint_id/logs` to insert values into the database. Before moving to the next waypoint, all logs of the current waypoint are processed into a single averaged result which is sent to the stakeholders. This will hopefully provide the stakeholders with real-time insight of the state of the water body and enable them to respond accordingly.

Table 3 shows the complete list of the protocols used in the above discussions—from the driver script to the web app API. This summarizes the format that the sender party sends and the receiver end expects.

4.3 Evaluation of the ASV Platform

4.3.1 Simulation Test. The mission created through the web app was successfully navigated by the simulated vehicle. Successful network communications were observed between the components.[4] The defined communication protocols enabled the smooth flow of data through the expected path: Driver script -> ToTer script -> Sensing script -> SmsSender -> SmsReceiver -> Web app API.

4.3.2 Field Tests. Videos of the field tests are available online.[5] The pool in City of Springs is 16 meters by 8 meters while UPLB Baker Hall pool is 25 meters by 12 meters. The Mission Planner view of the test in UPLB Baker Hall pool is shown in Figure 15. During the field tests, a challenge encountered was the inaccuracy of map points vs actual latitude and longitude. The relatively small pool made it difficult to plot waypoints and caused the initial trials to be aborted. After finding the appropriate locations on the map, the ASV was able to navigate through the waypoints.

Another challenge faced during the field tests was the difficulty of connecting the laptop to Erle-Brain's WiFi network. The laptop initially establishes a connection but gets disconnected from time to time. This caused Mission Planner to lose connection and the driver script to fail. To address this, Erle-Brain's WiFi dongle was moved out from the navigation and control box to the sensing and transmission box on top. A USB extension cable was used. This workaround improved the laptop's connectivity to the vehicle during missions.

The Mission Planner view of the rectangular mission done in Laguna de Bay is shown in Figure 16. The rectangular mission is about 11.5 meters wide and 8 meters long. The center point of the mission is about 24 meters away from the home location and 45 meters away from the shore. The collected water quality

[4]https://bit.ly/36FxdYR

[5]https://bit.ly/2FDKQvv

Table 3: Summary of the protocols used for the ASV platform.

Sender	Receiver	Form	Data Format
ToTer driver (Laptop)	ToTer (Erle-Brain)	Socket stream	"START/waypoint_id_string"
ToTer (Erle-Brain)	Server (Raspberry Pi)	Socket stream	"SENSE/waypoint_id,lng,lat"
Server (Raspberry Pi)	SmsSender (Phone 1)	Socket stream	"SEND/waypoint_id,time,long,lat,tempC,tempF,do"
SmsSender (Phone 1)	SmsReceiver (Phone 2)	SMS	"waypoint_id,time,long,lat,tempC,tempF,do" OR "waypoint_id,done"
SmsReceiver (Phone 2)	Wep app API (Server)	POST request	Request body: message = "time,long,lat,tempC,tempF,do" OR message = "done"

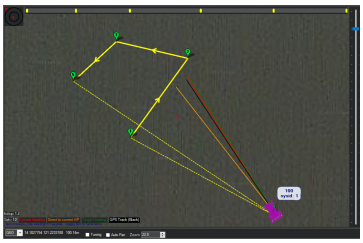


Figure 16: Rectangular Mission performed by the ASV in Laguna de Bay.

Figure 17: ASV deployed in Laguna de Bay.

data in the rectangular mission is shown in Table 4. Figure 18 shows the variance of the trials for each of the waypoints. For dissolved oxygen, values range from 4.47-5.02. On the other hand, temperature values range from 28.8 to 29.1. Minimal fluctuations in the values can be observed but standard deviations are close to zero.

Manual test. During manual tests, the ASV was able to respond to the throttle and steering commands from the remote controller. Higher throttle input translated to higher throttle output.

Waypoint test. The waypoints in the rectangular mission were successfully reached within the 2 meter waypoint radius. The summary of distances between the waypoints and loiter locations in the rectangular mission is shown in Table 5. Figure 19 shows how close the ASV was to the waypoint when it switched to Loiter. Upon reaching the 2 meter radius to each waypoint, the ASV immediately switched to Loiter. This explains the distances ranging from 1.17 to 1.9.

Loiter test. The Loiter mode was found to be effective in maintaining the position of the ASV. Figure 20 shows mode changes from Manual to Loiter and back to Manual. The vehicle drift from the target location is shown by the growing waypoint distance (green line). However, throttle output (blue line) corrects the location of the vehicle so that it is within the 2 meter waypoint radius (yellow line). Correction time ranges from three to five seconds. Faster correction was observed when the boat heading is facing towards the target location compared to when the boat heading is perpendicular to the target location. During the field tests, the tightest WP_RADIUS value tested was 1 meter.

For the rectangular mission, Figure 21 shows how close the GPS logs were from their target waypoint location. The summary of distances between the waypoints and log locations is shown in Table 6. The vehicle was able to stay within the 2 meter radius except for waypoint no. 2 where the distances of the GPS logs from the waypoint range from 2.12 to 2.66. This waypoint, consequently, has the highest MSE which is 5.96. This can be attributed to the heading of the vehicle when it reached the waypoint. The vehicle was perpendicular to the disturbance direction where it was observed to have longer correction response. Another factor to look into is the fact that the loiter points are not exactly on the target waypoints as previously shown in the waypoint test.

Transmission test. The recorded log time and the corresponding phone time when the log SMS was received is shown in Table 7. Unfortunately, time data of SMS received in the phone only includes the hour and minute. Log intervals range from four to six seconds. All of the logs were received on the same minute they were sent.

Parameter changes were done to achieve the desired movement of the ASV during ToTer mission such as speed and radius of turns. For the ArduRover parameters, The CRUISE_SPEED and CRUISE_THROTTLE were set to 0.6 and 18 respectively. This is a relatively slow motion to allow smooth observation of the ASV's movement. NAVL1_PERIOD dictates the aggressiveness of the boat during turns. It was originally set to 8 but a larger value of 16 was used instead to allow smoother turns. A lower value of 0.5 for TURN_MAX_G and higher value for TURN_RADIUS are used to lessen jerky turns when the ASV switches to Loiter mode. These parameters were found appropriate for the fabricated boat but may not be applicable to other boats with different shapes, sizes, and weights.

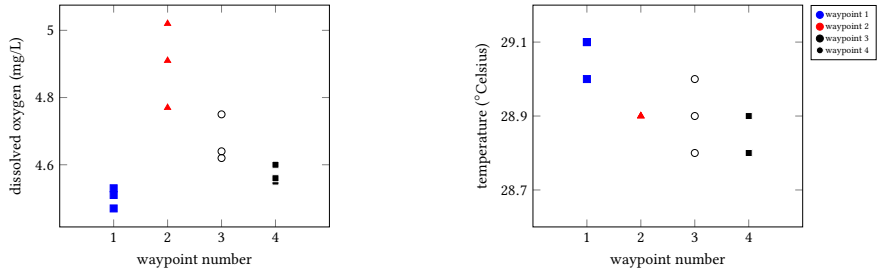
4.4 Costing

A complete list of the materials used in building the ASV platform is shown in Table 8. A huge portion of the cost is the Erle Copter[6].

[6]https://erlerobotics.gitbooks.io/erle-robotics-erle-copter/en/

Table 4: Collected water quality data in the rectangular mission.

Waypoint No.	Trial No.	DO (mg/L)	Temperature (Celsius)	Ave DO (mg/L)	DO SD (σ)	Ave Temperature (Celsius)	Temperature SD (σ)
	1	4.53	29.1				
1	2	4.47	29.0	4.50	0.0249	29.1	0.0471
	3	4.51	29.1				
	1	5.02	28.9				
2	2	4.91	28.9	4.9	0.1023	28.9	0
	3	4.77	28.9				
	1	4.75	28.9				
3	2	4.64	29.0	4.67	0.0571	28.9	0.0816
	3	4.62	28.8				
	1	4.55	28.9				
4	2	4.56	28.8	4.55	0.0216	28.8	0.0471
	3	4.60	28.8				



(a) Dissolved oxygen **(b) Temperature**

Figure 18: Sensor values per waypoint in the rectangular mission.

Table 5: Distances between the waypoints and loiter locations in the rectangular mission.

Waypoint No.	Waypoint Location (lat,lng)	Loiter Location (lat,lng)	Distance from Waypoint (m)
1	(14.1828513, 121.2231649)	(14.1828525, 121.2231816)	1.81
2	(14.1829327,121.2232302)	(14.1829226, 121.2232256)	1.22
3	(14.1829498,121.2231569)	(14.1829431, 121.2231653)	1.17
4	(14.1829079,121.2231071)	(14.182921, 121.2231137)	1.62

Table 6: Distances between waypoints and collected GPS logs during loiter in the rectangular mission.

Waypoint No.	Waypoint Location (lat,lng)	Trial No.	Log Location (lat,lng)	Distance from Waypoint (m)	Mean Squared Error (m^2)
		1	(14.1828464, 121.2231702)	0.79	
1	(14.1828513, 121.2231649)	2	(14.1828477, 121.223164)	0.41	0.27
		3	(14.1828517, 121.2231636)	0.15	
		1	(14.1829423, 121.2232132)	**2.12**	
2	(14.1829327,121.2232302)	2	(14.182936, 121.2232072)	**2.51**	5.96
		3	(14.1829333, 121.2232056)	**2.66**	
		1	(14.1829453, 121.2231473)	1.15	
3	(14.1829498,121.2231569)	2	(14.182946, 121.223147)	1.15	1.29
		3	(14.1829448, 121.2231482)	1.10	
		1	(14.1829193, 121.2230931)	1.97	
4	(14.1829079,121.2231071)	2	(14.1829171, 121.2230934)	1.80	3.17
		3	(14.1829158, 121.2230952)	1.55	

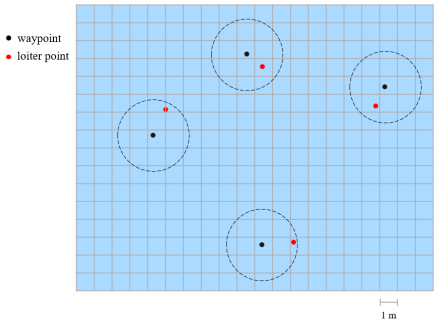


Figure 19: Plotted waypoints vs loiter locations in the rectangular mission.

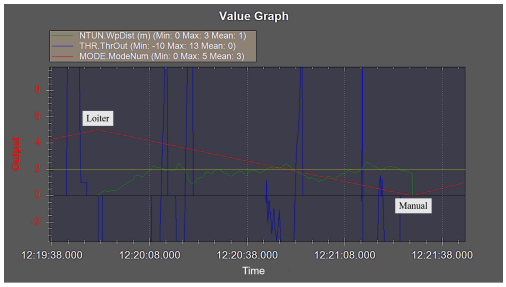


Figure 20: Mission Planner log graph during Loiter.

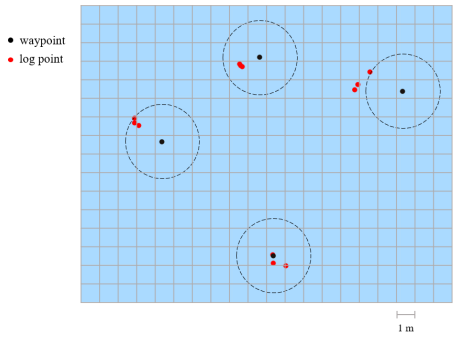


Figure 21: Plotted waypoint vs log locations of the rectangular mission.

Majority of the Erle Copter unit's parts were used in building the ASV except from the frame and four motors and propellers. This means that the actual total cost may be lower. Erle Copter can also be replaced with a cheaper brain and peripherals in the market.

5 CONCLUSION AND FUTURE WORK

An autonomous surface vehicle platform for water quality monitoring was developed in this study. A catamaran-style ASV was fabricated with Erle-Brain 2 as the brain for the navigation and control component and Raspberry Pi 3 for the sensing component. The web application developed enabled the creation of missions and its web API backend allowed the storage of data and serving of HTTP requests. Mobile applications successfully facilitated the sending and receiving of data.

The ASV platform was evaluated through simulation tests and field tests in City of Springs Resort pool, UPLB Baker Hall pool, and Laguna de Bay. Autonomous missions were successfully achieved. The ASV was able to reach the waypoints. The Loiter mode was found effective in maintaining the position of the ASV for up to 1 meter radius. The ASV was able to collect sound water quality data and transmit and store them using the developed protocols.

Possible future work may include the incorporation of additional water quality sensors such as pH and conductivity. Obstacle avoidance capability can also be added to allow deployment of the ASV alongside fish pens. A larger ASV hull can also be made with sturdier material. Using ballast to attain deeper drafts could also help in reducing the effect of disturbances but applicability still depends on the payload electronics.

Table 7: Log time vs phone time when they were received.

Waypoint No.	Trial No.	Log Time (hh:mm:ss)	Time Received (hh:mm)
	1	07:01:14	07:01
1	2	07:01:20	07:01
	3	07:01:26	07:01
	1	07:02:12	07:02
2	2	07:02:16	07:02
	3	07:02:21	07:02
	1	07:02:54	07:03
3	2	07:02:59	07:03
	3	07:03:03	07:03
	1	07:03:38	07:03
4	2	07:03:43	07:03
	3	07:03:48	07:03

Table 8: Cost of ASV platform components.

Qty	Description	Unit Price	Total Price	Grand Total
1	Erle Copter	36,738.97		
1	USB Extension Cable	150.00		
1	Turnigy Remote Controller	3,409.48		
2	T100 Thruster, Basic ESC	7,479.36	14,958.72	
	Shipping fee	1,807.51		
	Customs duties and taxes	2700.00		
1	Raspberry Pi 3, case, power supply	2801.00		
1	Breadboard, Resistor, Jumper wires	121.00		
1	Cobbler breakout, Cable	299.75		
1	Waterproof temperature sensor	149.75		
	Shipping Makerlab	150.00		
1	Servo Motor	449.75		
1	Atlas Scientific Dissolved Oxygen Kit	17,500.00		
	Boat: Foam Board	1113.00		
	Waterproofing	1302.00		
	Paint Tools	355.00		
	Aluminum	50.00		
	Batteries	459.00		
	Sim card	40.00		
	Plastic Containers	88.00		
				84,642.93

ACKNOWLEDGMENTS

The authors would like to thank the Institute of Computer Science and its Systems Research Group for the hardware resources used. The authors would also like to acknowledge Prof. Jaime Samaniego, Prof. Danilo Mercado, and Prof. Mark Dondi Arboleda of U.P. Los Baños for their valuable comments and helpful suggestions.

The authors would also like to thank Prof. Nelio and Mr. Bryan Altoveros for their assistance in the instrumentation process and Mr. Andric Quinn Baticos and Ms. Bianca Ruth Bautista for collaborating on a mini project associated to this research. This work is made possible by the grant from the Faculty Development Program II of the Commission on Higher Education (CHED).

REFERENCES

[1] Massimo Caccia, Marco Bibuli, Riccardo Bono, and Gabriele Bruzzone. 2008. Basic navigation, guidance and control of an unmanned surface vehicle. *Autonomous Robots* 25, 4 (2008), 349–365.

[2] Massimo Caccia, Riccardo Bono, Gabriele Bruzzone, E Spirandelli, G Veruggio, AM Stortini, and G Capodaglio. 2005. Sampling sea surfaces with SESAMO: an autonomous craft for the study of sea-air interactions. *IEEE Robotics & Automation Magazine* 12, 3 (2005), 95–105.

[3] DENR. 1990. DENR AO No. 34 Series of 1990. *Department of Environment and Natural Resources, Administrative Order* 34 (1990).

[4] DENR. 2008. Water Quality Monitoring Manual. https://water.emb.gov.ph/wp-content/uploads/2017/09/Water-Quality-Monitoring-Manual-Vol.-1-ambient_14aug08.pdf. 1 (2008). Accessed: October 2019.

[5] Grinham Alistair Dunbabin, Matthew and James Udy. 2009. An autonomous surface vehicle for water quality monitoring. In *Australasian Conference on Robotics and Automation (ACRA)*. 2–4.

[6] Hugo Ferreira, C Almeida, A Martins, J Almeida, N Dias, A Dias, and E Silva. 2009. Autonomous bathymetry for risk assessment with ROAZ robotic surface vehicle. In *Oceans 2009-Europe*. IEEE, 1–6.

[7] Roger Hine and Philip McGillivary. 2007. Wave powered autonomous surface vessels as components of ocean observing systems. *Proc. of PACON* (2007), 1–9.

[8] Gregory Hitz, François Pomerleau, Marie-Eve Garneau, Cédric Pradalier, Thomas Posch, Jakob Pernthaler, and Ronald Y Siegwart. 2012. Autonomous inland water monitoring: Design and application of a surface vessel. *IEEE Robotics & Automation Magazine* 19, 1 (2012), 62–72.

[9] Zhixiang Liu, Youmin Zhang, Xiang Yu, and Chi Yuan. 2016. Unmanned surface vehicles: An overview of developments and challenges. *Annual Reviews in Control* 41 (2016), 71–93.

[10] LLDA. [n. d.]. Laguna de Bay and its Tributaries. http://http://llda.gov.ph/ldb-and-its-tributaries/. Accessed: November 2019.

[11] Kian Hsiang Low, Gregg Podnar, Stephen Stancliff, John M Dolan, and Alberto Elfes. 2009. Robot boats as a mobile aquatic sensor network. In *Proc. IPSN-09 Workshop on Sensor Networks for Earth and Space Science Applications*.

[12] Damasa Magcale Macandog, Christian Paul P de la Cruz, Jennifer D Edrial, Marlon A Reblora, Jaderick P Pabico, Arnold R Salvacion, Teodorico L Marquez Jr, Paula Beatrice M Macandog, Diezza Khey B Perez, et al. 2014. Eliciting local ecological knowledge and community perception on fishkill in Taal Lake through Participatory Approaches. *Journal of Environmental Science and Management* 17, 2 (2014).

[13] Justin E Manley. 1997. Development of the autonomous surface craft "ACES". In *Oceans' 97. MTS/IEEE Conference Proceedings*, Vol. 2. IEEE, 827–832.

[14] Justin E Manley. 2008. Unmanned surface vehicles, 15 years of development. In *OCEANS 2008*. IEEE, 1–4.

[15] Justin E Manley, Aaron Marsh, Whitney Cornforth, and Colette Wiseman. 2000. Evolution of the autonomous surface craft AutoCat. In *OCEANS 2000 MTS/IEEE Conference and Exhibition. Conference Proceedings (Cat. No. 00CH37158)*, Vol. 1. IEEE, 403–408.

[16] Jaderick P. Pabico, Arnold R. Salvacion, and Damasa B. Magcale-Macandog. 2015. Data Manipulation Approaches in Learning a Model for Predicting Rare Events: A Case Study of the Taal Lake Fish Kill Predictive Model from Weather Data. In *8th National Research Workshop on Modeling, Simulation and Scientific Computing*. Computing Society of the Philippines, University Hotel, University of the Philippines, Diliman, Quezon City.

[17] Jaderick P. Pabico, Arnold R. Salvacion, and Damasa B. Magcale-Macandog. 2015. Mining the Sentiments in Social Media Microposts for Nowcasting Fish Kill Events in Taal Lake. In *2015 International Conference on Building Resilience and Developing Sustainability*. Baguio Convention Center and UP Baguio, Baguio City.

[18] Jaderick P. Pabico, Arnold R. Salvacion, and Damasa B. Magcale-Macandog. 2015. Social Sensor as an Emerging Remote Sensing Tool and Its Application to Sensing Taal Lake Fishkill. In *The 36th Asian Conference on Remote Sensing*. Asian Association on Remote Sensing, Crowne Plaza Manila Galleria, Quezon City.

[19] Sundarambal Palani, Shie-Yui Liong, and Pavel Tkalich. 2008. An ANN application for water quality forecasting. *Marine Pollution Bulletin* 56, 9 (2008), 1586–1597.

[20] Arvind Pereira, Jnaneshwar Das, and Gaurav S Sukhatme. 2008. An experimental study of station keeping on an underactuated ASV. In *2008 IEEE/RSJ International Conference on Intelligent RObots and Systems*. IEEE, 3164–3171.

[21] Blesshe L Querijero, Airill L Mercurio, et al. 2016. Water quality in aquaculture and non-aquaculture sites in Taal Lake, Batangas, Philippines. *Journal of Experimental Biology and Agricultural Sciences. http://dx. doi. org/10.18006/2016.4 (1)* 109 (2016).

[22] M. Rabe. 2011. Weather change led to Batangas fishkill, says BFAR. http://newsinfo.inquirer.net/10339/weather-change-led-to-batangas-fishkill-says-bfar. Accessed: August 2017.

[23] R Rivera, D Turcotte, A Boyd-Hagart, J Pangilinan, and R Santos. 2002. Aquatic resources in the Philippines and the extent of poverty in the sector. (2002).

[24] Edoardo I Sarda, Huajin Qu, Ivan R Bertaska, and Karl D von Ellenrieder. 2016. Station-keeping control of an unmanned surface vehicle exposed to current and wind disturbances. *Ocean Engineering* 127 (2016), 305–324.

[25] Paul Scerri, Prasanna Velagapudi, Balajee Kannan, Abhinav Valada, Christopher Tomaszewski, John Dolan, Adrian Scerri, Kumar Shaurya Shankar, Luis Bill, and George Kantor. 2012. Real-world testing of a multi-robot team. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*. International Foundation for Autonomous Agents and Multiagent Systems, 1213–1214.

[26] S Shrestha and F1 Kazama. 2007. Assessment of surface water quality using multivariate statistical techniques: A case study of the Fuji river basin, Japan. *Environmental Modelling & Software* 22, 4 (2007), 464–475.

[27] Abhinav Valada, Prasanna Velagapudi, Balajee Kannan, Christopher Tomaszewski, George Kantor, and Paul Scerri. 2014. Development of a low cost multi-robot autonomous marine surface platform. In *Field and service robotics*. Springer, 643–658.

[28] Thomas W Vaneck, Claudia D Rodriguez-Ortiz, Mads C Schmidt, and Justin E Manley. 1996. Automated bathymetry using an autonomous surface craft. *Navigation* 43, 4 (1996), 407–419.

[29] Serge Zhuiykov. 2012. Solid-state sensors monitoring parameters of water quality for the next generation of wireless sensor networks. *Sensors and Actuators B: Chemical* 161, 1 (2012), 1–20.