Fall 2019

A Novel and Inexpensive Solution to Build Autonomous Surface Vehicles Capable of Negotiating Highly Disturbed Environments

Jason Moulton

A Novel and Inexpensive Solution to Build Autonomous Surface Vehicles Capable of Negotiating Highly Disturbed Environments

by

Jason Moulton

Bachelor of Science
Clarkson University 1999

Master of Engineering
University of South Carolina 2008

Submitted in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy in

Computer Engineering

College of Engineering and Computing

University of South Carolina

2019

Accepted by:

Ioannis Rekleitis, Major Professor

Jason O'Kane, Committee Member

Pooyan Jamshidi, Committee Member

Shamia Hoque, Committee Member

Nikos Vitzilaios, Committee Member

Cheryl L. Addy, Vice Provost and Dean of the Graduate School

Dedication

I would like to dedicate this thesis to my family.

To my wife Kendra and daughter Madeleine and Reagan, without you, the motivation for seeking out knowledge would not matter.

To my parents, Cathy and Don, without you, I would not have been instilled with the values and perseverance required to complete this work.

Acknowledgments

Throughout my Ph.D. research, I have been fortunate to have the support and encouragement of many people. I personally thank everyone who has had a hand in making this thesis possible. First, I would like to thank my supervisor Yiannis Rekleitis. Without his broad vision and foresight, I am unsure if I could have found a research area with the knowledgable cadre and resources to complete the work leading to the thesis. Additionally, I would like to thank Alberto Quattrini Li. His supervision and unparalleled expertise in the Autonomous Field Robotics Lab (AFRL) have ensured that the lab and me personally, have maintained the resources and guidance to succeed.

A special thanks goes to the members of AFRL. The unique blend of backgrounds, expertise, styles, and broad range of research interests make the lab a fun and interesting place to work. In particular, I would like to thank Nare Karapetyan for her technical expertise and constant support relevant to machine learning algorithm implementation. Sharmin Rahman, MD Modasshir and Marios Xanthidis made AFRL a fun place to work and were always supportive with ideas and intelligent input through all the marathon group meetings.

I would like to thank the University's Department of Civil and Environmental Engineering for the support and use of their hydraulic flow lab. Special thanks to Enrica Viparelli for lab access and support throughout my trials and use of their equipment to conduct sensor calibration.

Many thanks go out to the people from the Department of Computer Science and Engineering. A special thank you goes out to Randi Baldwin, Sri Satti, and

Sarah Dyer from the department's administrative office for your support and help completing numerous graduate forms, travel planning requests, purchases, and reimbursements. You are all very professional and pleasant to work with.

In addition thank you to my personal friend Sean Reed for his assistance and structural engineering ideas as it pertained to adding splash and water guards to the boat, maintaining the engine and drive system, and creating a stable sensor mounting platform.

Most of all, I am grateful to my family for their support through this process. There was high times, low times, and some very trying times that they stood by me and endured with me. For that, I am forever grateful.

Abstract

This dissertation has four main contributions. The first contribution is the design and build of a fleet of long-range, medium-duration deployable autonomous surface vehicles (ASV). The second is the development, implementation, and testing of inexpensive sensors to accurately measure wind, current, and depth environmental variables. The third leverages the first two contributions, and is modeling the effects of environmental variables on an ASV, finally leading to the development of a dynamic controller enabling deployment in more uncertain conditions.

The motivation for designing and building a new ASV comes from the lack of availability of a flexible and modular platform capable of long-range deployment in current state of the art. We present a design of an autonomous surface vehicle (ASV) with the power to cover large areas, the payload capacity to carry sufficient batteries to power components and sensor equipment, and enough fuel to remain on task for extended periods. An analysis of the design, lessons learned during build and deployments, as well as a comprehensive build tutorial is provided in this thesis.

The contributions from developing an inexpensive environmental sensor suite are multi-faceted. The ability to monitor, collect, and build models of depth, wind, and current in environmental applications proves to be valuable and challenging, where we illustrate our capability to provide an efficient, accurate, and inexpensive data collection platform for the community's use. More selfishly, in order to enable our end-state goal of deploying our ASV in adverse environments, we realize the requirement to measure the same environmental characteristics in real-time and provide them as inputs to our effects model and dynamic controller. We present our methods

for calibrating the sensors and the experimental results of measurement maps and prediction maps from a total of 70 field trials.

Finally, we seek to inculcate our measured environmental variables along with previously available odometry information to increase the viability of the ASV to maneuver in highly dynamic wind and current environments. We present experimental results in differing conditions, augmenting the trajectory tracking performance of the original way-point navigation controller with our external forces feed-forward algorithm.

Table of Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

"Develop success from failures. Discouragement and failure are two of the surest stepping stones to success."

-Dale Carnegie

1.1 Motivation

Why do we need to design and build a new ASV, given the plethora of existing platforms? Given a marine environment, its associated challenges and expanding requirements to monitor, inspect, and record its dynamics, it is becoming increasingly difficult to meet the demands using manual methods. Furthermore, operations from an oceanographic vessel are costly and limited to open sea deployment. Coupled with the growth of human population and the desire to expand residential

Figure 1.1: Extreme events, such as floods, can result in severe infrastructure damage, bridge collapse. A capable Autonomous Surface Vehicle (ASV) can assess the soundness of the structure in a safe and timely manner.

and commercial interests to remote waterways, the ability to automate exploration and monitoring is necessary to overcome manual collection shortcomings, specifically the ability to access remote areas as well as ensuring complete coverage of large areas that usually change between human visits.

This thesis deals with the construction and deployment of autonomous surface vehicles into unknown and dynamically changing environments. By integrating wind, current and depth sensors, we enable our ASV to safely map and navigate high-flow rivers, river deltas with eddy currents, and ocean mouths with tidal currents. Ultimately, the ability for our ASV platform to remain deployed in highly disturbed environments for sustained periods of time will enable monitoring, inspection, recovery, and development activities far beyond our current capabilities.

1.2 Defining Dynamic Disturbances

Throughout this thesis, we will use the phrases dynamic environments and disturbances to describe areas where there are forces acting on an ASV are usually not visually detectable through video. In our case, wind and water current are the two main such variables which we attempt to measure, predict and model to enable the ASV's navigation. In addition, the depth can be considered as a contributing variable due to its effect on hydraulic flow intensities. The combination of these variables ultimately results in an external force on the ASV, creating a dynamic environment.

1.3 Sensing and Exploiting Dynaminc Disturbances

While there exist numerous sensors capable of measuring wind and current ranging in price from approximately 100 dollars to multiple thousands of dollars, and size from a baseball to a large suitcase, we adapt the smaller, lightweight and less expensive sensors for our use. Through the use of micro-controllers and some basic mathematical and geometric calculations, we are able to measure, record, predict and utilize the

measured environmental dynamics for mapping, route planning and developing ASV control strategies.

1.4 Contributions of this Work

This thesis provides comprehensive design, build, and experimentation results of an autonomous surface vehicle platform capable of operating in restrictive and highly disturbed environments. Specifically, this initiative is supported through five supporting research areas. The first requirement to building a group of mobile autonomous surface vehicles, integrating them with inexpensive and commercially available sensors, mapping and modeling wind, current, and depth dynamics of a volatile environment, developing an exploration strategy algorithm to collect data in such environments, and finally implement our proactive control algorithm to enable operation when dynamics change from calm to disturbed.

A non-comprehensive list of contributions from this thesis follows:

1. Constructing an inexpensive, long-term operation capable, autonomous surface vehicle[48], published in Oceans 2018.

2. Improving multi-robot remote-frequency communication[42], published in Oceans 2018.

3. Covering large-scale areas with Multiple-ASV coverage algorithms[29], published at ICRA 2018.

4. Developing and integrating inexpensive sensors on an ASV and in ROS[49], published in ISER 2018.

5. Mapping and predicting dynamic environments [49], published in ISER 2018.

6. Designing and implementing exploration strategies for modeling dynamic environments[30], submitted to IROS 2019.

7. Providing comprehensive Jetyak build tutorial wiki[5].

8. Designing and providing sensor micro-controller source code in Appendix B.

9. Modeling the effects of environmental forces on an ASV[51], submitted to IROS 2019.

10. Enabling proactive control through feed-forward augmentation of PID navigation controller[50], submitted to FSR 2019.

1.5 Thesis Outline

Chapter 2, which follows immediately, presents the related work which supports the rest of this thesis in a consolidated manner. Chapter 3 presents the design, build and deployment of our Jetyak platform from which remaining chapters build. Chapter 4 presents the comparison, selection, modification, mounting and calibration of the wind, current and depth sensors required to enable our work. Chapter 5 captures the methods used to collect raw environmental data as well as transform, visualize, and predict the data from the sensor to the boat to the world reference frames. Resulting recorded values and predictions are validated against alternate measurements. Chapter 6 completes the research of this thesis as it defines our proactive ASV control algorithm and presents our results. And finally, Chapter chap:conclusion presents our closing thoughts and areas for future research, such as decreasing proactive control response time by integrating the countermeasures for external forces at the controller level, rather than augmenting the current state-of-the-art. In addition, research on path planning given limited information such as river direction flow and riverbank maps is further enabled by this thesis. The appendices include communication architecture optimization as a necessity to support this dissertation in Appendix A. In addition, due the uniqueness of the low-cost design and implementation for en-

vironmental sensors, we include the Arduino code for wind, current, and kill-switch override drivers in Appendix B

Chapter 2

BACKGROUND

> "In science, moreover, the work of the individual is so bound up with that of his scientific predecessors and contemporaries that it appears almost as an impersonal product of his generation."

-Albert Einstein

In this chapter, we examine the relevant background for our work. It is worth mentioning that there exists a wide array of disciplines attempting to understand and model the dynamic characteristics of water current and wind forces and the impacts to their research area. A non-comprehensive reference list such as [58] compares over 175 traditional methods for determining the water current dynamics along coastal ocean regions. We present relevant work focused on developing small, lightweight surface vehicles in Section 2.1, then a brief overview of physical sensor deployment in Section 2.3. Section 2.4 discusses work on collecting and predicting environmental dynamics. Section 2.5 examines PID, feed-forward, and predictive control implementations. Finally, Section 2.6 contains contains closely related research that has been completed and closely impacts the problems and solutions presnted throughout this dissertation.

2.1 ASV Designs

The most relevant work to the proposed design is the WHOI Jetyak by Kimball *et al.* [32], on which we have based our design. Their Jetyak is capable of autonomous

operations carrying single-purpose payloads for extended periods of time; however, the design does not allow for manual operations or easy reconfiguration of the payload. Next, we discuss other approaches to the ASV design and how their contributions have influenced our design.

Among the earlier designs of a small scale ASV was MARE by Girdhar *et al.* [23]. Driven by low cost considerations, it enabled collection of visual data over shallow coral reefs and operated as a communication point in multi-robot operations [67]. The design was based on the catamaran style with two electric motors that where controlled in a differential drive configuration. Battery powered, the range of operations was limited. Similar catamaran design with two electric motors have also the Kingfisher and the Heron Unmanned Surface Vehicle (USV) models from Clearpath Robotics[1]. While portable, their range of operations is limited to one to two hours.

In 2005, Curcio *et al.* introduced their surface craft for oceanographic and undersea testing (SCOUT)[19]. SCOUT is geared for oceanographic deployment based on an obstacle avoidance system working in conjunction with a remote palm device for high-level mission control. Their pioneering design and build of a truly unmanned boat set the stage for a variety of expansions of their original design. For our purposes, the electric drivetrain results in an increased draft to allow clearance for the electric motor head and propeller to displace water below the stern of the kayak. As well, the mission specific sensor design and implementation offer us insight for our design to remain flexible and modular to accommodate larger and heavier instrumentation. Examples include sidescan sonar sensors and acoustic Doppler current profilers (ADCP) that are becoming attainable as the devices become more affordable, but still not inexpensive. The small size limiting manual operation capabilities, low operating maximum speed of 5.6 kilometers per hour, and medium operation time of eight hours between charging are shortcomings of SCOUT that we seek to

[1] `https://www.clearpathrobotics.com/`

improve.

In 2008, Mahachek *et al.* introduced their small waterplane twin hull (SWATH)[41] ASV. SWATH employed two under the surface catamarans running electric motors mainly for shallow and inland water operation. They originally deployed multibeam sonar as a tool for bathymetric mapping. Again in 2013, Rasal[61] sought to improve the path following capabilities of SWATH in environments with wind and current present. While the results are successful in moderate conditions, its speed and deployment duration do not fit our needs for long distance deployment. However, their off-board control system inspires our design and implementation for customized control sequences for future mission specific tasks, such as object placement and retrieval.

In December 2012, Rodriquez *et al.* published a comparison study[63] of existing ASVs for the specific purpose of measuring the environmental indicators that bear directly on climate change. Throughout their study, they cover capabilities of satellites, weather balloons, RADAR, stationary buoy arrays, manned boats, autonomous underwater vehicles (AUV), and ASVs. They compared each platform's capabilities of measuring wind speed, wind direction, water salinity, water temperature, barometric pressure, and oil mapping. Their comparison of generic platform capabilities logically concluded that only manned boats and ASVs were capable of monitoring all indicators. Their report goes on to compare several AUV and ASV implementations with much insight gained from interviews with scientists and engineers from NOAA, Worcester Polytechinic Institute (WPI), Social and Environmental Research Institute (SERI), and the Applied Ocean Physics and Engineering Department from WHOI. Leveraging their conclusion that both manned boats and ASVs provide the greatest capabilities led us to our requirement to keep our design flexible enough to support manned and unmanned operating modes.

In 2014, Fraga *et al.* introduced Squirtle[21], an autonomous electric catamaran for inland water environmental monitoring. While their lack of passenger carrying

Figure 2.1: Woods Hole Oceanographic Institutes arctic exploration ADCP Jetyak.

capability, deeper draft due to propeller shafts and reliance on an electric power source are not in line with our design goals, their methods for implementing a ROS node to provide autonomous control based on precise real time kinematic (RTK) GPS and inertial measurement unit (IMU) measurements, provide insight to our challenges with a maintaining a fully capable, self-reliant platform.

Based on the payload, speed, and mission duration capabilities in the reviewed literature above, we decided that the WHOI approach was the best starting point for our development. Their ingenuity and pioneering approach to expand the capabilities of a commercial platform led us to select Mokai as the base platform from which to build the ASV. From there, we seek to add modularity and flexibility to their design in order to provide a multipurpose platform.

2.2 Understanding Currents

Our motivation to combat environmental dynamics through sensing and proactive controls requires a deeper understanding of currents and their properties. Unlike

wind and depth, where point measurements are quite simple to collect, currents are more challenging given the borderline between air and water domains. Much of the current research seeks to measure currents and their effects within the entire water column[33, 16, 76].

However, we are seeking the determine the current velocity at the surface of the water that is the force directly acting on our small, lightweight, buoyant ASV. Surface current specific technical notes to support data gathering goals[45] is more closely aligned with our goals.

2.3 Sensors

The basic telemetry data provided by almost every on-board controller at a minimum consists of IMU, GPS, and velocity data. In order to enable our ASV to adapt to environmental variables, we must add methods for measuring the wind, current, and depth dynamics surrounding it. In this section, we will discuss the fundamental concept of Hall-effect sensors as well as compare the cost of several technologies in our three areas of interest. Finally, we will cover some previous work which aided greatly in the mounting and calibration of sensors presented in Chapter 4.

2.3.1 Hall-effect Sensors

The first reference text one will probably find when investigating how sensors work is Ramsden's textbook titled Hall-effect Sensors: Theory and Application[60]. While this thesis is not focused on the electrical and magnetic properties that have been used for decades, this reference is important to recognize due to the wide-array of comparably inexpensive sensors built on its foundation that can be leveraged to provide a working solution to our goal.

2.3.2 Sensor Cost Comparison

While there are several much more sophisticated sensors that provide data points far beyond the input space we are considering for this thesis, it is practical to do a cost comparison to verify. In addition, once the decision is made to purchase more advanced sensors, it is necessary to weigh the cost of time to reverse engineer any proprietary code to be able to access the data online. Table2.1 provides a sampling of the three types of sensors that we are seeking to include as input to our improved controller implementation.

Table 2.1: Comparison of commercially available depth sonar, anemometers and water current speed sensors.

Type	**Model**	**Open-source**	**Price**	**Note**
Depth	CruzPro ATU120AT	Yes	$180	NMEA 0183
Depth	Furino DST-800PSF	No	$295	NMEA 2000
Depth	Humminbird Helix 5	No	$500	Proprietary
Wind	Spark Fun SEN-08942	Yes	$125	RS-232
Wind	TSI 9535-A	No	$1080	Proprietary
Current	Ray Marine ST800	Yes	$60	Analog
Current	Rickly PROFILER 6600	No	$29,160	Custom SW

NMEA 2000 output data is openly accessible with additional equipment and a subscription fee to access the data sentence structure output, while NMEA 0183[13] sentence structure is publicly available for free. Given the price range and time required to extract the required data from the proprietary sensors, as we will cover in Chapter 4, in all three areas we selected the less expensive and open-source solution.

2.3.3 Sensor Placement, Orientation & Calibration

Leveraging the patented speed correcting paddle-wheel sensor invented by Boucher and Frederic in 2006[14] and the insights from Schroeder's vessel hull transducer modular mounting system in 2007[65], it is readily apparent that special considerations must be paid to proper placement and orientation to maintain calibration of under-

water sensors. Using the principles referred to in these patents along with mounting recommendations from the active patent assignees, we have experimentally optimized the placement and orientation of the underwater sensors that we will see in Chapter 4.

2.4 Data Collection and Prediction

Current research modeling wind and currents mainly focuses on large scales, with applications to oceanic navigation and power generation. In the problem addressed in this work, the effects of environmental forces impacting the platform are greater considering the size of the ASV. For instance the work of Soman *et al.* [70] reviewed existing wind prediction strategies for optimizing efficiency and profits in power generation applications. Their work focuses both on long term and short term forecasting, but mostly as it pertains to direction, since the turbines they are designing for are stationary. One more closely related study is by Al-Sabban *et al.* [6], which focuses on the effect of wind on Unmanned Aerial System (UAS). They implement a hybrid Gaussian distribution of a wind field and a slightly modified Markov Decision Process (MDP) to identify the optimal path and optimal power consumption trajectory for a UAS. This work may prove valuable in our future work, but at the current time, we are concerned with an environment with leeward and windward effects caused by surrounding landscape[25, 47, 35]. Another work, by Encarnaçao and Pascoal [20], studied the problem of developing control systems for marine crafts that are able to follow trajectories to track another boat, under the effects of water currents. Their models center on inertial tracking and compensation of roll, pitch, and yaw rates after the force is sensed to provide course corrections. In our work, we aim at being more proactive, in that we actively measure and model environmental variables with the end-state intent on taking corrective control measures prior to the robot coming under the effects of the wind or currents. Finally, Hsieh *et al.* [26] have provided

many contributions related to the problem of mapping the effects current phenomena with their design of a control strategy for collaborative underwater robots to track coherent structures and manifolds on generally static conservative flows [46, 34, 27, 18]. Huynh *et al.* presented a path planning method for minimizing the energy consumption of an autonomous underwater vehicle (AUV) [28]. Their work addresses varying ocean disturbances that are assumed to not exceed the capabilities of the AUV. These works, while addressing aspects of collecting and modeling dynamic environmental characteristics, do not fully address time and space considerations that impact our lightweight, small-scale ASV. To establish models that can drive future adaptive controls and online planning strategies for our ASV, we adopt the technical approach presented in Chapter 5.

2.5 Controllers

The main goal of this thesis is to provide a solution for controlling an ASV in turbulent conditions. To enable this goal, we must first establish a baseline understanding of the controllers and methods that have been used for decades in several somewhat unrelated applications. In our case the main issue is tracking. We define the tracking problem as minimizing the distance between our specified target path and the actual path traversed.

2.5.1 Proportional-Integral-Derivative (PID) Controllers

Aastrom and John's book, *PID controllers Theory, Design, and Tuning* [7] and Visioli's book[73], *Practical PID Control*, together provide a solid foundation for understanding the function and importance of PID controllers to almost any mechanical application. Several researchers provide extensions of these basic concepts for advanced tuning techniques[9, 68, 53] such as automatic tuning. This however, falls short when attempting to deploy a PID standalone controller in rapidly changing

environments.

2.5.2 Closed Loop Feed Forward Controllers

The introduction of closed loop feed forward controllers[37, 24, 8] seeks to bridge the gap, where external disturbances to the system can be input at different stages of the controller referred to as cascade control. Novel advancements to the input and tuning of cascading control introduces an internal model control (IMC) approach for synthesizing the controller's transfer function, opposed to the single-loop[71] and cascade approaches[69, 37].

Closer to our problem set, in 2013, RASAL[61] sought to improve the path following capabilities of SWATH in environments with wind and current present. While the results reflect moderate success in moderate conditions, their off-board control system inspires our design and implementation for customized control sequences for future mission specific tasks in longer-range, more volatile environments.

Related research focussed solely on minimizing error tracking includes work from Tsu-Chin[72]. And robust digital tracking control based on a disturbance observer from Lee *et al.* [36] closely models our setup with our wind and current sensors taking on the role of the disturbance observers. Pereira *et al.* focus on position anchoring of small underactuated ASVs in windy conditions[56]. In all cases, performance was good in conditions the authors admitted to be moderate. So in order to enable accurate tracking in volatile conditions, in future work, we will extend our tracking control research to leverage more recent advances in machine learning.

2.6 Mission Planning

As part of our initial work in developing and testing the ASV, we used the platform to investigate other open research areas such as coverage. Since this dissertation focuses specifically on controls with direct support considerations for ASV coverage, we must

consider closely related constraints bearing on both problem sets. There are numerous ways to formulate coverage, including *static* or *dynamic* coverage, *complete* or *partial*, *offline* or *online* [17, 22]. In addition, there are many different approaches to tackle such a problem, such as defining it as graph partitioning problem, performing region-based decomposition, or defining it as sub-modular optimization problem [66, 22].

In our problem set, we are focusing on *online* measurements and course correction. The strategies used to plan the mission path should already take into account higher level maps and tendencies such as predominant wind and current directions and the morphology of a body of water. Once deployed into a given environment, the ASV must be able to adapt and overcome changing dynamics that were likely not considered in the mission planning.

Another closely related topic between planning and controls is the coverage problem for robots with turning constraints, a simple boustrophedon coverage plan may introduce wasted time–that is, time spent not covering an environment. The Dubins vehicle is a common robot model in coverage and Savla, Bullo, and Frazzoli [64] consider a control-theoretic solution. Lewis *et al.* [38] and Kareptyan *et al.* [29], however, provide an algorithmic approach to minimizing path length by minimizing the time spent not actively covering. In a similar manner, the controls and augmented controls must constantly contend with the Dubins vehicle turning radius constraints.

Given this background and the void in ASV development for autonomous control in dynamic environments, we seek to use inexpensive sensors and leverage geometric relationships to systematically measure, model, and countermeasure environmental dynamics resulting in an ASV capable of closely following a desired trajectory.

Chapter 3

LONG-RANGE, MEDIUM-DURATION DEPLOYABLE AUTONOMOUS SURFACE VEHICLE

Environmental monitoring of marine environments presents several challenges: the harshness of the environment, the often remote location, and most importantly, the vast area it covers. Manual operations are time consuming, often dangerous, and labor intensive. Operations from oceanographic vessels are costly and limited to open seas and generally deeper bodies of water. In addition, with lake, river, and ocean shoreline being a finite resource, waterfront property presents an ever increasing-valued commodity, requiring exploration and continued monitoring of remote waterways. In order to efficiently explore and monitor currently known marine environments as well as reach and explore remote areas of interest, we present a design of an autonomous surface vehicle (ASV) with the power to cover large areas, the payload capacity to carry battery weight to provide sufficient power and sensor mounting equipment, and enough fuel to remain on task for extended periods. An analysis of the design and a discussion on lessons learned during deployments is presented in this chapter.

3.1 Introduction

The University of South Carolina's Autonomous Field Robotics Lab (AFRL) Jetyak is an ASV modeled after the Woods Hole Oceanographic Institution (WHOI) Jetyak[32]. This work focuses on improving modularity and performance throughout the design

and build phases in order to expand capabilities for operating in dynamically changing environments. Furthermore, the proposed design and implementation aims to expand deployment capabilities to include highly dynamic environments typically occurring in remote, uninhabited areas. Throughout our development, we define these dynamics as wind, water currents, and depths. Along with our desire to maintain an on-board manual operation mode, this expansion is guided in increasing the diversity of the operating modes and payloads, by setting the modularity and control as core implementation requirements of the platform.

The ASV described in this thesis is based on the Mokai Es-Kape [2] boat. It is controlled using a Pixhawk PX4 micro-controller, and is capable of communicating using 900 MHz radio modems, 2.4 GHz remote control radios, and a 2.4 GHz WiFi connection. The communication capabilities enable connectivity with: a remote control transmitter, a remote computer termed Ground Control Station (GCS), and other ASVs using an ad-hoc network. The proposed design enables the following operating modes:

- **Manual operation on-board:** A human uses the manual control of the Mokai Es-Kape [2] to drive the vessel. This mode is valuable for a scientist to manually drive to a location and collect data, as well to test the dynamics of the vessel in challenging conditions.

- **Manual operation off-board:** A human uses a 2.4 GHz remote control radio to operate the vehicle. This mode can be employed to send the vessel to collect data in a challenging situation, especially when such operation raises safety concerns for a human operator.

- **Autonomous way-point operation off-board:** The boat is sent GPS way-points via a remote computer. That way a single GCS can control multiple vehicles and coordinate with respect to collision avoidance.

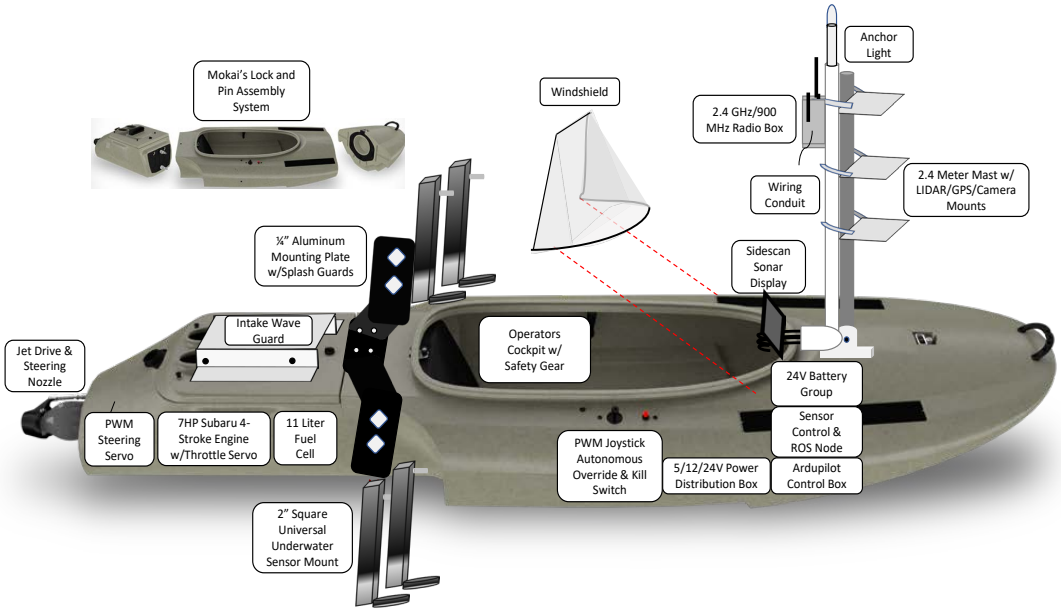


Figure 3.1: Stock Mokai Es-Kape with additional modular components added by UofSC's AFRL to accommodate a multitude of underwater and above the surface sensors. Splash guards were added in 3 locations to protect engine and on-board electronics. Navigation lights and safety equipment are included on each JetYak for Coast Guard compliance.

- **Autonomous way-point operation on-board:** A computer on-board sends GPS way-points to the vehicles micro-controller (Pixhawk PX4). Decisions are made locally, and the vehicle can operate even if the communication with the GCS is intermittent.

- **Autonomous velocity control on/off-board:** A computer uses a control algorithm (PID, adaptive, or model based) to change the steering angle and the forward velocity of the vessel based on sensory input. Such capability is critical for operating in adversarial conditions, such as high currents and strong winds.

The main contributions of this chapter lie, first, in expanding the modularity and flexibility of existing ASV platforms; providing a modular design of an ASV with publicly available documentation and software [5]; discussing lessons learned during

the construction of the vehicle and various deployments; and demonstration of our implementation's expanded capabilities through providing preliminary data collected in stable and highly dynamic environments.

The chapter is structured as follows. Section 3.2 covers current design goals and where the design goals diverge between previous approaches and the proposed design. Section 3.3 details the construction process of the base platform. Section 3.4 presents additional lessons learned not covered in the build section as well as valuable lessons learned through over 50 deployments of up to four Jetyaks simultaneously. Section 3.5 illustrates the current navigation and data collection success of this platform. Section 3.6 describes the ongoing work that AFRL is planning for increasing the platform's capabilities, and finally we conclude with Section 3.7.

3.2 Design Goals

Our design differs from WHOI with respect to expanding capabilities to include long term deployments for inland waterways with highly dynamic currents and landscape. Our design considerations include physical platform modularity, sensor mounting versatility, and controls integration flexibility. To enable these end-state goals, we researched and planned for robust communication and micro-controller platforms modified and configured to operate in marine environments with safety features such as the ability to remotely kill the engine. The physical layout must support on-board manual operation to support environmental scientists requiring a level of supervision during data collection. The platform must include the capability to host numerous above water sensors such as cameras, LIDAR, anemometer, GPS, radar, and communication components. Also, the platform must be capable of hosting at least four underwater sensors such as depth sounders, bathymetric imaging transducers, water current sensors, and cameras. On-board layout requirements include power planning for a 24 volt power source and plug-n-play distribution panel for 12 and 5 volt

devices. In addition to the factory joystick controls, the on-board footprint must include space for our autonomous and teleoperation control box, programmable control boards (PCB) servicing desired sensors and additional minicomputers. Space for on-board companion minicomputers must be retained for our on-line autonomous control interface using the Robot Operating System (ROS)[59] as a framework for software development and standardized data collection. The specific components, placement, and integration of these components is covered in Section 3.3 and illustrated in Figure 3.1.

3.3 Autonomous Field Robotics Lab's Jetyak

3.3.1 Stock Platform

Our base platform consists of the commercial Mokai Es-Kape[2] boat, whose predecessor has been previously modified and termed the WHOI Jetyak by Kimball *et al.* [32]. The latest model ES-Kape is 3.6 meters long and is propelled by a seven horsepower, four stroke, internal combustion engine, with total cost of $5,400. With its 9.8-liter fuel reservoir, the ES-Kape can operate at lower speeds for 18 hours and top speed for four hours before refueling is required. Top speed with an average payload of 90 kilograms is 21.7 kilometers per hour, and the maximum payload capacity is 163 kilograms. Additionally, the factory ES-Kape includes an improved jet drive with a clutch allowing the impeller to be stopped without stopping the engine, a modification that the WHOI team had to implement that we did not. As noted in WHOI's work, Mokai released their ES-Kape model in 2014, which includes pulse width modulated (PWM) servos for both throttle and steering controls. This electronic control upgrade allows us to forgo developing electromechanical controls. This provides direct access to the servos controlling the throttle lever and steering nozzle by piggy-backing on factory joystick controls and wiring harness. As a result, teleoperation and way-point autonomous navigation controls are able to be directly implemented. In turn, this

enables a ROS-based control interface as a gateway to our research in developing an adaptive control system for operating in highly dynamic environments.

The remainder of this section describes the physical modifications to the platform, power distribution panel as well as the robotic controller integration. While our latest design and build is shown here, it should be noted that this design includes several lessons learned throughout the first four iterations of the modified Jetyak.

3.3.2 Physical Platform Modifications

Our end-state goals of maintaining manual operation capability and robust underwater and terrestrial sensor deployment flexibility translate to challenging spatial planning and layout considerations. Additionally, through lessons learned from initial deployments, three areas that require protection from the marine environment were identified.

Interior Footprint

When received, the inside hull of the Es-Kape is a single layer of plastic maintaining the same shape as the exterior. In order to mount boxes, plan for a mast, and keep wires off the bottom of the boat where water could collect, a sub-floor of marine-grade polymer starboard was constructed. This sub-floor sits on the side steps of the hull and is fastened to the hull with stainless steel screws in the reinforced area of the bow, above the waterline. In addition, footings and tie downs were installed for the batteries and our electronics control box. Finally, the base for the mast is added directly under the top mast port for added rigidity, illustrated in Figure 3.2.

Terrestrial Sensor and Communication Platform

Another area that our design diverges from previous implementations is derived from our long-range communication and robust terrestrial sensor requirements. In order

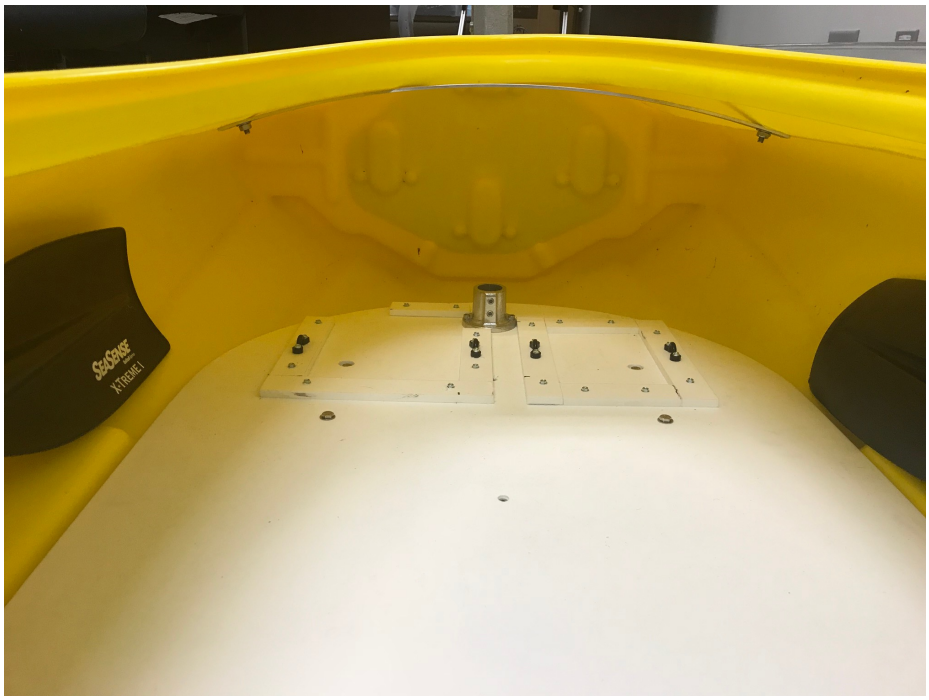


Figure 3.2: Front portion of marine grade starboard sub-floor with footings for battery, control box, as well as topside and anchor mast ports installed.

to extend 2.4GHz and 915MHz radio communication range, we experimented with different materials and mast lengths until we found the most rigid setup allowing the highest modem deployment to be 2.4 meters of fiberglass pipe. This height allows for the Jetyak to be trailered on highways and while maximizing height for longer range line-of-sight capabilities. Its rigidity, low weight and electrically non-interfering properties with the cables and antennas are desirable properties for this application. Since this mast anchors to the bottom of the sub-floor, we added a second lightweight PVC pipe to the outside to serve as a conduit for cables. As seen in Figure 3.3a, the mast is capable of supporting the radio box, lidar sensor, GPS, stereo camera, and monocular camera. The flexibility of the mast mounting strategy is illustrated in Figure 3.3b where the Jetyak hosts an anemometer on the same mast.

(a) Supervised Jetyak 3 with Ping DSP 3D sidescan imaging transducer, anchor light, communication hub, lidar, GPS, stereo camera, and monocular camera deployed. Engine wave guard and sensor splash guards are also installed and functioning.

(b) Unsupervised Jetyak with anchor light, communication hub, anemometer, 2 surface current and depth sonar sensors deployed.

Figure 3.3: Modular design allows for flexible deployment configurations.

Underwater Sensor Platform

Again, our design goals were to develop a highly modular platform capable of deploying all types of sensors without the need to retrofit or make structural changes to the base boat. We decided to develop a strong, lightweight universal outboard mounting plate to permanently attach to the Jetyak. The complementary component to such a design is the vertical mounting poles that have a universal mounting ring welded to the bottom. The plate and pole design was delivered to a local water jet facility for cutting and welding the 6.35mm aluminum plate and brackets. The bracket shown

in Figure 3.3a allows each pole to be raised, lowered or removed independently according to researcher requirements. The underwater sensor in Figure 3.3a is the 8 kg 3DSS-DX-450 sidescan transducer from Ping DSP[3].

Engine & Electronics Water Protection

As we continued to develop and test the Jetyaks in rougher lake waters and faster moving currents, we learned quickly that protection for the air intake of the air cooled engine would be required. There are two ways that water can enter the engine compartment and air intake in our design. First, and consistent with all Mokai stock platforms, is the possibility of water from waves overflowing the top of the engine. Using examples from our predecessors, we fabricated and installed a simple galvanized metal protection hood as visible in Figure 3.3a to guard against this hazard. The second hazard, due directly to our outboard sensor design, is from water deflecting up the sensor mounting poles into the air intake. This is overcome by cutting 3.175mm thick Lexan plastic to mount under the plate and extend forward and rearward of the plate to deflect water back away from the engine. Finally, as seen in previous implementations where humans are part of the payload, we integrated a windshield to abate spray from the front of the boat away from occupants and electronics.

3.3.3 Power Distribution

To meet the requirements of many high end oceanographic sensors such as sonars, radars, ADCPs, and sidescan sonars, a 24 volt power source is required. We accomplish this by connecting two 12 volt deep cycle batteries in series for direct wiring of 24 volt electronics. Figure 3.4 illustrates the power sources for each on-board component. We provide 12 volts from one of the bank batteries to our power distribution panel. Within the power distribution panel, we add an additional step down of the 12 volt line to five volts for our Pixhawk, Raspberry Pi, and Arduino PCB power

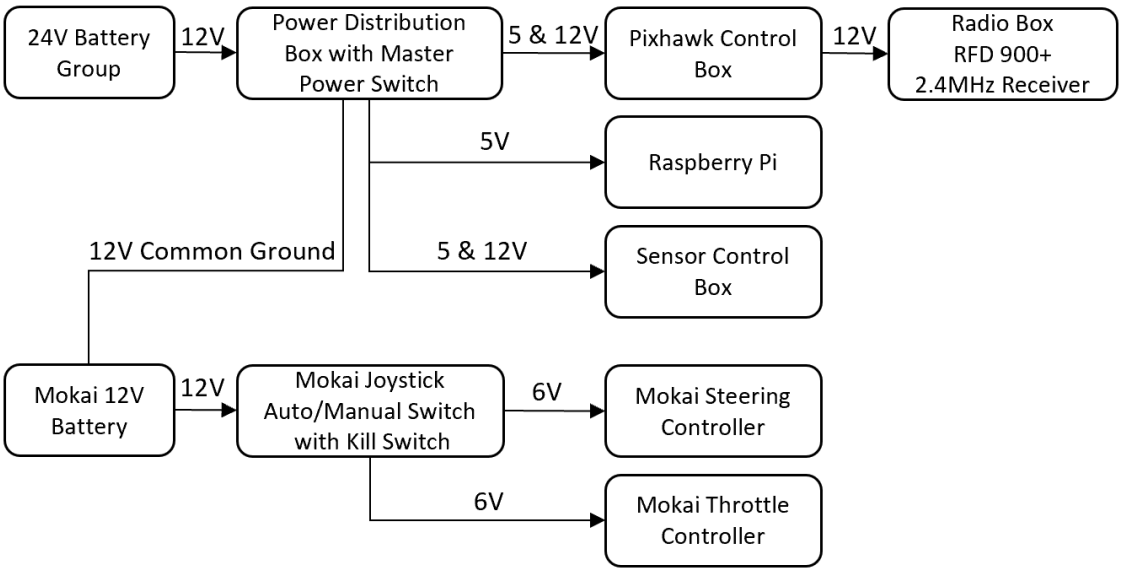


Figure 3.4: Power distribution diagram with our 24 volt power bank added to the Mokai factory system.

supplies. This initiative is a result of a lesson learned after the addition of more than three sensors caused a confusing and cumbersome array of power and sensor wires, resulting in the damage of one Pixhawk PX4 and one Arduino UNO due to water intrusion. A second lesson learned after witnessing some erratic servo behavior when controlling through the Pixhawk, was the requirement of ensuring a common ground ties both systems together. Since on a watercraft there is not a true ground but rather a floating ground reference, providing a common ground connection ensures the factory servos and our added control system maintain the same zero voltage reference. Once the electrical bugs were identified and corrected, the integration of the power distribution panel resulted in a clean plug-n-play system which also provides better durability when deployed with a human on board.

3.3.4 Robotic Control Integration

Our requirement to maintain five methods of operating the Jetyak required significant design and planning. The following subsections describe our design and build of

the Jetyak with a natural progression from preserving manual operation to remote control teleoperation and way-navigation to our current work in developing adaptive controls on a ROS node. The schematics, PCB controller code, diagrams, pictures and configurations are included in our open source Jetyak tutorial page at the AFRL resource page [5].

Integrating with Factory Components

In a worst-case scenario, we ensure that the Mokai Jetyak maintains its factory manual operating capability. This drove our decision to place a manual/auto switch in the factory joystick control box that would always allow us to take over manual control of the boat. As illustrated in Figure 3.5, there are two intersections of our equipment with the Mokai's controls in the joystick box.

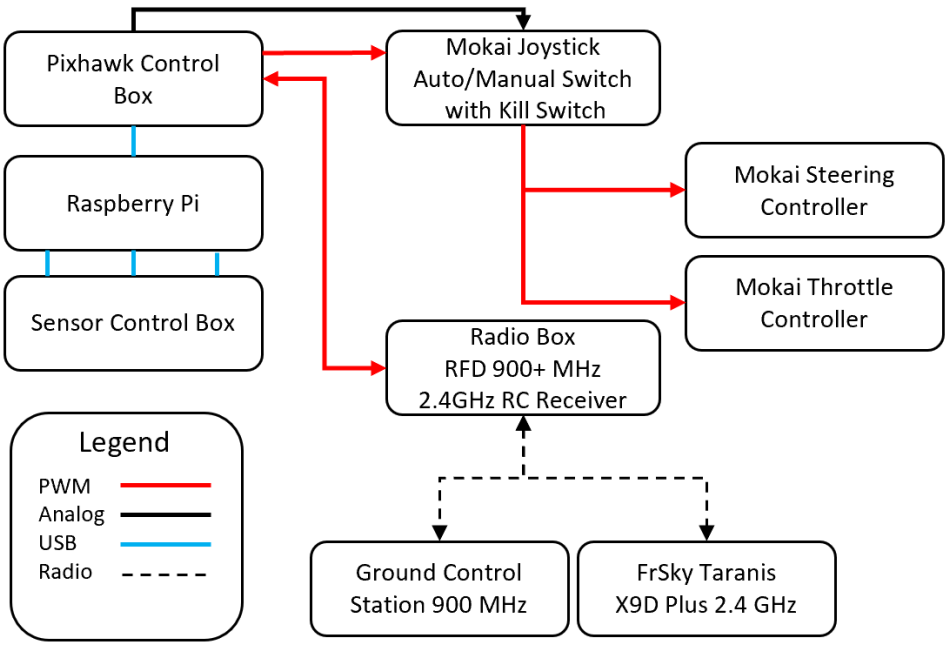


Figure 3.5: Jetyak controls architecture, illustrating the integration of factory, on-board, and off-board components.

The PWM wires connect the steering and throttle outputs of the Pixhawk to

the autonomous side of manual/auto switch and the factory PWM outputs of the joystick are attached to the manual side. At this point the factory 6 volt and ground connections are passed through with the manual/auto switch output on the factory harness to the servos. There are two advantages to this implementation. First, we reduce the footprint and exposure of our additional PWM carrying wires to the protected area in the factory joystick box. Second, by passing through the factory voltages, we eliminate the need to step up our five volt control voltage to the six volts required by the servos.

Remote Shutdown and Power Loss Safety Circuit

Although our goal is toward a fully autonomous Jetyak, safe testing and deployment requires a method for remotely shutting down the Jetyak in case of emergency or imminent crash. To accomplish this, we provide a parallel kill analog control connection to the factory circuits. Kill or shut down is accomplished through closing a loop which shorts the engine magneto to ground. We emulate this active low behavior through programming a digital channel on the Taranis radio and in the Pixhawk to normally operate in the high state, and when kill is activated, change to the low state. Using this output of the Pixhawk as the coil input for a relay results in the relay being energized during normal operation. When the Pixhawk signal goes low, normally closed contact is made in the relay. We provide this circuit in parallel to the factory system so that if either our system or the factory kill switch is activated the boat shuts down. It should be noted that this configuration also shuts the boat down when power is lost to the Pixhawk. As this remote safety feature may not be desirable when manually driving or recovering a Jetyak, we provide a physical override switch on the side of the Pixhawk box to effectively disable this feature.

Baseline Teleoperation and way-point Navigation

At the heart of our design, we selected 3DR Pixhawk 1 running PX4 on the NuttX[4] operating system, along with the ArduPilot Software Suite[40] to enable teleoperation and way-point navigation capabilities, shown in Figure 3.6 in its Pixhawk box. This version of the Pixhawk includes an internal compass and external I2C (inter-integrated circuit) compass port for an external compass, included with most GPS antennas. Configuration of the PixHawk as a Rover [1] allowed us to start from a point where the throttle and steering servos on the stock Mokai Jetyak are directly imitable. Both the Hitec HS-5485HB[1] throttle linkage servo and the Torxis i04903[2] steering servo are controlled by modifying the Pulse Width Modulation (PWM) values in the PixHawk to match their operating specifications. The latter steering servo is internally controlled by the Polulu Jrk21V3 USB motor controller, which allows the

[1] `https://hitecrcd.com/products/servos/discontinued-servos-servo-accessories/hs-5485hb-standard-karbonite-digital-servo/product`

[2] `https://gearwurx.com/product/torxis-industrial-outdoor/`

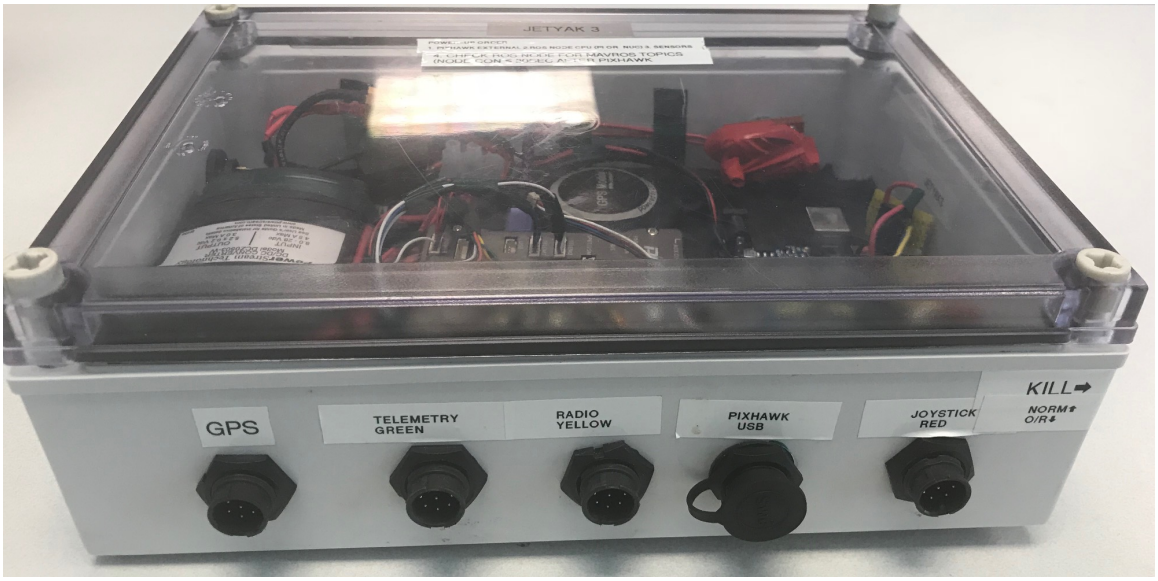


Figure 3.6: Pixhawk Box with power conditioner, Pixhawk PX4, Arduino Uno, GPS, joystick outputs, remote control interface, MAVLink -MAVROS node proxy inputs/outputs collocated.

user to access the PWM cutoffs and allows direct calibration between the mechanical steering angle and the input signal. This results in one valuable lesson learned, in that electromechanical devices with identical specifications operate differently given the specification tolerances. In order to overcome these challenges, it is extremely beneficial to measure the factory PWM output widths at minimum, maximum and center with an oscilloscope prior to attempting to teleoperate the servos. This allows the parameters for the throttle and steering output channels on the Pixhawk to be properly aligned from the start. Another lesson learned is to ensure the orientation and calibration of the compass(es) are accurately completed, since different manufacturers of external compasses may assume a non-forward mounting orientation. The second enabling component to our implementation is the Taranis X9D plus radio system, which offers great flexibility in programming explicit controls of up to 16 channels when paired with the Taranis D8R-XP receiver. In our case, we use the community standard of channel one to control steering and channel three to control throttle. We use channel six to provide our teleoperated kill capability, and channel five to control the mode of operation. The last step to enabling remote control operation of the Jetyak is to configure the appropriate channel outputs in the Pixhawk to match behavior characteristics required for the controlled device, *e.g.*, servo PWM minimum and maximum thresholds, PWM trim (neutral) position and forward/reverse direction. The highly modular programming interface allows for fully customized servo and switching controls based on logical functions, making the remote control capabilities very granular. For instance, there are five desired control modes for our application from manual progressing to guided or off-board control which cannot be programmed with a single 3-position switch. The X9D is capable of assigning a distinct PWM signal reflecting the logical result of the positions of a 3-position and 2-position switch. This example is completed when the Pixhawk is programmed with the corresponding mode functions for the received PWM signal on

that channel. As a result, the Jetyak is capable of being supervised when testing autonomous capabilities and can always be overridden, which is a necessary feature when conducting field trials in the public domain. Next, we will describe the platform enhancements we have added to enable greater autonomy beyond way-point following.

ROS Integration

Initially, the platform was used to conduct preplanned missions, collect data, and return to its home location. To accomplish this task, we only need a common time source, location and the desired sensor measurements to be synchronized. Since the goal is to produce an autonomous Jetyak for deployment in highly dynamic environments, we preplanned implementation of a system that could collect the information in a format that would be available for online usage. Luckily, there exists a well-supported, open-source solution readily available to support our needs, ROS. ROS is a robust middleware providing a framework for publishing and subscribing to topics and messages between different process, low-level device controllers and on-board computers. In addition, it provides a package management environment enabling add-on packages such as MAVROS to interface with many off-the-shelf controllers such as the Pixhawk. This allows access and integration with IMU, GPS, heading, velocity, pose and several other Pixhawk telemetry topics. These topics are then published by the on-board ROS node through USB connection on its host Raspberry Pi or Intel NUC. ROS also accommodates the addition of our depth sonar, current speed, and anemometer measurements directly into the same ROS framework. We have included depth, wind, and current sensors as a standard component to our Jetyak design, enabling operation in highly dynamic environments. Lastly, in order for the Jetyak to use sensor measurements for on-line path planning, the ROS framework provides an integration of sensing and acting commands. More specifically, ROS

integration provides a topic publishing conduit for sending general navigation as well as channel-level steering and throttle control commands directly to Pixhawk using MAVROS and the MAVLink protocol. The specific approach is up to the specific application.

Robust Communication

AFRL's Jetyak maintains three forms of communication to allow interfacing and programming at different levels and distances. Short-range communication is maintained through 802.11g wireless ad-hoc connectivity to the NUC and Pi devices. Remote teleoperation and low-level telemetry communication is provided through the FrSky Taranis X9D Plus transmitter to D8R-XP receiver radio link operating in 2.4MHz spectrum. While the long-range communications is provided through RFD 900+ MHz modems, with one as a base station and one modem per deployed Jetyak node. With the addition of our 2.4 meter mast, we have been able to extend our line of sight communication with the base station to 2.8 kilometers. Note: While not a best practice, it is possible to allow the Pixhawk to continue its programmed mission without this communication link. Appendix A discusses our side project where we experimentally improve the throughput of the 900 MHz modems to support multiple ASVs operating on the same network.

Initial Tuning Requirements

Initial deployment and testing included manual refinement of steering and throttle servo proportional, integral, and derivative (PID) controls to establish the reliable path following capability of the ASV. The starting point and manual procedure for this tuning is included on the AFRL's Jetyak tutorial[5]. A good starting point for steering PID parameters are shown below:

- STEER2SRV_D 0.005

- STEER2SRV_I 0.2
- STEER2SRV_P 2
- WP_RADIUS 5 meters

The manual procedure we used to tune the PixHawk PID coefficients in calm water follows:

1. ASV turns too slowly, increment proportional gain.
2. ASV oscillates more than three times before finding target line, decrement proportional gain.
3. ASV oscillates more than 0 and less than 3 times before finding target line, increment derivative gain.
4. ASV oscillates small amounts at high frequency, decrement derivative gain.
5. ASV still oscillates at lower frequency, decrement integral gain.
6. ASV starts to turn before reaching way-point, decrement way-point radius.
7. ASV starts to turn too far after reaching way-point, increment way-point radius.

Note: We began by adjusting the proportional coefficient.

Once tuning the PID controller coefficients is complete, we were able to deploy a single Jetyak on way-point tracking missions. These missions were created from a Dubins vehicle grid search coverage algorithm developed in UofSC's AFRL[29] shown in Figures 3.7 and 3.8.

3.4 Lessons Learned

In addition to the implied lessons learned in Section 3.3, we will discuss other valuable lessons learned during the building and deployment of a fleet of Jetyaks. First, the

Figure 3.7: An ideal North-South grid search mission used to provide a baseline for measuring tracking performance with minimal turning radius of 5 meters.

lessons we have learned in building the first five Jetyaks of UofSC's fleet will help anyone seeking to develop their own Jetyak avoid some of the pitfalls that cost us time and money in terms of replacement costs and repair time. The second general area, often overlooked, is the lessons learned during field testing any platform in the real world. In our specific design and implementation, we learned some valuable lessons associated with the outboard sensor mount that must be understood and overcome to collect reliable, consistent data. Lastly, time and resources should be allotted for maintaining the fleet. Ignoring these lessons often costs precious time, especially when considering the logistics involved with hauling and launching one or more boats.

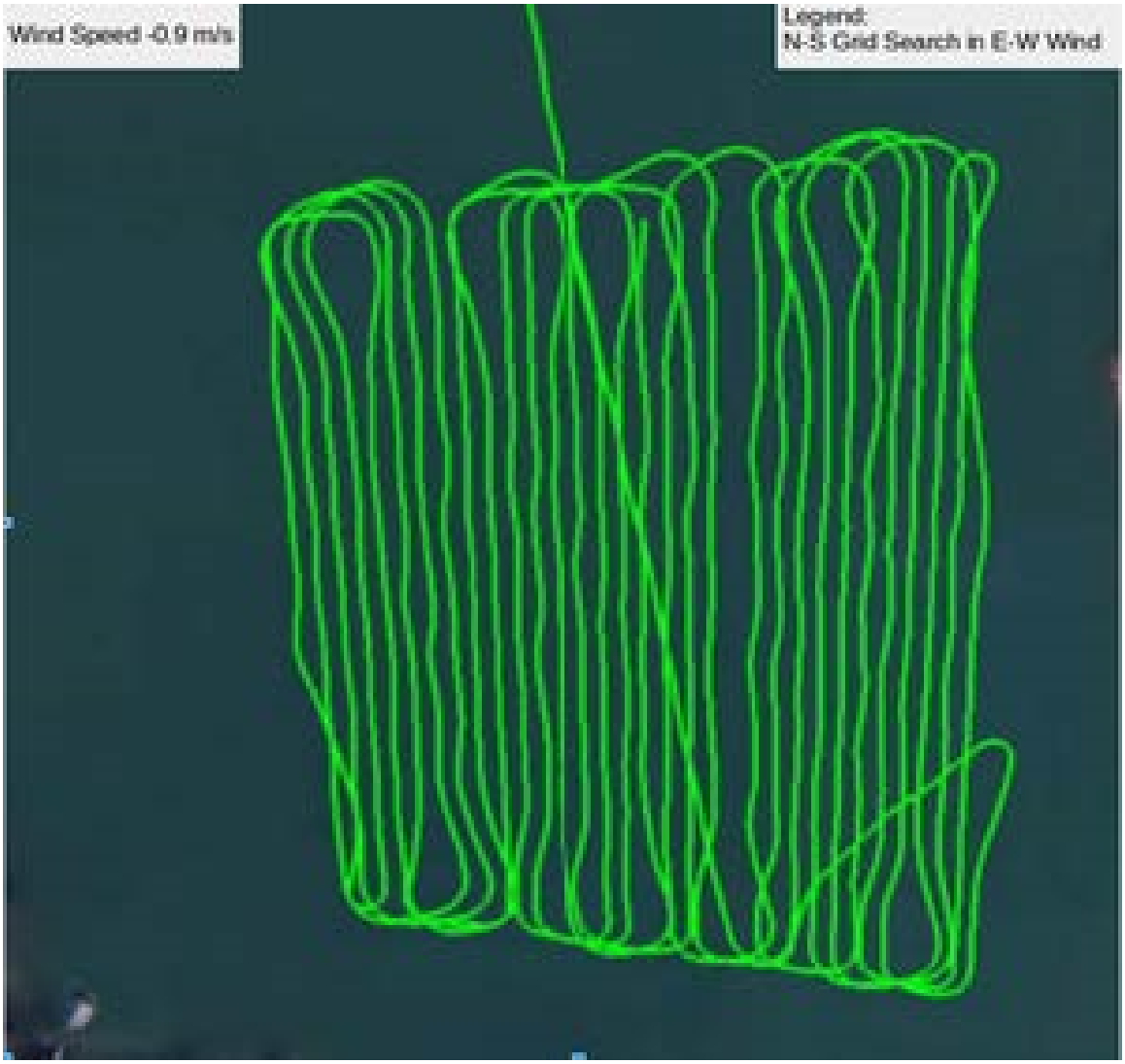


Figure 3.8: Trajectory for grid search conducted by a single Jetyak with tuned PID coefficients.

3.4.1 Building a Fleet instead of a Single Vessel

While attempting to build a fleet of Jetyaks for the AFRL laboratory, we forcefully adopted several best-practices to reduce repeatedly making the same errors. The following few subsections, outline these hard realizations.

Electromagnetic Interference

When working with an internal combustion engine, the magneto introduces interference. In the first iteration of building a Jetyak, we implemented a separate auto/manual switching box that used the factory joystick outputs and our Pixhawk control outputs as its inputs and used our in-house fabricated cables to connect to factory ports on the engine bay. After the second day of testing continued to produce unpredictable behaviors, we began monitoring the switch box outputs with an oscilloscope to find that when we switched the system to auto (Pixhawk) signals, an inordinate amount of noise was introduced. This can have catastrophic effects when working with PWM signals. If the last signal sent happens to correspond to the servo manipulation for accelerate, then the servo will continue to hold that position until overridden. In order to rectify this, we designed a system with the same signals, but this time, eliminating any non-factory wiring beyond the outputs of the control box. The results were much cleaner signals in both operating modes and stable behavior.

Maintaining Compass Accuracy

The Pixhawk is capable of maintaining two compass headings, its internal compass and an external. Depending on the quality of the external I2C compass purchased, in our case 3DR, it is often beneficial to assign priority to externals. In order to compare the reliability of two compasses, simply select each compass as the primary compass in Mission Planner and select the one that drifts the least in a stationary environment. Finally, when the compass is mounted in an orientation not aligned with the Jetyak, it is essential to input the axes orientation and calibrate the compasses in Mission Planner to ensure the proper offsets are maintained.

Repeatability & Quality Control

A large scale project such as this will thoroughly test any lab's methods for ensuring efficiency and best practices. Recognizing differing methods for maintaining best practices, some examples of areas in the project that will consume time and money follow:

- **Standardized Wiring Color Scheme:** Develop a standard wiring color scheme beginning with the factory joystick box scheme.
- **Documentation Standards & Sharing:** Developing a standard for real-time collaboration and sharing of design changes is crucial when lab turnover occurs.
- **Adopting Industry-like Quality Assurance & Control Standards:** Establishing a quality control mechanism within the build team will save several hours of troubleshooting the dozens of circuits required to make this architecture function.

Repeatability

While every attempt was made by our small team developing the first iteration of the Jetyak to capture every change, it did not take long into the build of the second Jetyak to realize we did not document each change and to compound errors, did not standardize the wiring color scheme during testing and troubleshooting. One of the most time consuming and frustrating realizations in development of such vessels are the hours spent testing individual signals just to find an error that would have been obvious with a better standard schematic and wiring going in. Coupled with the turnover of personnel in most labs and differing levels of experience, it has become standard for anyone modifying or designing changes to AFRL's Jetyak to adhere to our standard scheme. It is worth noting that while an commercial operation has such standards in place, in an academic setting, where undergraduate students work and

at the same time are educated, enforcing the above mentioned standards will reduce errors and improve the education of the students in training.

Quality Control

Attempting to build a fleet of six Jetyaks in a relatively small lab has forced us to adopt a quality control hierarchy within our team. As with all processes, it is an invaluable practice to create a hierarchy of individuals responsible for validating each portion of a circuit prior to placing circuit into the overall architecture. Quality control of measuring for proper voltages and ensuring PWM outputs are in the expected ranges will save hours of troubleshooting as each phase of testing is completed.

3.4.2 Real-world Deployment

Field deployment of the Jetyak brought a new level of learned lessons to our team, especially in the domain of highly disrupted marine deployments. These lessons fall into three categories expanded below: field trial deployments, outboard sensor deployment, and maintenance.

Field Trials

The logistics required to plan and safely execute field trials with one or multiple Jetyaks cannot be overlooked. After our first attempt to deploy a Jetyak at Lake Murray, SC in a generally calm and stable environment, we developed a comprehensive startup checklist to ensure all components were operational in a sequential manner. The general component groups we test are the steering, throttle and kill operations as well as ground control station connectivity in manual, remote control, and autonomous modes. Each aforementioned test is carried out with the boat engine off as well as running for thoroughness. Before deeming the Jetyak ready for launch, we ensure our ROS node is operational and receiving all required MAVROS

and sensor topics. Our field trial log sheet is included as an appendix to AFRL Jetyak tutorial[5].

Outboard Sensor Orientation

Other physical phenomena we contend with are the resulting cavitation and aeration effects of moving a body through water. Cavitation must be considered when deploying physical measurement sensors such as current sensors, and aeration will quickly become the enemy of sonar based sensors, causing erratically high or undefined readings. Effects of aeration and cavitation are illustrated in Figure 3.9. In our case, several trials were required to find the best location and orientation with relation to the ASV to ensure accurate readings. Generally, the sensor needs to be mounted slightly deeper than any hull of the boat traveling in-front of the sensor, and away from the disruption area of the propulsion system. In addition, the mounting pole of the sensor should be mounted behind the sensor. These two tactics allow unperturbed water to cover the bottom of the sensor. Planning for and reducing sensor exposure to the effects of cavitation and aeration will save much frustration and time

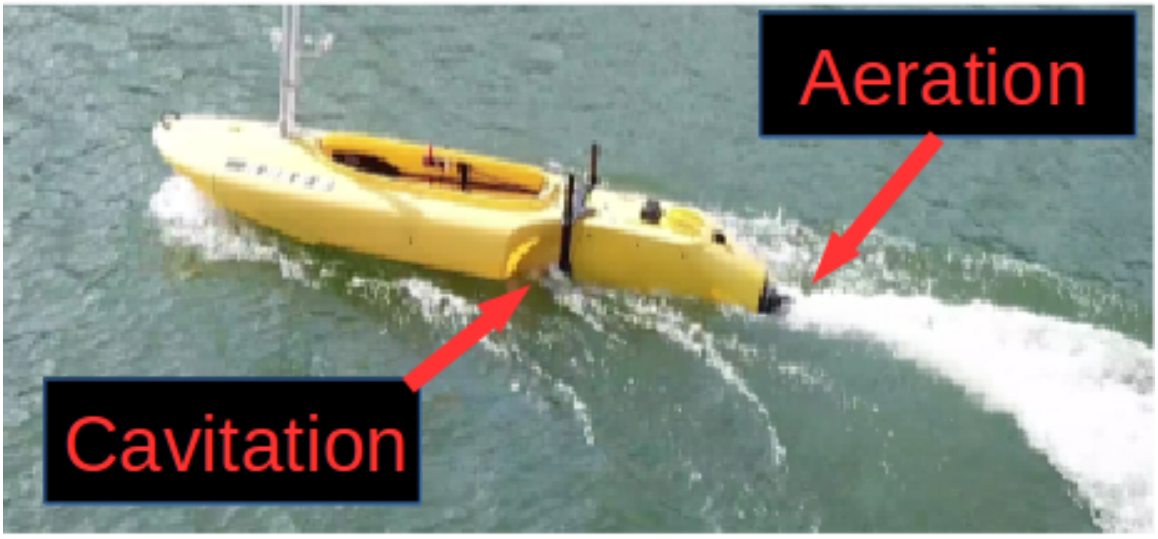


Figure 3.9: The effects of cavitation from the hull and aeration from the sensor and mount acting on a single paddle wheel speed sensor on a Jetyak.

lost in trips to and from the launch site for future builders.

Maintenance

Proper routine maintenance of the Jetyak will ensure proper mechanical operation for the next deployment. Tasks such as topping off fuel, checking and changing the oil when required, charging batteries, greasing the drive shaft coupler can be completed days or weeks prior to the next deployment. These tasks are also captured on our startup checklist included in the tutorial.

Lastly, if excessive water does make its way into the engine air intake, impromptu maintenance must take place otherwise catastrophic failure may occur. When this happens, the engine should be stopped and the boat returned to safety where the engine can be removed and the oil changed several times until the milky appearance has disappeared. Due to the modular design of Mokai's lock and pin assembly, it is feasible to include an extra engine or engine box as part of the field trial support package to reduce downtime if this does occur.

3.5 Results

In this section, we provide some examples of field deployments we have completed with the Jetyak. Initial sensor payload includes different combinations of Ping DSP sidescan sonar, Humminbird Helix 7 sidescan, Velodyne lidar, stereo camera, depth sonar, anemometer, and current sensors. The measurements and predictive data illustrated here is still under analysis and development for future improvement goals. Nonetheless, they give some intuition into the utility and versatility of the Jetyak for exploration and task focused data collection. Real world data collection results include sonar, anemometer, and surface current measurements in stable environments on Lake Murray, SC and the highly dynamic environment of the Congaree River near

Columbia, SC. In addition to the raw measurements, initial mapping and prediction capabilities are illustrated for close temporal planning.

3.5.1 Stable Environment Deployments

Initially, we deployed a single Jetyak on Lake Murray in South Carolina with a single sonar depth sensor to test the Jetyak's performance from launch to autonomous operation to data collection and logging. Afterwards, multiple Jetyaks were used as an experimental setup in the work by Karapetyan *et al.* [29] for deploying a Multi-Robot coverage algorithm with Dubins kinematic constraints. The trajectories of each robot are illustrated in Figures 3.7 through 3.10. The coverage was performed locally, by each Jetyak tracking preassigned way-points programmed in the mission planner. Given the recent addition of Jetyak 3 to the fleet at the time of this experiment, the lack of time to tune the PID coefficients resulted in Jetyak 3's erratic behavior in Figure 3.10. The resources required to maintain accurate tuning and overcome this behavior for the entire fleet along with how naturally occurring disturbances (wind, current) adversely affect our ASV platform has motivated our future work in adaptive controls.

Results Illustrated Through Multi-Robot Coverage Experiments

Once the intial platform was developed with basic autonomous way-point navigation capabilties, we initially deployed to support multiple robot coverage algorithm field trials. A variety of experiments were performed using teams of two or three robots in different areas of the Lake Murray; see Figure 3.13 for an instance of the three ASVs in action. These results illustrate the early testing and verification of the initial platform capabilities in way-point navigation mode.

Figures 3.14a and 3.14b shows the ideal path for two and three ASVs as generated by Kareptyan's Dubins Coverage with Route Clustering (DCRC) algorithm; while

Figure 3.10: Trajectories for three Jetyaks searching their respective areas of responsibility according to mission resulting from implementation of our multi-robot coverage algorithm for Dubins vehicles.

Figure 3.11: GPS path of the ASV during coverage.

Figures 3.16a and 3.16b shows the actual path followed by two and three robots, respectively, which deviated from the ideal path presented in Figures 3.14a and 3.14b. GPS error, current, wind, and waves from other vessels were expected sources for these deviations. The ill-structured path of one of the robots (robot following the blue trajectory) results from hysteresis of its on-board PID controller, illustrating the real world challenges with field trials. In particular, it shows that, even if the boats are supposed to be identical, they are not, and they should each undergo an initial

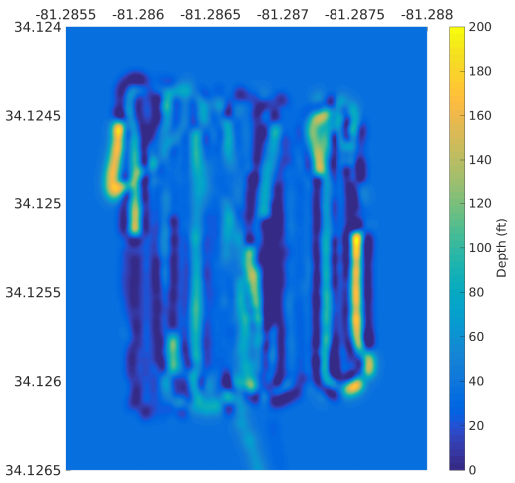


Figure 3.12: The lake-floor reconstruction from the single ASV coverage.

Figure 3.13: Three ASVs during multi-robot coverage experiments.

tuning phase of the different parameters of the boats. Such an issue is an interesting research direction, when working with multiple heterogeneous robots.

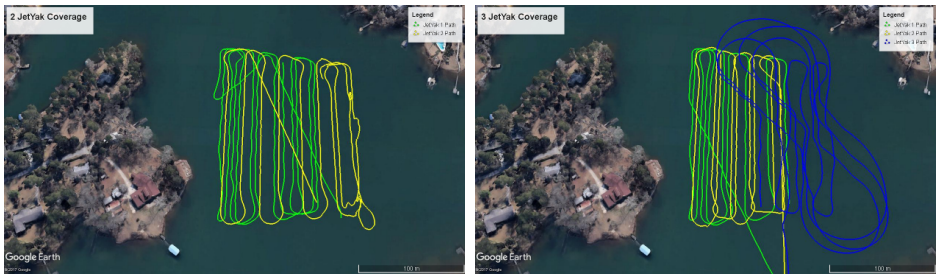
3.5.2 Highly Dynamic Environment Deployments

As we extended our platform to operate and collect data in more volatile environments, we are able to complete our data collection goals in water currents reaching nearly 3 m/s. The PID controller is able to track way-points against the current by slowly working against the current until it reaches the desired point. However, the Jetyak misses several way-points that are downstream or cross stream where faster moving surface currents exist. This experimental realization has reinforced our desire for future work in adaptive controls. Figure 3.17 illustrates a fleet of four Jetyaks

(a) Ideal path produced for two robots.

(b) Ideal path produced for three robots.

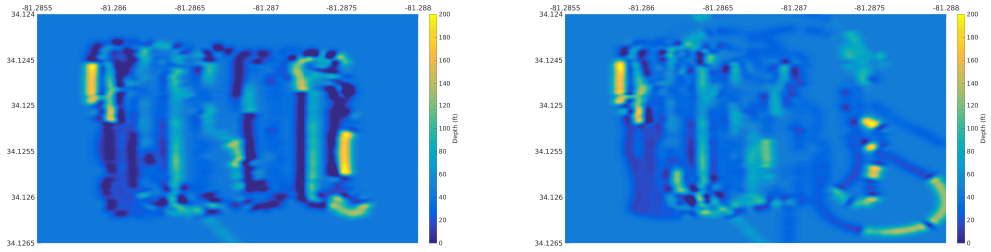
Figure 3.14: Ideal paths from algorithms as waypoint input to the ASVs.



(a) GPS track of the actual coverage path for two robots.

(b) Ideal path produced for three robots.

Figure 3.15: GPS track of the actual coverage path for three robots.



(a) Depth map produced using a GP-based mapping using data from two robots.

(b) Ideal path produced for three robots.

Figure 3.16: Depth map produced using a GP-based mapping using data from three robots.

operating in these adverse conditions.

In addition, we have successfully deployed sidescan imaging sensors similar to the one on the MIT SCOUT using the modular poles without modification to our universal mounting bracket.

Figure 3.17: Four Jetyaks operating autonomously in the Congaree River near Columbia, SC.

3.6 Progression

Our work builds on the demonstrated capability to collect measurements of depth, wind, current, side scan images, lidar data, and stereo camera images, by enabling on-line methods for control as an augmentation to the PID controller. Yang *et al.* [75] published similar work focused on reactive controls once the phenomenon has affected the ASV's course. Expanding on their work with ocean-going vessels, we focus on deployments in ports, tributaries, canals and rivers to enable exploration and monitoring of remote waters. In addition to providing the ability to generate models of the environment, we are exploring Gaussian Process based techniques to predict and model current and wind disturbances in short temporal windows to enable proactive controls for deployments in highly volatile situations.

3.7 Conclusion

In this chapter, we have shown the design and build of AFRL's Jetyak including the design considerations, components, build details, and lessons learned. The AFRL Jetyak is the result of customizing a commercially available Mokai Es-Kape, including design considerations and comparisons to previous pioneers from WHOI, MIT, WPI, and Santa Clara University in their similar implementations. We demonstrate the utility of our design and build in demonstrations in both stable environments as well as highly dynamic environments. We illustrate our future work with this platform through identifying its current limitations of maintaining accurate trajectories in environments with high winds and surface currents; seeking to provide a solution that will allow deployment in such environments through adaptive controls. Finally, we provide a publicly available tutorial [5] with component lists, vendors, costs, and pictures to enable interested marine domain researchers to duplicate the presented system.

Chapter 4

INEXPENSIVE SENSOR IMPLEMENTATION TO ENABLE ACCURATE MAPPING AND FORECASTING

In order to overcome environmental dynamics negatively affecting ASV performance, we must, at a minimum, have a method for measuring the forces impacting the physical platform. In the surface vehicle domain, these forces are quite obviously wind and current. In addition, we have included depth in our input variables, since there is a correlation between water depth and currents. In order to meet our stated goal of providing an inexpensive overall solution, we chose the sensors for the reasons given in Section 2.3.2. This chapter focuses on the implementation, calibration, and mounting of our selected sensors on the Jetyak.

4.1 Implementation

The first and easiest sensor to configure for outputting serial data for use in later topics is the depth sensor. As stated in the background chapter, the NMEA 0183 specification of the CruzPro ATU120AT (Figure 4.1) requires serial output, which is easily converted to a USB connection using any RS232 conversion cable. This allows the sensor to be powered from our power distribution panel with its data transmission to be connected to our companion computer's USB ports. In this case, when we sought to validate the readings, we did so in two methods. Our brute force method took relatively shallow measurements up to 12 feet in known depth pools and

lakes. In order to validate the deeper portion of its specification (150 meters), we conducted side by side comparisons with trusted and much more expensive sensors. Being that this sensor is well established and trusted in the commercial world, it is of no surprise that it is accurate out of the box and requires no calibration or offset, unless mounting depth is changed.

Figure 4.1: CruzPro ATU120AT through hull sonar transducer for depth measurements.

The second easiest sensor to implement is the anemometer, since it operates in a single domain and provides wind speed and direction through hall effect sensor for the speed and reed switch interrupts for the direction. The SparkFun weather vane we selected in shown in Figure 4.2. The code in Appendix B.2 provides serial output of wind speed and direction and is configurable to increase frequency as required. In our case, the optimal reporting frequency is four readings per second. The Aduino Uno and weather shield riser micro-controller host the driver code for this application and are easily configurable within the free Arduino IDE application. The calibration method for validating our wind sensor accuracy is very simple. The direction can be directly manipulated and verified and adjusted by comparing to a stationary compass. The wind speed validation consisted of a trip to the local ground weather reporting station to place the sensor in close proximity to their weather vane and adjusting the coefficients in the Arduino driver until they matched.

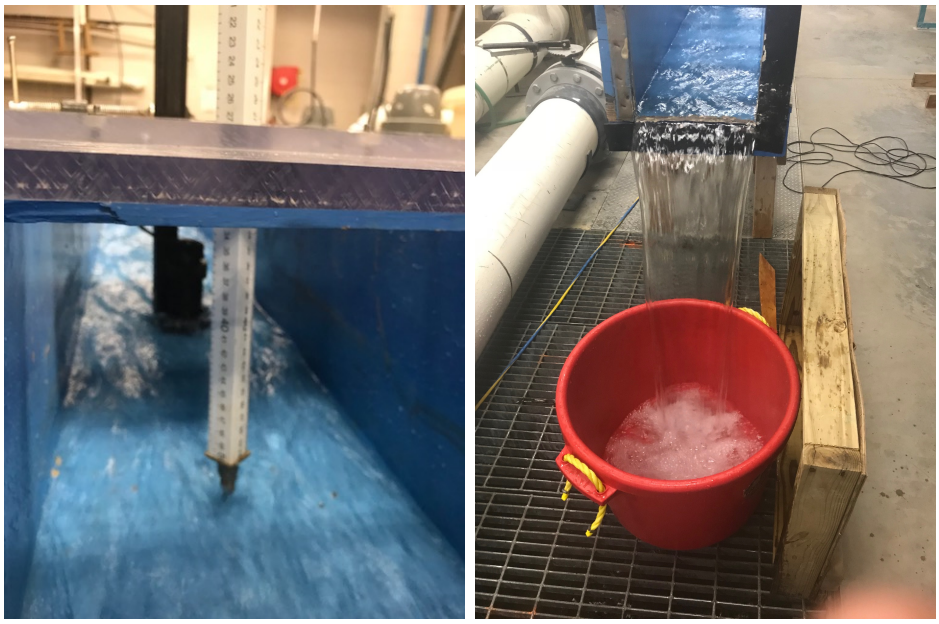
Figure 4.2: SparkFun anemometer mast mount for wind speed and direction measurement.

Finally, the last and most challenging sensor to configure is the paddle wheel current sensor. Since the sensor from Raymarine (Figure 4.3) was not so proprietary that they provided us with the operating frequency, we were able to program another Arduino UNO to interpret its hall-effect output clicks as interrupts and directly calculate the velocity of the force on paddle wheels. Appendix B.1 reflects the final implementation version of this code. Unfortunately, the challenges with calibrating this sensor are much more complex than that of the depth and wind sensors.

Figure 4.3: Raymarine ST800-P120 through hull paddle wheel speed sensor transducer.

Initial current sensor testing was conducted in the flow lab of UofSC's Civil En-

gineering Department(Figures 4.4 and 4.5.



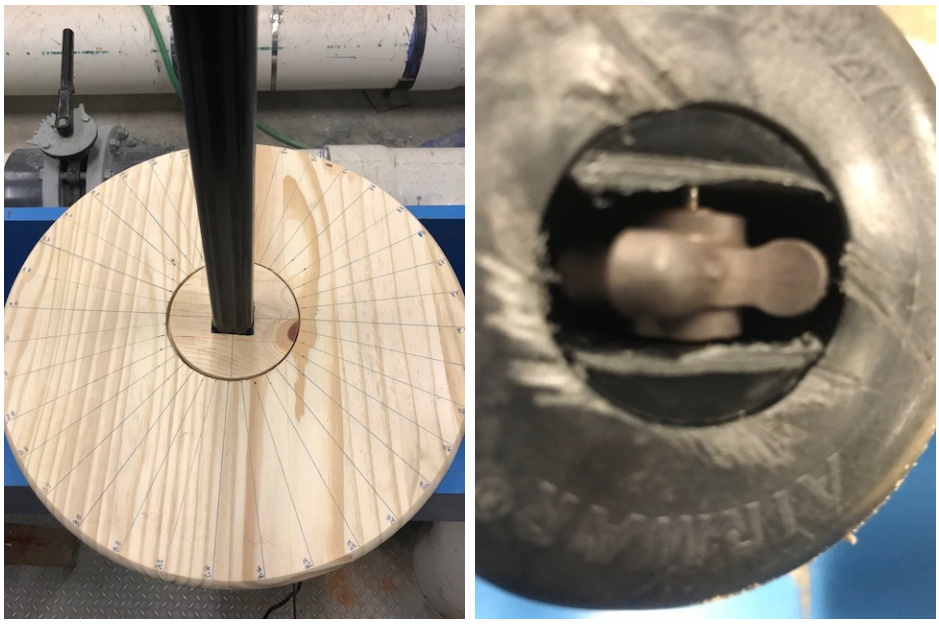
(a) Measuring the cross section of the water flow on the flow table.

(b) Measuring the time the flow takes to fill a known volume.

Figure 4.4: Calculating discharge rate of the table allows for close approximation of the surface current speed in the center of the flow table.

Using calculated flow rates to verify our measurements are on the same order of magnitude, followed by time of travel comparison for extreme lightweight low displacement floating objects traversing the table, we were able to tune the current sensor driver's coefficients to be accurate for currents measured in the upstream direction. Once an accurate current measurement was established, sensors were rotated in the flow to establish the speed readings when the sensor was impacted by a non-direct vector. In this case the factory sensor was only designed to measure the forward velocity of the current. In order to enable the sensor to provide readings for angular velocities, a slight modification was made to the transducer to remove the fins that deflect angular currents away from the paddle wheel, as shown in Figure 4.5b.

Figure 4.6 illustrates the points where the expected sin and cos vectors hold and do not, establishing our effective range of 90 degrees, 45 on each side of forward for each sensor. The first deployment resulted in accurate current calculations when the



(a) Flow test bench in University of South Carolina's Civil Engineering Lab. Fixed angle mounting allows for samples to be collected over a large period of time at the same orientation.

(b) Modified paddle wheel sensor to allow angular velocity measurements.

Figure 4.5: Sampling current speeds using our sensor allows us to determine the angles where angular velocities become unreliable.

ASV was heading against the current or in downstream cases where the ASV velocity was faster than the current. However when the ASV was using the current to reach its programmed speed, the current measurements were inaccurate using only two sensors due to the shape of the fins on the paddle wheel sensor. This led us to add two more rear facing sensors as in Figure 4.8 to provide full 360 degree coverage. The method for using the four readings to calculate the true speed and direction of the current is covered in later Chapter 5.

4.2 Mounting

Prior to developing the modular outboard sensor mounting system reported in Chapter 3.4.2, we took a brute force approach to mounting the underwater or surface

Raw Data Measurements																			
Sensor	Velocity Measured by Ray Marine ST800-P120 paddlewheel sensor (meters per second)																		
System Orientation	85°	75°	65°	55°	45°	35°	25°	15°	5°	0°	-5°	-15°	-25°	-35°	-45°	-55°	-65°	-75°	-85°
Actual Reading	.01	.33	.78	1.01	1.29	1.73	1.87	1.94	2.01	2.05	1.97	1.87	1.85	1.75	1.33	1.04	.71	.37	.06
Starboard Front (45° offset)	.45	.24	.02	0	0	.06	.37	.71	1.04	1.33	1.75	1.85	1.87	1.97	2.05	2.01	1.94	1.87	1.87
Port Front (-45° offset)	1.75	1.85	1.87	1.97	2.05	2.01	1.94	1.87	1.85	1.75	1.33	1.04	.71	.37	.06	0	.02	.24	.45

Figure 4.6: The angular velocity calculations of two sensors offset by 90 degrees. Addition of two rear facing sensors provides 360 coverage.

sensors. Figure 4.7 illustrates the evolution from brute force to intelligent design of the outboard mounting system. This modular design has proven sturdy and modular enough to host every underwater sensor we have available.

(a) Brute force early mounting system.

(b) Modular, rigid, and adjustable mounting system.

Figure 4.7: Initial testing for proof of concept resulted in a marginally functional mount system that was not rigid or adjustable enough to host our requirement for a variety of sensor shapes, sizes, and weights. This lead to the design, construction, and integration of the rigid outboard mounting system shown on the right.

As covered in Chapter 3, the last challenge we contend with in sensor integration is

the problem of placement and unintended interference from cavitation and aeration.

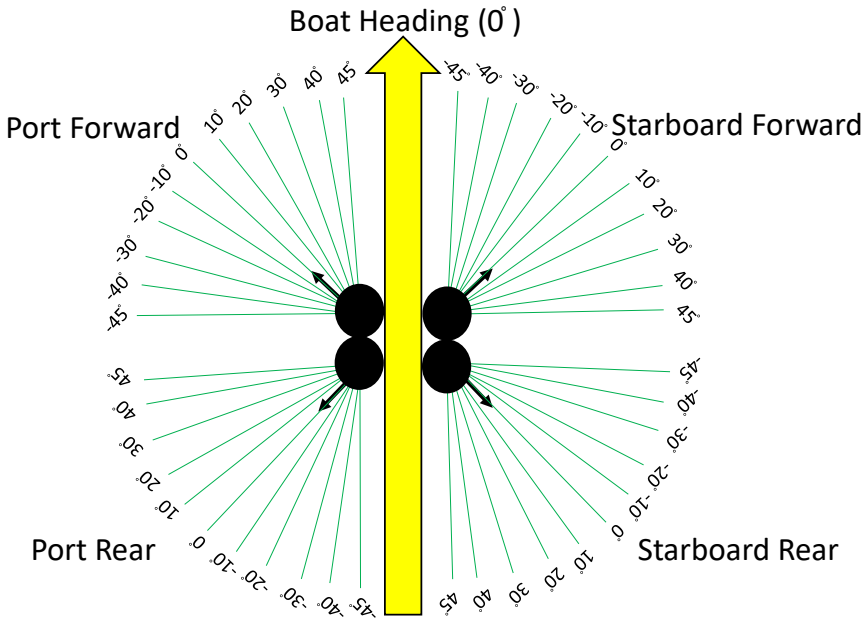


Figure 4.8: Mounting configuration for surface current sensors allows selection of the highest sensor reading for F1 and its next highest neighbor for F2. Calculations for speeds and directions are explained below.

4.3 Conclusion

The research, testing, and application of developing and integrating inexpensive sensors covered in this chapter are essential to enabling future work. In this case, the depth, wind, and current measuring methods used, directly feed the prediction methods used in the following chapter. Additionally, their accuracy and reliability are required to enable the effects modeling and proactive controls portion of this thesis in Chapter 6. In the following chapter, we will demonstrate the sensors effectiveness and provide real-world applications for their employment.

Chapter 5

Monitoring and Modelling Dynamic Environments

Operating in the presence of strong adverse forces is a particularly challenging problem in field robotics. In most robotic operations where the robot is not firmly grounded, such as aerial, surface, and underwater, minimal external forces are assumed as the standard operating procedures. The first action for operating in the presence of non-trivial forces is modeling the forces and their effect on the robots motion. In this work, an Autonomous Surface Vehicle (ASV), operating on lakes and rivers with varying winds and currents, collects wind and current measurements with an inexpensive custom-made sensor suite setup, and generates a model of the force field. The modeling process takes into account depth, wind, and current measurements along with the ASV's trajectory from GPS. In this work, we propose a method for an ASV to build an environmental force map by integrating in a Gaussian Process the wind, depth, and current measurements gathered at the surface. We run extensive experimental field trials for our approach on real Jetyak ASVs. Experimental results from measurements taken in a variety of environments validate the proposed modeling approach.

5.1 Introduction

While robots become increasingly common, from the robotic vacuum cleaner and warehouse product-moving robots, to the prospect of autonomous cars, robots are

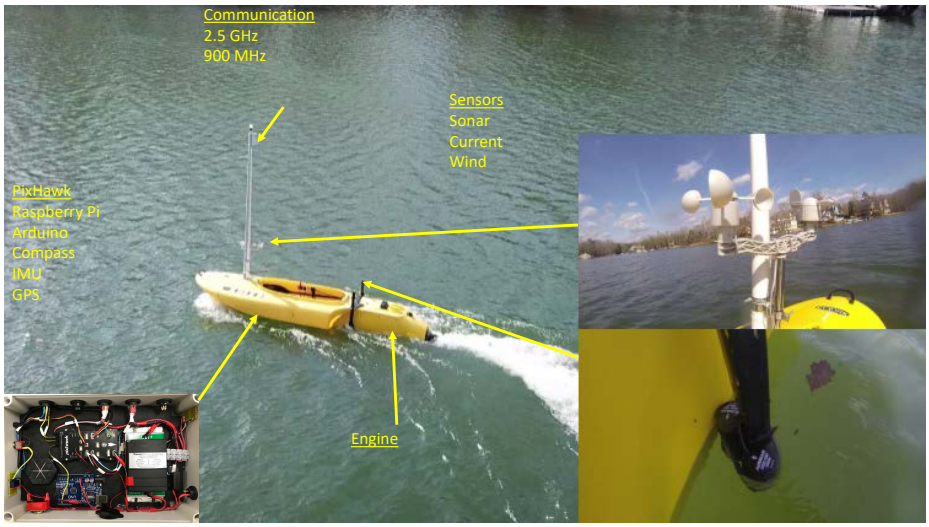


Figure 5.1: UofSC autonomous surface vehicle outfitted with anemometer, depth sonar, and current sensors.

assumed to operate largely undisturbed. Most fielded robots operate on fairly flat grounds, with minimal wind, wave, and current forces. Contrary to these scenarios, we consider robots that are best suited to operate in environments restrictive to humans. As such, capabilities to operate in unknown/dynamic environments, and in the presence of adverse external forces, are required to ensure that robots become ubiquitous and safe in many applications, such as safe inspection of infrastructure [52], search and rescue [44], environmental sampling [43], monitoring of water quality [39], and mapping inaccessible regions in more efficient, less cost prohibitive means.

In this work, using an Autonomous Surface Vehicle (ASV) (see Figure 5.1) we provide the following contributions: a reliable inexpensive platform for collecting depth, wind, and current data in different environments and conditions; and a data processing and model derivation approach for spatially varying environments.

5.1.1 Motivation

Currently, as observed in a variety of experiments – see Figures 5.2 and 5.5 for some examples – an ASV relying on a conventional robotic actuator controllers for way-point navigation is unable to maintain its course when faced with non trivial external

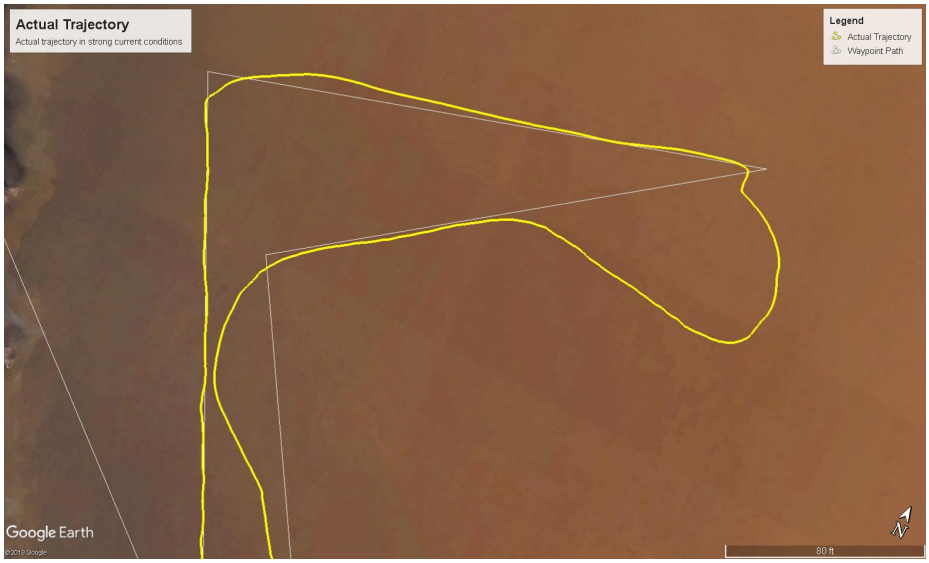


Figure 5.2: ASV unable to maintain course in heavy current when turning in downstream directions. Predominant downstream direction in this figure is from top to bottom. The white line represents ideal path and the yellow line the actual (GPS) trajectory of the ASV.

forces such as heavy wind or current. Due to the PID controller being tuned for conditions which are stable, there is no simple method to provide tuned coefficients for dynamic environments where wind and currents are always changing. This drives our motivation to pro-actively model, plan for, and adapt to these dynamics so the robot can maintain its course and not miss large swaths of its planned trajectory.

5.2 System and Methodology

In this section, we present the proposed hardware setup, calibration, and verification of inexpensive sensors in a controlled environment; experimental environments; and data collection and processing required to develop comprehensive models of external forces. We break this into two components, the physical and technical characteristics associated with the design and build of our ASV, and the data gathering/processing approach we take to model depth, current, and wind. The modeling process takes into account wind and current measurements, the ASVs trajectory from GPS, compass, and IMU data, together with the morphology and bathymetry of the environment.

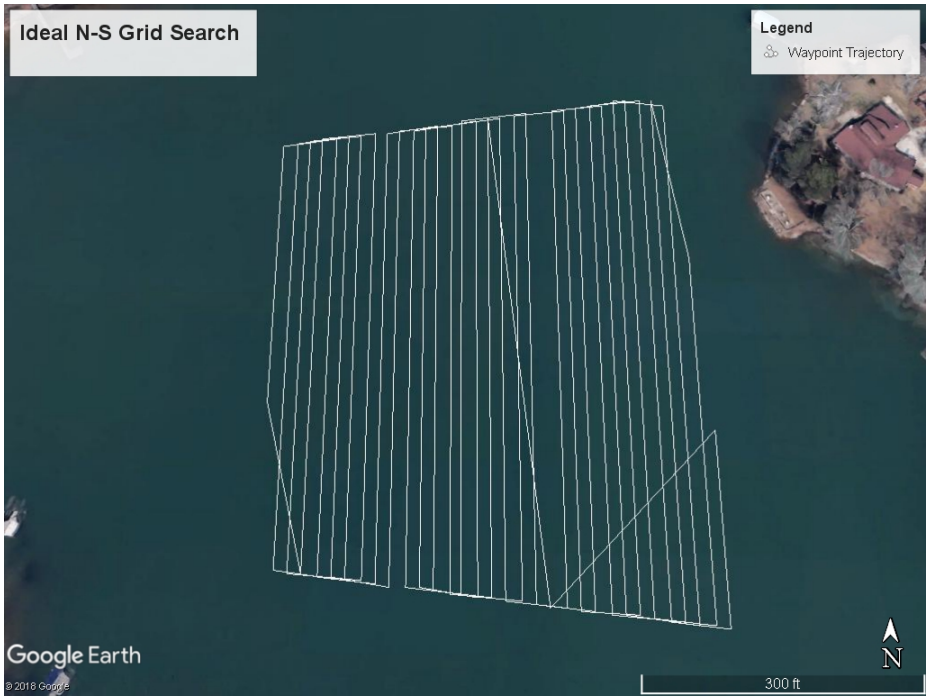


Figure 5.3: Ideal North-South grid search missions are used to provide a baseline for measuring tracking performance.

Figure 5.4: Trajectories for grid searches conducted in calm conditions.

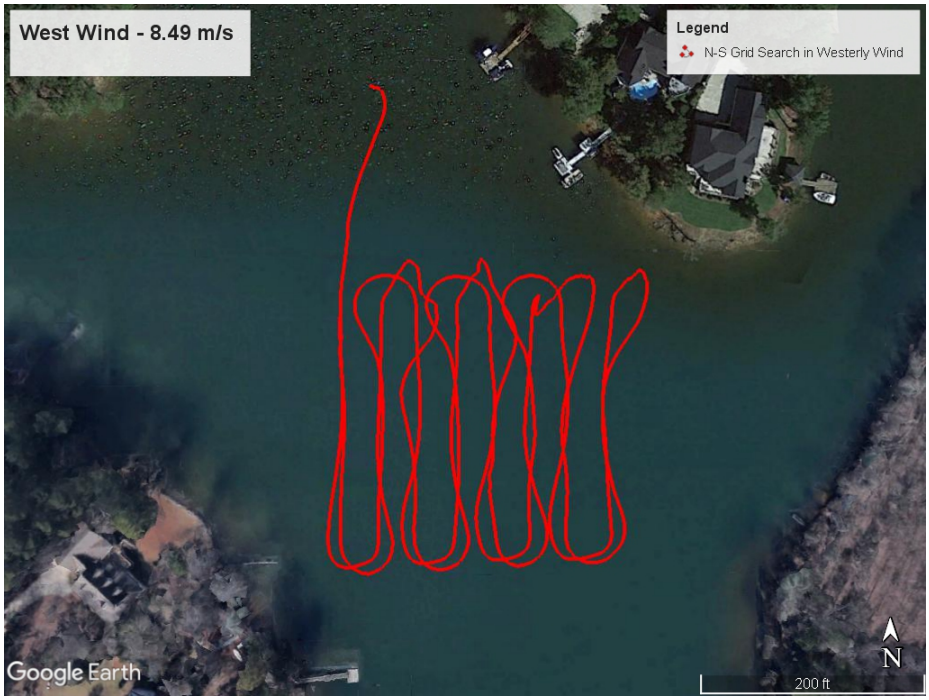


Figure 5.5: Trajectories recorded in 8.49 m/s wind illustrating the effects of the wind on the ASV.

5.2.1 Data Gathering and Processing

The wind and current sensors record data are influenced by the motion of the ASV. More specifically, the sensor measurement ($\overrightarrow{R}$) is the vector sum between the motion of the ASV ($\overrightarrow{A}$) and the real value of the physical phenomenon ($\overrightarrow{W}$). The motion of the ASV, $\overrightarrow{A}$ can be inferred by the GPS velocity and compass sensors. Therefore, the true value can be estimated as $\overrightarrow{W} = \overrightarrow{R} - \overrightarrow{A}$.

The current sensors are mounted in fixed locations and measure scalar current velocities (f). We select the component forces for ($\overrightarrow{F}$) by assigning the highest measured force to $F1$ and the highest of $F1's$ two neighboring sensor readings to $F2$. The forces are then offset by 45° to account for the angular velocities read by the sensor as the ASV ($\overrightarrow{A}$) traverses the current ($\overrightarrow{C}$). In our optimal setup, we have aligned four sensors, one offset at 45° on each quadrant of the boat, illustrated in Figure 4.8. This provides the equation $\overrightarrow{C} = \overrightarrow{F} - \overrightarrow{A}$ which we will solve using the

trigonometric properties resulting from our sensor array alignment.

Algorithm 1 Current Calculation

Input: $sensor_1, sensor_2, sensor_3, sensor_4$, *boat_heading*, *A_x*, *A_y*
Output: $\overrightarrow{C}$ (Surface current speed and direction)

$f1 \leftarrow \max(sensor_1, sensor_2, sensor_3, sensor_4)$.
if $f1$ is from starboard sensor **then**
    $f1 \leftarrow -f1$: *Reference frame is rotated +90 degrees*
end if
$f2 \leftarrow \max(neighbor\text{-}f1_1, neighbor\text{-}f1_2)$.
if $f2$ is from starboard sensor **then**
    $f2 \leftarrow -f2$: *Reference frame is rotated +90 degrees*
end if
$current\_magnitude\_boat \leftarrow \sqrt{f1^2 + f2^2}$
$\theta \leftarrow \text{atan2}(f1, f2)$: θ *is angle of current WRT*$f2$
$\phi \leftarrow \theta - 45°$: ϕ *is angle of current WRT boat heading*
$C_x \leftarrow current\_magnitude\_boat \cos(\phi)$
$C_y \leftarrow current\_magnitude\_boat \sin(\phi)$
$current\_magnitude\_world \leftarrow \sqrt{(A\_x - C_x)^2 + (A\_y - C_y)^2}$
$current\_direction\_world \leftarrow \phi - boat\_heading$

Once accurately aligned, the collected data were combined using a Gaussian Process (GP) mapping technique [62] to build a model of forces and depth. Initial modeling and correlation uses boat heading, GPS velocity of the boat, depth measurements, wind speed and direction measurements, and four current measurements as input. More formally, to calculate characteristics of a phenomenon $f(\mathbf{x})$, a GP can be used to estimate $f(\mathbf{W})$ at locations $\mathbf{W} = [\mathbf{w}^1, \mathbf{w}^2, \ldots \mathbf{w}^k]$ with a posterior distribution fitted over noisy measurements $\mathbf{Y} = [y^1, y^2, \ldots, y^n]$ collected by the robots at the corresponding GPS locations $\mathbf{X} = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n]$:

$$p(f(\mathbf{W}) \mid \mathbf{WX}, \mathbf{Y}) \sim \mathcal{N}(\mu_{\mathbf{W}}, \mathbf{\Sigma}_{\mathbf{W}}). \tag{5.1}$$

As typically done in the mainstream approach, assuming a zero-mean GP, the estimate of the phenomenon is given by the mean vector $\mu_{\mathbf{W}} = K(\mathbf{W}, \mathbf{X})\text{cov}(\mathbf{Y})^{-1}\mathbf{Y}$, where $\text{cov}(\mathbf{Y}) = K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I_q$ is the correlation between observed values and σ_n^2 is the noise affecting the measurements $\mathbf{Y}$. The covariance matrix is calculated

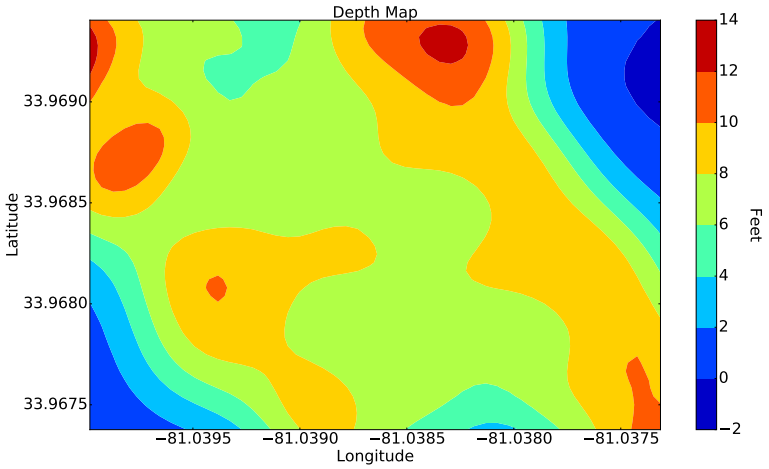


Figure 5.6: Depth map of Congaree River bottom resulting from GP.

as $\mathbf{\Sigma_W} = K(\mathbf{W}, \mathbf{W}) - K(\mathbf{W}, \mathbf{X})\text{cov}(\mathbf{Y})^{-1}K(\mathbf{W}, \mathbf{X})^T$. For accuracy evaluation, we tested different $\boldsymbol{K}()$ kernels. Kernels used for this initial comparison include linear, ExpQuad, Matern 3/2, and radial basis function (RBF), among which best performance was achieved by Matern 3/2, expressed by the following equation:

$$k(\mathbf{x}^i, \mathbf{x}^j|\Theta) = \sigma_n^2 \left(1 + \frac{\sqrt{3}r}{\sigma_l}\right) \exp\left(-\frac{\sqrt{3}r}{\sigma_l}\right), \tag{5.2}$$

where r is the Euclidean distance between $\mathbf{x}^i$ and $\mathbf{x}^j$, and σ_l is a positive parameter. Using the observations $\mathbf{X}$ and $\mathbf{Y}$ through the optimization of hyperparameters of the GP, predictions can be obtained.

5.3 Experiments

Experiments were carried out with a Jetyak equipped with the sensor suite described above in two different environments: a lake (Figure 5.3) in a 100 m × 100 m region, relatively calm; and a river (Figures 5.7 and 5.8), where the conditions are changing over time depending on rain and planned discharges by the local hydro-electric company. Our methodology uses standard grid-search patterns to establish a baseline for performance comparison.

We then process the collected data in two stages. First, we verify all timestamp and world orientation data by converting the PixHawk data logs to keyhole markup language (KML) for visual inspection of the missions, sequences and trajectories of the robots during the field trial. Once time stamps are verified to contain no gaps, we then process the ROS bag file by a Python script to align time stamps from each sensor, based on an approximate time synchronizer scheme[1]. It is at this point where we can verify the integrity of the data collected. Once complete, we are able to deem our system, scripts, calculations, and data as sound for further processing. The second stage processes the data for transformations, visualizations, and GP predictions as described in the previous section.

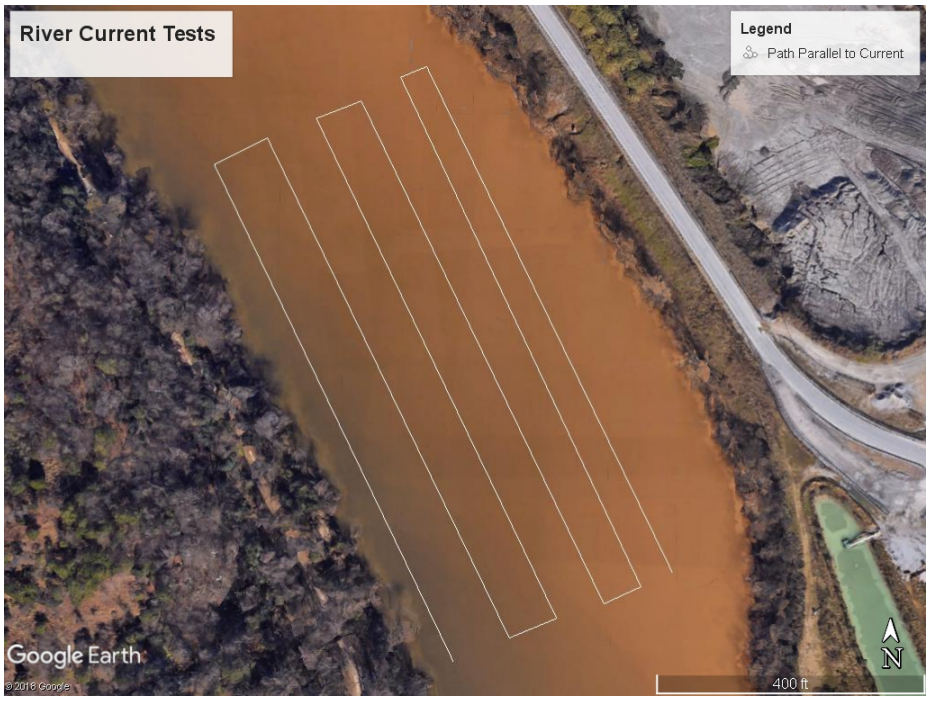


Figure 5.7: Baseline testing pattern parallel to the predominant current on the Congaree River, SC.

Some preliminary experiments were completed to fine tune the PID controller parameters and to assess the effects of taking the ASV, whose controller was tuned in

[1] `http://wiki.ros.org/message_filters/ApproximateTime`

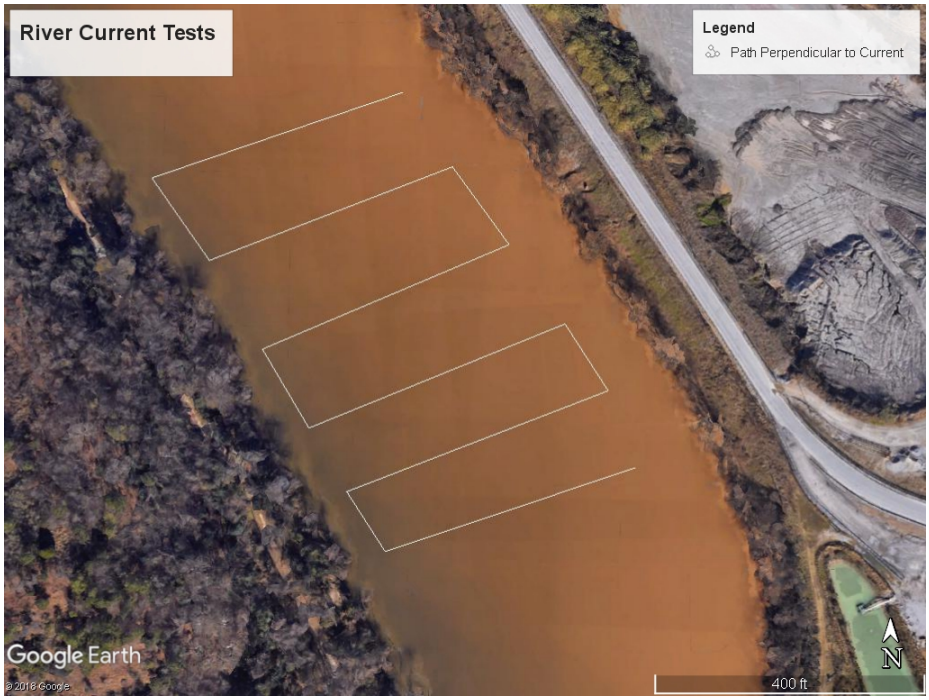


Figure 5.8: Baseline testing pattern perpendicular to the predominant current on the Congaree River, SC.

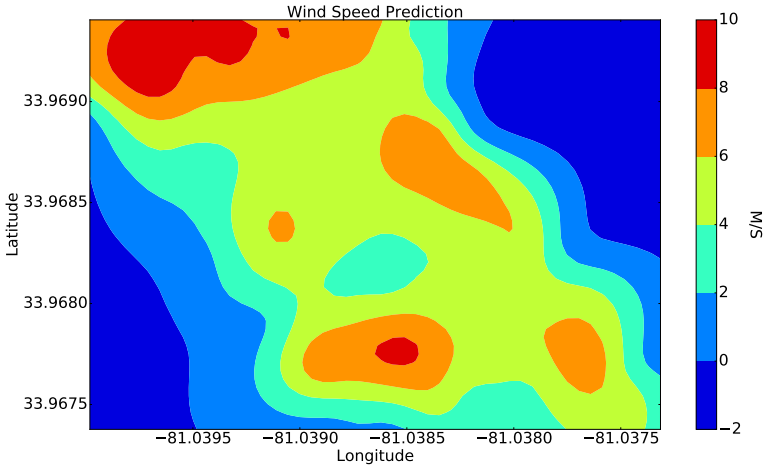


Figure 5.9: Wind Speed Prediction Map for Congaree River, SC.

the lake, to the river and evaluating its performance. As shown in Figure 5.5, clearly the ASV was not able to maintain the planned trajectory.

The main set of experiments included data collection under different conditions. In particular, first, we collected wind and current measurements by tying the ASV so

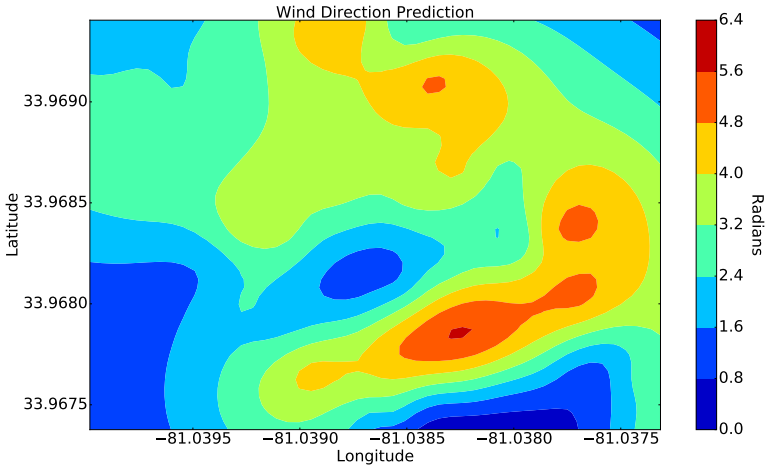


Figure 5.10: Wind Direction Prediction Map for Congaree River, SC.

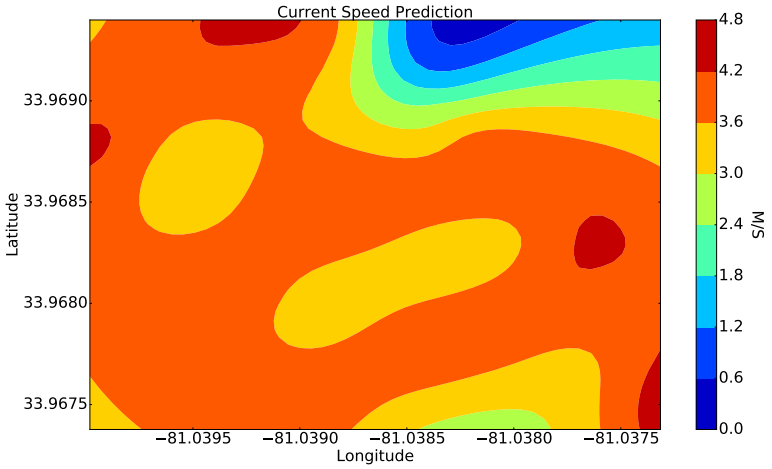


Figure 5.11: Current Speed Prediction Map for Congaree River, SC.

that is stationary. In this way, a baseline is available to compare with, when data is collected as the ASV is moving. Second, we planned different waypoint missions characterized by different patterns – i.e., parallel and perpendicular to current patterns – to collect data from different orientations.

Initial collected data in Figure 5.6 reflects the resulting topographical map of the riverbed after data collection using the coverage path in Figure 5.8. Wind and current data collected from the Congaree River clearly characterizes highly dynamic currents, changing winds, and a highly variant depth – see Figures 5.15 and 5.17

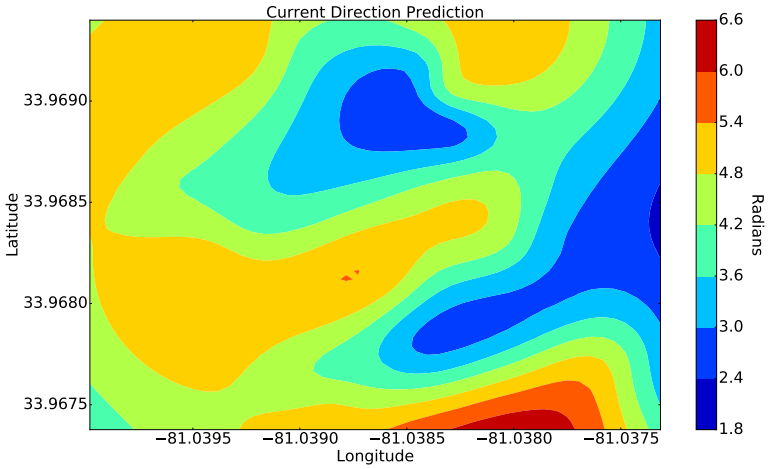


Figure 5.12: Current Direction Prediction Map for Congaree River, SC.

for recorded wind and current measurements; and Figures 5.9 through 5.12 for the predicted wind speed and direction, and current speed and direction, respectively, over the region. Figure 5.13 illustrates successful data collection and modeling of wind forces experienced in moderately windy conditions. Our system recorded sustained winds of 8.5 m/s and gusts reaching 12.5 m/s, accurate measurements according to the local weather recording station, which recorded sustained winds of 9 m/s and gusts at 14 m/s during the 3 hour trial. Given the mostly open terrain and the effects of the shoreline on wind patterns, our measurements are in-line with the recording station.

As illustrated in Figure 5.14, the average actual current measured while the ASV was navigating the cross sections of the river was 5.0 m/s, compared to the docked stationary current measurement of 3.0 m/s. The challenge with surface currents is establishing the ground truth due to the high degree of fluctuation with them. In addition to the previously mentioned lab calibrations, we ensure measurements are in the correct order of magnitude at the beginning of each field trial by recording the free floating GPS velocity. In this particular field trial, we approximated the ground truth current speed to be 2.5 m/s. Due to the hull displacement, as expected, we

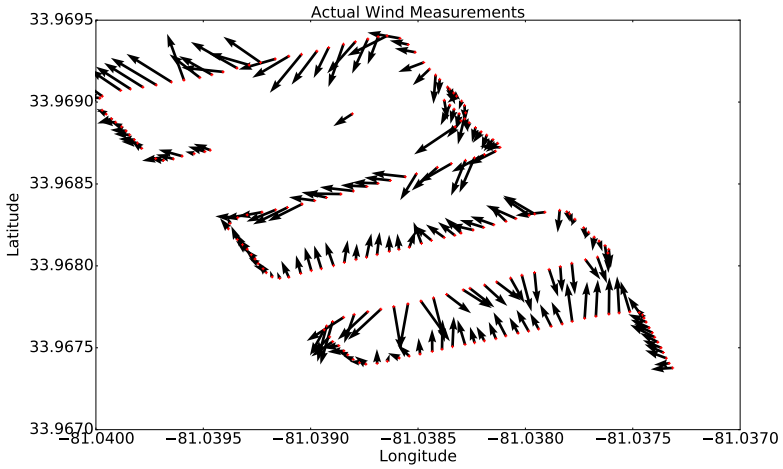


Figure 5.13: Recorded wind measurements for field trial conducted on Congaree River, SC.

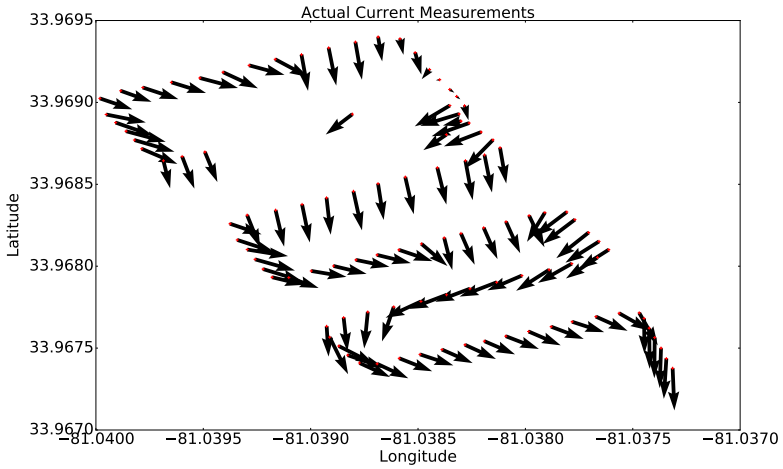


Figure 5.14: Recorded current measurements for field trial conducted on Congaree River, SC.

recorded a slightly lower velocity using this free float method.

5.4 Discussion

Our technical and experimental contributions in this chapter have produced a baseline of inexpensive tools and methods for recording the external forces – wind and current – acting on an ASV. GP regression modeling allowed us to predict for a large

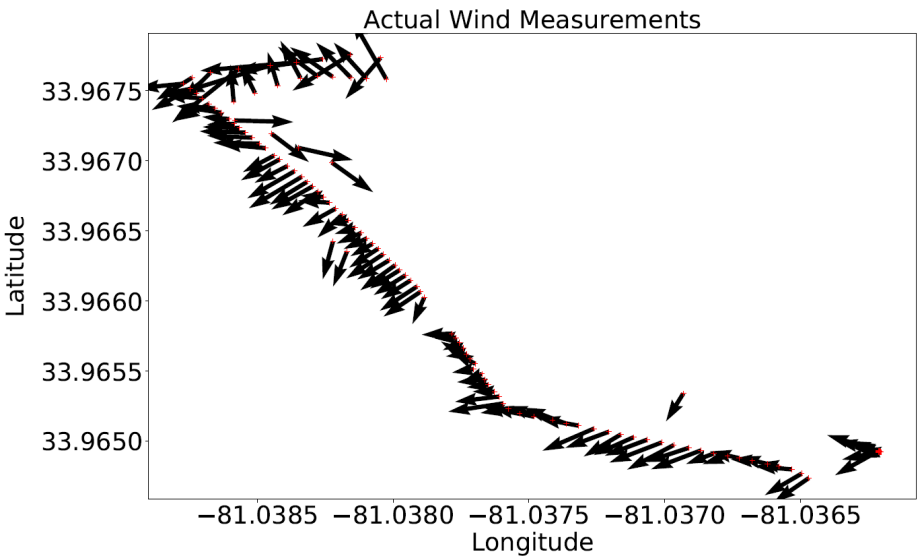


Figure 5.15: Actual wind direction and intensities on Congaree River after transformation from the boat reference frame to the world reference frame.

area with sparse measurements collected during the experiments, as shown in Figures 5.9 through 5.12. From the beginning of this endeavor, calibration of the analog current sensors to accurately reflect surface current required continuous refinement. Lab configuration and testing provided us an initial guideline on the best sensor configuration for a static vessel. Further on-free-float and static ASV testing confirmed our measurements to be reasonable for the conditions. Currently, this system is stable for recording and observing nature's phenomena in action.

Of particular interest in these experiments is the confirmation of the close correlation between the depth (Figure 5.6) and the affected current patterns in Figure 5.11. Verifying that our inexpensive system is able to collect and confirm US Geological Survey studies [15], encourages extending our research. As such, to improve the precision of predictions, our current work involves studying and analyzing different GP kernels to better emulate the short temporal livelihood of the predictions. The long-term goal is to combine all three models – i.e., current, depth, wind – into a comprehensive impact model that can be used to improve coverage and planning algorithms to enable ASV operation in highly dynamic environments.

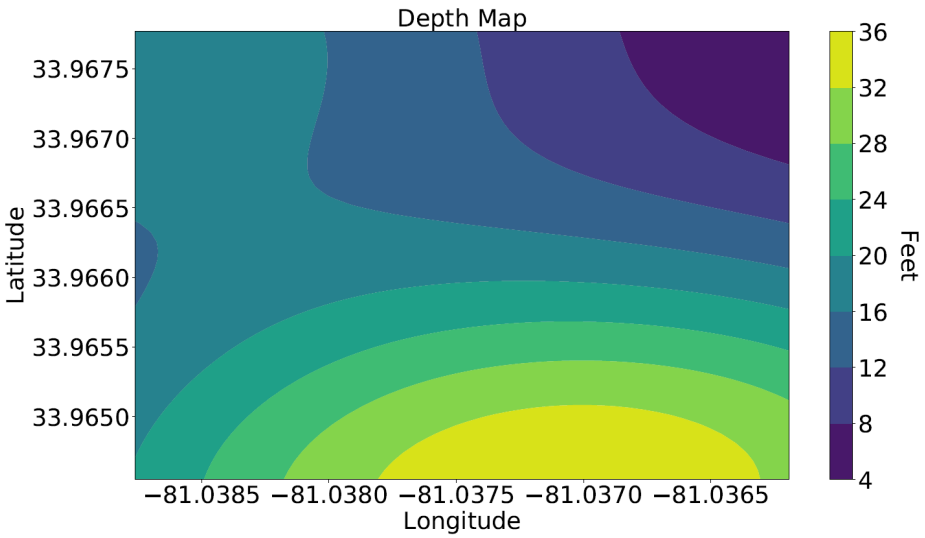


Figure 5.16: Depth map of 1 km portion of Congaree River, SC.

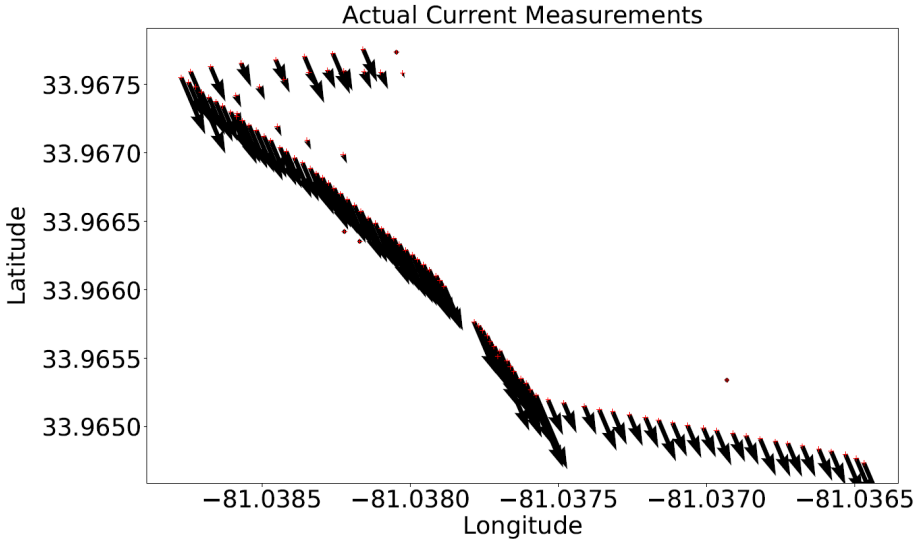


Figure 5.17: Actual current direction and intensities after transformation from boat reference frame to world reference frame taken on Congaree River during flood stage currents.

Chapter 6

EFFECTS MODELING AND PROACTIVE CONTROL IN DYNAMIC ENVIRONMENTS

In this chapter, we review the methods used to model the effects of environmental disturbances have on the ASV. Then we use this model with real-time measurements to resolve an improved dynamic control method.

6.1 Introduction

As robots become increasingly common, the demand for them to operate in environments where it is too dangerous for humans to go increases. In task-oriented missions where efficiency, power consumption, and precision are concerned, the ability to accurately maneuver becomes essential. While there exists much research into the effects of natural phenomena such as wind and current in ocean areas, there remains a void when it comes to studying the same type of effects on smaller ASVs in confined areas with higher currents such as rivers. In contrast to an aerial vehicle operating in windy conditions or an underwater vehicle operating in changing currents, marine surface vehicles must contend with forces from both domains. Small, lightweight ASVs operating areas such as lakes are affected by the wind, while ASVs operating in rivers can be affected by both wind and currents. It is fundamental to provide more acute control to enable the ASV to operate in these "non-nominal conditions".

In this work, we focus on the problem of modeling and counteracting the effects of non-trivial wind and current forces on ASVs. Robots operating on task-oriented

Figure 6.1: UofSC's custom-made ASV gathering wind data with anemometer and wind vane in windy conditions on Lake Murray, SC.

objective functions spend most of their resources deciding where to go next [11]. Typically, efficiency, cost, and risk are maximized karapetyan2018multior minimized in the objective function and, together, produce the next target trajectories [57]. From the target trajectories, maneuver is handed off to a low-level controller, such as Proportional, Integral, Derivative (PID) controllers, for execution [31].

Our research methodology derives an efficient wind and current effects model for use in our on-going work of a proactive controller. The goal is to improve ASV path following precision in dynamic environments. Modeling current and wind effects will allow a feed-forward loop working in tandem with a PID way-point navigation controller to correct course in real-time with event driven transitions between the two. There are four components required to achieve this goal. First, we required an ASV that has the durability, payload, sensing capability, and agility to operate effectively in a dynamic environment [48, 42], illustrated in Figure 5.1. Building on the capabilities demonstrated in Chapter 5, we are able to extend the mapping

capability to enable effects modeling and a proactive control method. The third requirement involves building a model of the effects of these adversarial forces on the ASV – the focus of this chapter. The final requirement is to implement a proactive controller to counteract these effects.

While there exists many open areas of research into modeling wind and currents on different scales for different oceanic-travel and power generation purposes, one relatively unresolved topic is enabling the more challenging scenario of small and agile robots operating in environments approaching their physical limitations. In this case we are focused on modeling the discrete effects of wind and current to enable real-time calculation that can be used then by a proactive trajectory following controller. We perform systematic experiments in several different scenarios, lakes and rivers, with different conditions, which allow us to extensively validate the proposed approach and discuss the insights from these field experiments to provide some directions for future work.

The next section shows related work on wind and current modeling, and it demonstrates the differences in our scenario with a small ASV. Section 6.2 formulates the problem we are addressing and describes the methodology for estimating ASV behavior given external force measurements and applying these behaviors as offsets in a feed forward controller. Section 6.3 presents our effect modeling experimental results, and we will discuss insights from our numerous experiments. Finally, in Section 6.4, we provide our observations, conclusions, and future work relating to proactive control in dynamic environments.

6.2 Methodology

To achieve our overall goal of providing a solution for controlling an ASV in dynamic conditions, we must first establish a baseline understanding of controllers. In our case the main issue is trajectory tracking. We define the tracking problem as minimizing

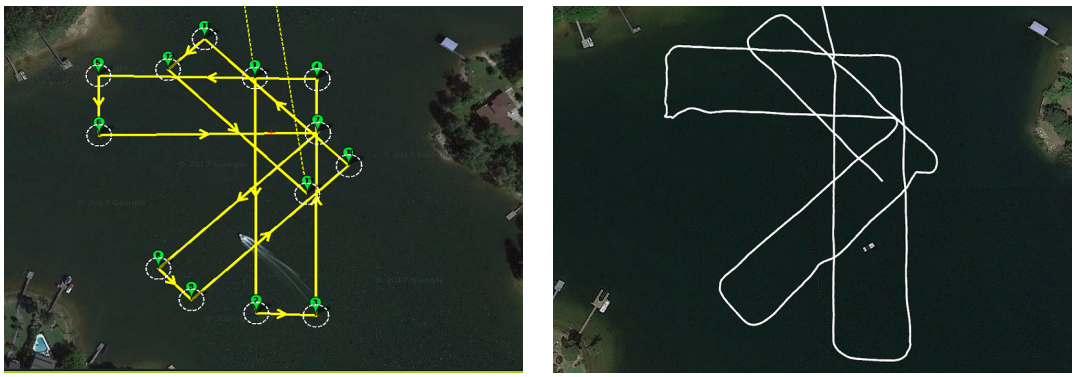


Figure 6.2: (a) One way-point mission with paths selected at 45 degree increments with respect to the predominant wind force. Target velocity for this iteration was 3 $\mathrm{m\,s^{-1}}$. (b) Actual path followed by the ASV on February 12, 2019 at Lake Murray, SC.

the distance between our specified target path and the actual path traversed. We derive the effects that external forces are having on our ASV by first collecting wind, current, and path following offsets in numerous conditions. Second, we model the effects based on the intended trajectory and the measured wind and current acting on the ASV.

Table 6.1: Weather Underground Wind Conditions - February 12, 2019

Time	**Wind Dir**	**Wind Speed**	**Wind Gust**	**Condition**
2:56 PM	SSW	7.60 m/s	11.62 m/s	Partly Cloudy
3:56 PM	SSW	8.05 m/s	14.31 m/s	Partly Cloudy
4:56 PM	SSW	7.15 m/s	10.79 m/s	Mostly Cloudy
5:56 PM	NW	14.75 m/s	24.14 m/s	Light Rain

Finally, the development of a feed-forward controller to augment the way-point navigation of the ASV will integrate this effects model to enable online course corrections as environmental forces are sensed. This method improves on similar methods covered in the related work section, where our feed-forward augmentation occurs when external force measurements are made while previous work focused on countering effects of the external phenomena after they are sensed by the IMU.

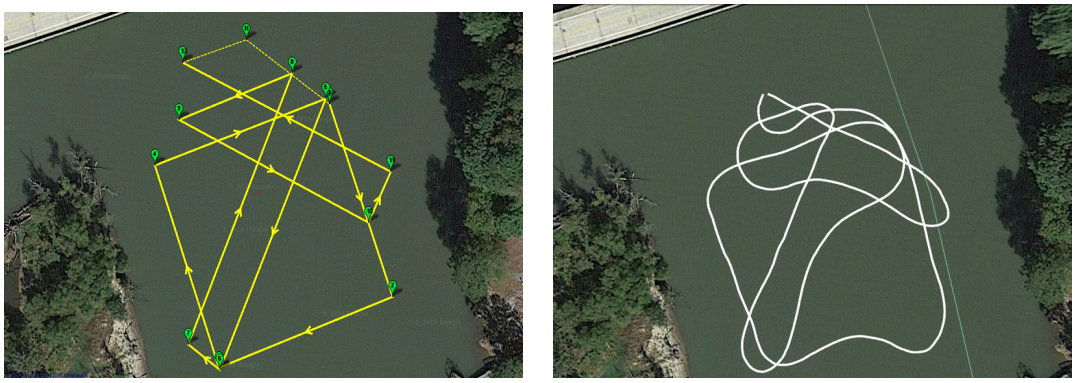


Figure 6.3: (a) A Way-point mission with paths selected at 45 degree increments with respect to the predominant current force. (b) Actual path followed by the ASV on February 25, 2019 at Saluda River, SC.

6.2.1 Data Collection Strategy

We conduct extensive calibration in non-disturbed conditions, data collection in several conditions, and compute experimental predictions to evaluate differing strategies to address them. As intuitively expected the stronger wind and current forces on the boat accentuate over-correction and oscillatory behaviors inherent to the PID controller, in this case resulting in the boat not being able to accurately maintain the target trajectory, illustrated in Figures 6.2(b) and 6.3(b).

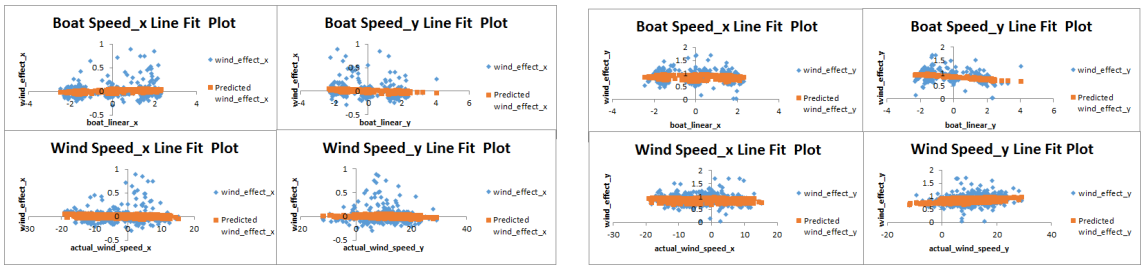


Figure 6.4: (a) 510 sample linear regression plot fitting ASV velocity and wind velocity to calculated x component of the wind effect vector. (b) Linear regression plot fitting ASV velocity and wind velocity to calculated y component of the wind effect vector.

For this, as well as future work in proactive controls, we developed a core testing pattern as a basis for behavior comparison reflected in 6.2(a) and 6.3(a). In order to enable high-fidelity bathymetric measurements, we seek to achieve straight line

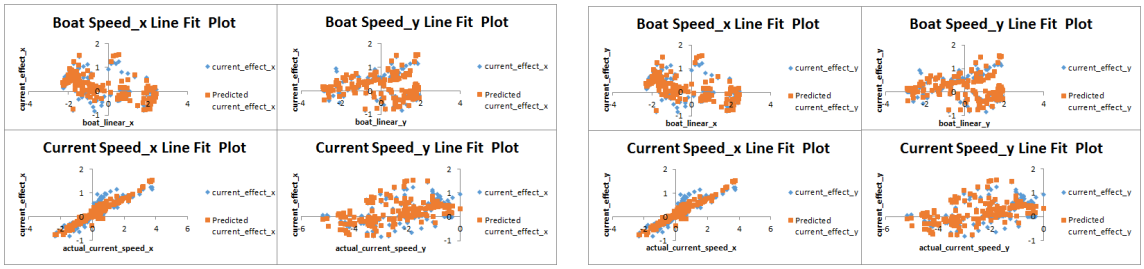


Figure 6.5: (a) 180 sample linear regression plot fitting ASV velocity and current velocity to calculated x component of the current effect vector. (b) Linear regression plot fitting ASV velocity and current velocity to calculated y component of the current effect vector.

trajectories. Each of the headings were 45° increments beginning with heading in the predominant direction of the force, while the target speed was also predefined for each iteration, allowing for comprehensive effects to be measured. The ground truth wind speeds observed during the period of the wind trial conducted on February 12, 2019 on Lake Murray[1] were South-Southwest sustained winds of $14\,\mathrm{m\,s^{-1}}$ with gusts over $24\,\mathrm{m\,s^{-1}}$. These measurements validate our collected measurements in Figure 6.7(a). While ground truth is difficult to establish in currents, we were able to measure the average current from a stationary position to be $4\,\mathrm{m\,s^{-1}}$ during the trial with our measurements presented in Figure 6.7(b).

6.2.2 Modeling the Effects

From Chapter 5, we focus on modeling the effects of the external disturbances in our close vicinity rather that over the larger area of interest. We model the effects of these disturbances on the ASV, where the inputs are the ASV vector velocity $\overrightarrow{A}$ and the phenomena force vector velocity $\overrightarrow{P}$ expressed in their component forms, A_x, A_y, P_x, P_y, respectively. The output we model is the effects vector velocity $\overrightarrow{E}$, expressed in its component form, E_x, E_y. We express our output with the following

[1] `https://www.wunderground.com/history/daily/us/sc/irmo/KCAE/date/2019-2-21?cm_ven=localwx_history`

equation:

$$\overrightarrow{E} = \overrightarrow{A} - \overrightarrow{P} \tag{6.1}$$

We adopt a systematic methodology for extensive data collection and verification to increase precision in discrete force modeling. The culmination of over 20 field trials garnered over 75,000 data points collected over 100 kilometers of linear distance in diverse lakes and river areas to support this research area.

Throughout numerous field trials, we realized the intuitively linear cause and effect nature of the wind and current with the associated behaviors of the ASV. Coupled with the amount of raw data collected, we chose to create our model of the observed effects using linear regression. Using standard techniques to train and test our data at the 80/20 ratio, we derived the linear regression plots illustrated in Figures 6.4- 6.5. The resulting RMSE and Variance Score from Scikit-learn [54] for the regressions are listed in Table 6.2. Most outliers on these charts are attributed to the points where the ASV is turning according to the predefined mission. In these small transitions the heading of the ASV and the propagation time for the sensor measurements to be calculated and recorded become momentarily out of synchronization.

Table 6.2: RMSE and Variance Scores for Linear Regression of Boat Velocity and Wind/Current Velocity and their respective predictions

	RMSE	**Variance**
Wind Effect x	0.26	0.02
Wind Effect y	1.72	0.08
Current Effect x	0.37	0.09
Current Effect y	0.48	0.21

6.3 Effect Modeling Experimental Results

In order to provide results that are readily intuitive, we created experimental test trajectories that deliberately create edges along the cardinal and inter-cardinal direc-

tions with respect to the wind or current phenomena of which we are seeking to learn their effects.

The following results reflect the capability for our Jetyak and its accompanying sensor package to measure and provide real-time environmental dynamics for an on-board controller to immediately counter-measure dynamic disturbances.

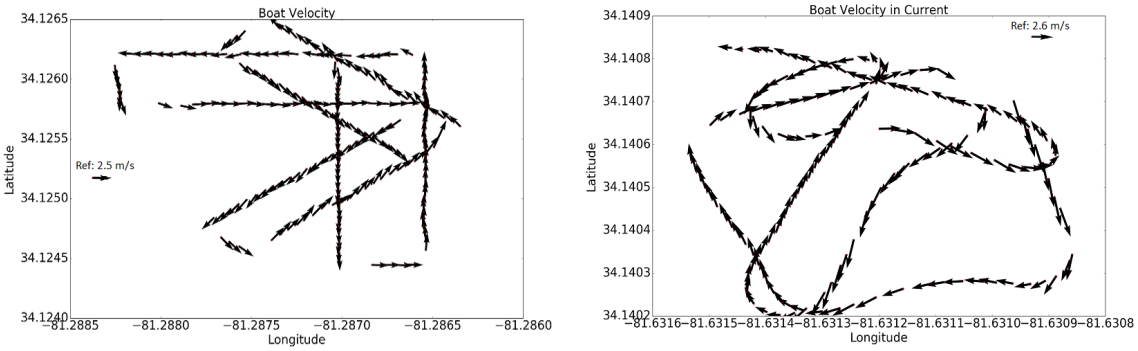


Figure 6.6: (a) Measured ASV velocities, Lake Murray, SC, USA. (b) Measured ASV velocities, Saluda River, SC, USA.

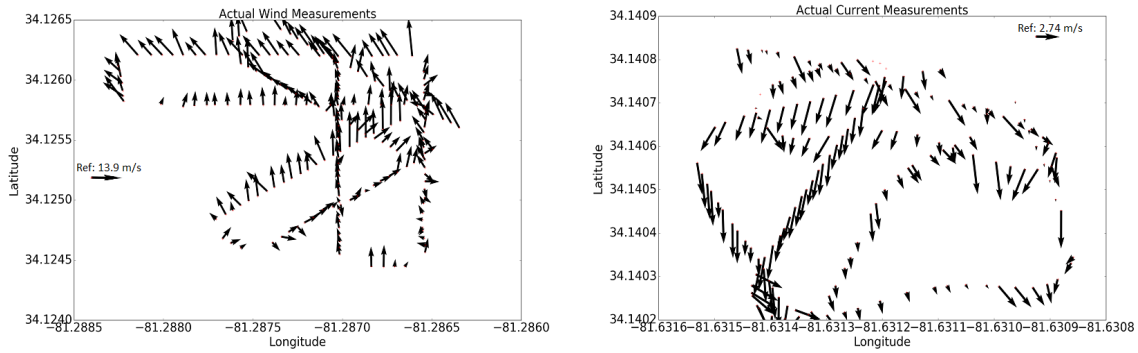


Figure 6.7: (a) Calculated wind velocities (0–13 $\mathrm{m\,s^{-1}}$) from ASV on-board anemometer and wind vane. (b) Calculated current velocities (0–4 $\mathrm{m\,s^{-1}}$) from ASV on-board Hall-effect current sensor array.

Table 6.3 provides a consolidated reference of the average ASV velocities, wind/current velocities, and the effects of each on the ASV for the field trials presented in this chapter. Note the effect that substantially high winds have on the ASV are lower than the effects of a moderately flowing current. These effects must be weighted accordingly in the feed-forward controller presented in future work.

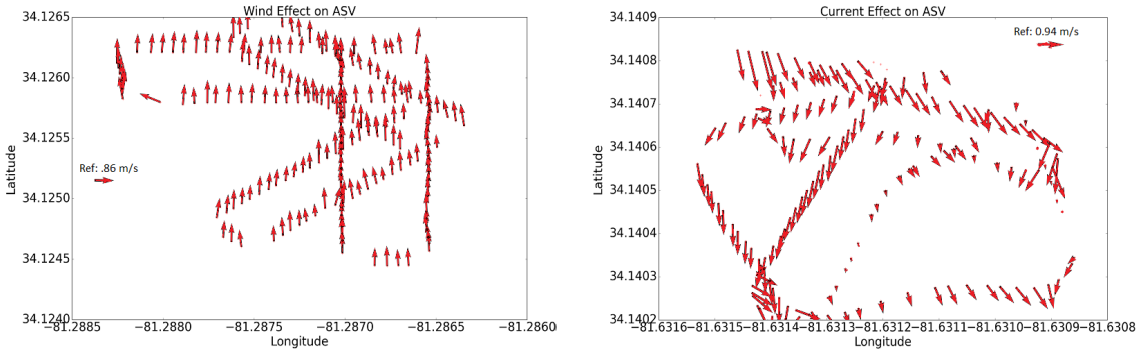


Figure 6.8: (a) The effects of the wind on the ASV. (b) The effects of the current on the ASV.

The results presented here represent one of over 40 test missions run during the data collection for the effects modelling and control development portion of this thesis. Even within the duration of a single field trial, conditions can change abruptly, like when water control systems for hydro-electric dams are opened without notice. One last observation and opportunity for future work results from several of our field trial deployments occurring during flood stages. What happens when the current velocity is too high for the ASV to overcome? Are there strategies to overcome this? The answer will complete the work and contributions in this thesis.

Table 6.3: Average velocities for lake and river trials presented.

	Lake Mean Speed	**River Mean Speed**
ASV	2.14 m/s	2.05 m/s
Wind/Current	13.9 m/s	2.75 m/s
Effect	0.85 m/s	0.94 m/s

6.4 Effects Modeling Conclusion

As shown through systematic experimentation and results, our method for correlating environmental measurements to ASV behavior illustrates the requirement for PID controller augmentation to overcome imprecise path following in dynamic winds and currents.

Contributions of this section include the experimental development of a model capable of accounting for the effects of temporally dynamic phenomena, such as wind and current, acting on a free-body system, such as a boat. In our case, it is readily discernible from the presented material, that the bulk of efforts should be placed on refining controls in highly disturbed currents, due to their dominant effects on a small ASV's trajectory.

Future work specific to small and lightweight ASVs includes completing a robust controller hand-off between way-point navigation and a feed-forward dynamic environment navigation controller, where a threshold for a measured external force determines the transition between the two routines, as shown in Figure 6.9.

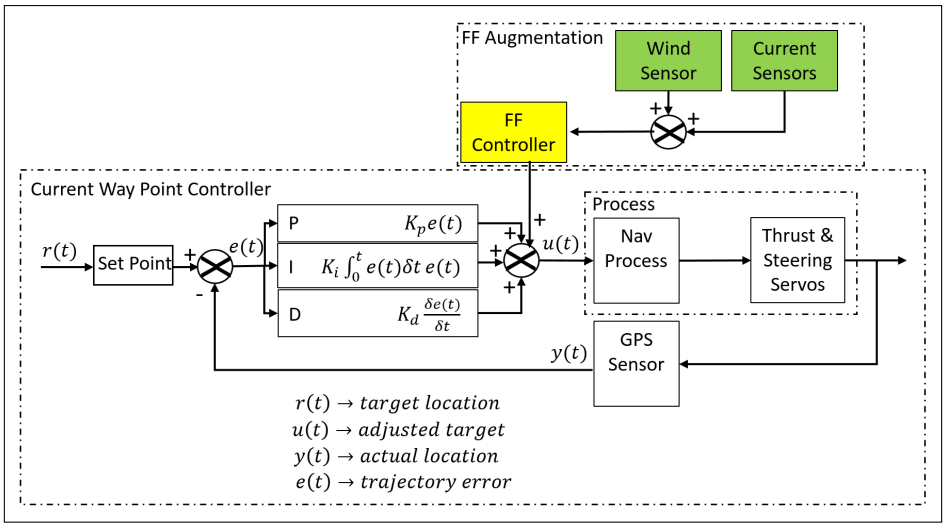


Figure 6.9: Increment 1 for ongoing work to integrate the output of the effects model into a robust controller capable of maintaining its planned trajectory when the PID controller from a way-point navigator cannot. White boxes indicate existing onboard way-point controller; green identifies the sensor group enabling this chapter and yellow identifies our ongoing initiative.

In the following sections of this chapter, we will introduce increment 2 of our augmented proactive controller design. From this implementation future generations can be improved.

6.5 Augmented Control Introduction

In this work, we present novel and inexpensive methods for sensing these external forces, together with methods for accurately controlling an ASV in the presence of such external forces. The resulting platform is capable of deploying bathymetric and water quality monitoring sensors. Experimental results in the local lakes and rivers demonstrate the feasibility of the proposed approach. In addition, this ASV is capable of hosting added payload for protective equipment when deploying in areas deemed too dangerous for human exposure.

As the demand for data collection and monitoring continues to expand across all reaches of the globe, research and development of Autonomous Surface Vehicles (ASV) control in uncertain environments is essential. While the tasks and missions to which an ASV could be assigned are only limited by one's imagination, our desire to explore the unexplored increases the capabilities required in an ASV. One such hypothetical employment for an ASV would have been to assist with monitoring and recovery after the Fukushima Daiichi nuclear disaster following the 2011 earthquake in Japan (Figure 6.10).

For a less catastrophic scenario, with over 3.5 million miles of rivers in the United States alone, the ability to access, cover, and navigate them requires an ASV with long range potential, as well as a precise trajectory following capability to ensure safe maneuvers. In addition, the ability to take into account the effect of external forces would improve the efficiency in planning for coverage as well as savings in power and fuel consumption. While there exists much research into the effects of natural phenomena such as wind and current in ocean areas, there remains a void when it comes to studying the same type of effects on smaller ASVs in confined areas with higher currents such as rivers. Operating in the air and water domains simultaneously exposes ASVs to wind and current external forces that can easily overwhelm current Proportional, Integral, Derivative (PID) controlled navigation systems.

Figure 6.10: Fukushima Daiichi nuclear complex in Okumamachi, Japan prior to earthquake disaster of 2011.

This chapter pushes the research boundaries to advance the state of the art which will allow ASVs to be utilized in increasingly challenging conditions to ensure that ASVs become ubiquitous with researchers, engineers and environmental scientists.

6.5.1 Problem Definition

Addressing the challenge of operating in the presence of non-trivial external forces can be done in two different scales. If a long-range map of the external forces is available then large scale planning can take the effects of the external forces into account. For example, coverage planning algorithms [29, 30], can include the force map as an input variable in order to improve mission planning. In a smaller scale, real-time force measurements can be used in a reactive controller to accurately track the desired trajectory. In analogy, knowing the traffic patterns in a city can generate routes through less congested streets, while a driver sensing slipping on ice, or pushed by a wind gust can guide the vehicle accordingly. In this section, we provide novel

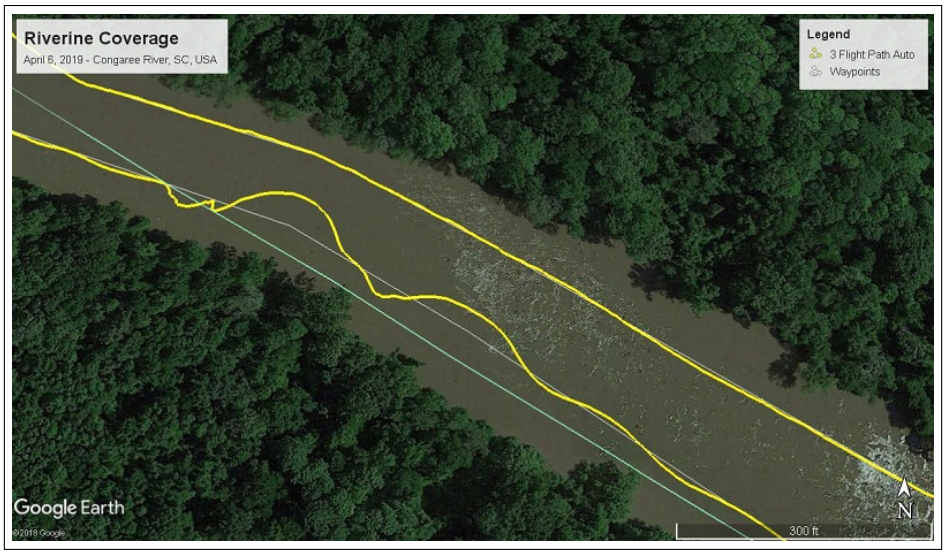


Figure 6.11: Target trajectory unable to be followed in the downstream pass due to high 3.0 m/s currents in the area.

methods for augmenting a controller with information from local disturbances. A manifestation of this problem is illustrated in Figure 6.11, where the ASV is unable to maintain an accurate trajectory due to the PID controller being overcome by the changing currents.

Another front of this challenge is to adapt real-time to ever changing dynamics so that the ASV can maintain accurate path following to collect the required data at the desired locations. In this work, we present a novel approach to inexpensively solve the second issue.

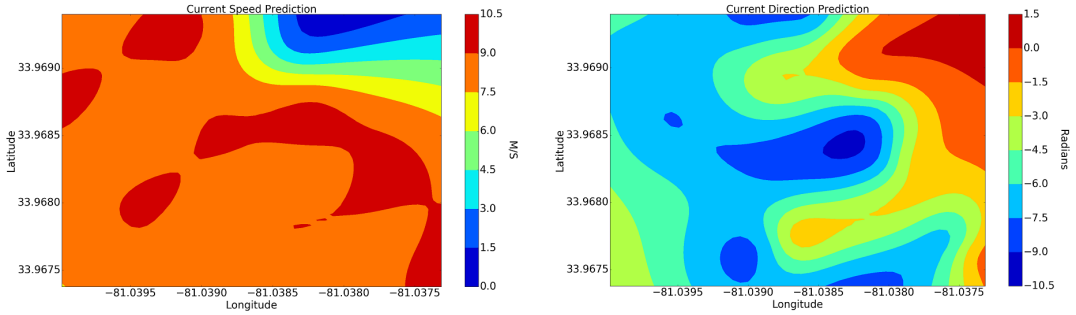


Figure 6.12: (a) Current speed prediction, (b) Current direction prediction during flood stage on Congaree River, SC..

In most riverine environments, it should be noted that a standard PID driven

way-point navigation controller can be tuned to maintain course either when moving with or against the external force, but not both conditions with the same gains. See for example Figure 6.11, where the trajectory is followed accurately upstream, meaning against the current, and the erratic trajectory is produced from a downstream path. The contribution of this work is the augmentation of the ASV's current control

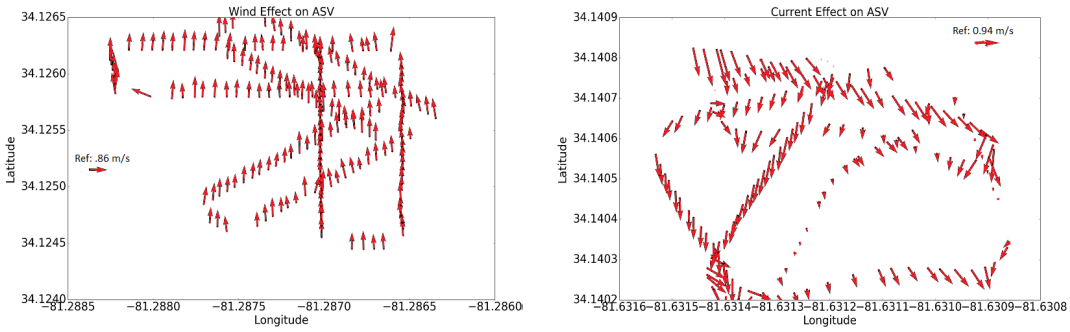


Figure 6.13: The effects of the wind and current on the ASV. Illustration reflects different scales due to the dominant effect of current over wind on the ASV.

system with feed-forward controls to overcome the external dynamics and maintain a more accurate trajectory, by using measurements and models of natural disturbances affecting an ASV proposed in our previous work. Such a contribution will provide the greater scientific community with a more precise platform for data collection in challenging environments. In addition, it can provide an efficient and robust tool to aid search and rescue operators as well as environmental monitoring and bridge inspection teams.

The following section presents the methodology for accomplishing our goal as inexpensively as possible. A brief discussion on the effects of external forces acting on an ASV followed by a detailed proactive control augmentation description. Section 6.7 presents our experimental setup and approach to create a field trial testing environment to produce meaningful results in Section 6.8. Finally, we conclude with a short discussion of the results and suggestions for future work in this area.

6.6 Methodology

This section describes the strategy we employ to solve the problem presented in Section 6.5.1. For completeness, we will first briefly review the method for measuring external forces and modeling their effects on an ASV; for more information please see Moulton *et al.* [49]. Then we present our approach of using these effects to implement proactive path-following control augmentation.

6.6.1 External Force Effects

There are two overlapping areas that benefit from measuring the external forces acting on the ASV. The first area, addressed in our prior work, is the ability to create a high-level force map of a given phenomenon (see Figure 6.12). This capability enables planning algorithms such as the one proposed by Lewis [38] Karapetyan *et al.* [29] to be extended to pre-select deployment sites and plan more efficient coverage solutions prior to launch. The second benefit results from the ability to use machine learning techniques for regression to produce effects models for the impact external forces are having on the robot. This capability enables the work presented in this section, which in high-level terms, the modeling of the effects feeds an adaptive controller which counteracts the external forces allowing for more accurate trajectory following of the ASV.

6.6.2 Proactive Control Through Way-point Augmentation

Given an accurate model of the environment dynamics and an ability to predict temporally close external forces and their effects on the ASV, we seek to provide an augmentation to the Pixhawk way-point navigation controller. By manipulating the target global pose based on the measurements and effects of external forces we are able to provide intermediate way-points to the Pixhawk, coercing it to maintain the original desired trajectory; see Figure 6.14. The intermediate way-points account

for the effects of external forces and are calculated proportional to the distance d_t between the ASV and the goal way-point.

Pos_t is composed of the ASV's latitude, longitude, and velocity. F_t is comprised of the expected effect on the ASV's speed and heading resulting from the effects models in Section 6.6.1. Pos_n is the goal way-point and d_t is the distance between the ASV and Pos_n. Target'_n is a calculated intermediate way-point to send the controller to maintain the desired trajectory.

This portion of feed-forward augmented controller is illustrated in Figure 6.15. The algorithm used to calculate the intermediate target way-points is presented in Algorithm 2.

The inputs to the algorithm are:

- The measured current speed magnitude spd_c and direction dir_c,
- The measured wind speed magnitude spd_w and direction dir_w,
- The ASV position (lat_t, long_t),
- The ASV speed spd_t and heading h_t,
- The target ASV speed spd_target,
- The list of way-points in the current mission.

The measurements are processed as they are received from the sensors during execution of each way-point from the mission. Based on the speed and orientation of the ASV we determine the absolute values of each measurement and use that to predict with a linear regression the effect of the force on the speed and direction of the ASV (Line $7-8$) [51]. While the target way-point is not reached, an intermediate way-point is calculated based on the effect_x and effect_y values . The speed is also adjusted based on the predicted error (Line 10). Finally, the ASV is sent to the newly calculated way-point (Line 11). When the new target position is processed by the

Pixhawk navigation controller, it results in a smoother and more accurate path, and with this we realize our original intended trajectory. In the following section, we will present the experiments carried out to demonstrate this capability.

Algorithm 2 Feed-forward Augmented Way-Point Navigation Controller

Input: spd_c, dir_c, spd_w, dir_w, $(lat_t, long_t)$, spd_t, h_t, way-point list $(lat_n, long_n)$, spd_target

Output: None

```
mission ← wp_list(lat_n, long_n)
count ← |mission|
for each i ∈ 1, ..., count do
    go_to_waypoint(lat_i, long_i, spd_target)
    wp ← lat_i, long_i
    while wp is not reached do
        effect_spd, effect_dir ← effect_model(
    spd_c, dir_c, spd_w, dir_w, spd_target, spd_t, h_t)
        effect_x, effect_y ← convert_to_coordinate_vectors(
    effect_spd, effect_dir)
        lat'_i, long'_i ← calc_intermediate_wp(lat_i, long_i, effect_x, effect_y)
        spd'_i = effect_spd + spd_target
        go_to_waypoint(lat'_i, long'_i, spd'_i)
    end while
end for
```

6.7 Experiments

Over ten deployments were completed in support of this initiative, collecting and testing in over 190 km of river and lake environments; for testing the proposed controller, four of the deployments were in the river testing the control, while the rest established a baseline behavior and tested the effect of wind and accuracy of current measurements.

6.7.1 Experimental Approach

In order to provide experimental results that are easily comparable to the original way-point PID controller, we use straight line test trajectories that run in the cardinal

directions parallel and perpendicular to the predominant external force. In this case, currents are being tested, and we illustrated in Figure 6.11 that the Jetyak's poorest path following performance occurs when traveling in the same direction as the current. This led us to select the four cardinal and four intermediate direction orientations to the current as our test baseline, shown in Figure 6.16. Straight line segments were produced to replicate the most common patterns from route planning experiments. The generated segments were initially used as input to the standard Pixhawk way-point controller. Then, the same segments were used as input to the augmented controller with the intermediate way-points enabled.

Given this controlled experimental setup, the results in Section 3.5 illustrate the success of this approach as well as directions for future work.

6.8 Proactive Control Results

In this section we will compare the performance of the standard Pixhawk GPS way-point navigation controller with and without the proposed feed-forward augmentation. Since the standard controller performs well in upstream maneuvers, the focus will be on the performance difference in the downstream cases. Due to weather constraints during the field trials, the results presented in this section of the thesis were obtained from data collected in a river with an average measured speed of 0.677 m/s during the trial for straight trajectories.

6.8.1 Way-point Navigation

As illustrated in Figure 6.17, the built-in Pixhawk controller is generally able to reach the required way-points. However, the PID coefficients are tuned to operate in a specific environment. When changing environments, the PID coefficients should be tuned again. This task becomes insurmountable when operating in environments with ever-changing dynamic forces at play. As shown in Figure 6.17 left, negotiating

currents in upstream to perpendicular directions is relatively stable. This is due to the fact that the speed of the ASV relative to the ground is slightly reduced, allowing enough time for the PID controller to compensate for the error. However, in Figure 6.17 right, we see the opposite effect when the speed of the ASV relative to the ground is increased, thereby accumulating too much error in the PID controller to overcome the external forces. This typically results in an overshoot scenario where the ASV begins harmonically oscillating back and forth over the desired trajectory. It should also be noted, that as the speed of the current increases, this behavior starts to present itself in trajectories perpendicular to the current. Adjusting the integral gain in the PID controller can help solve this problem, but it will also produce undesirable oscillatory behavior in upstream trajectories.

6.8.2 Proactive Effects Augmented Way-point Navigation Controller

As illustrated in Figure 6.18, by augmenting the built-in PID controller in the Pixhawk, we were able to follow much more precisely the desired path to each way-point than the non-augmented controller. These results serve as proof of concept for Algorithm 2. As shown in Figure 6.18, path following in currents in all orientations to the ASV is qualitatively improved.

The results in Table 6.4 show a quantitative comparison of the performance of our augmented proactive controller with the baseline way-point navigator. In particular, a marked improvement can be observed in both maximum error and percentage of the path that is more than a meter far from the target trajectory. Max error represents the largest distance between the straight-line trajectory and the actual path of the ASV. Percentage path error greater than one meter quantifies the portion of the path where the ASV was more than one meter from the ideal trajectory. Confirming intuition, the ability of the augmented control algorithm to change the forward thrust of the ASV provides the largest numerical improvement when moving with the predominant

Table 6.4: Comparing the performance of the standard Pixhawk way-point PID controller with our intermediate way-point augmented control.

		ASV Trajectory Relative to Current						
		Perpendicular	Parallel With	Parallel Against	L-R Diagonal With	L-R Diagonal Against	R-L Diagonal With	R-L Diagonal Against
WP Navigator with PID Control	Max Error	4.50 m	9.32 m	3.86 m	3.46 m	1.63 m	7.85 m	2.57 m
	% Path Error >1 m	44.2%	76.8%	18.3%	48.6%	12.7%	83.5%	40.1%
Augmented WP Navigator with PID Control	Max Error	1.58 m	1.48 m	1.08 m	0.75 m	0.74 m	1.07 m	0.68 m
	% Path Error >1m	9.3%	11.9%	7.9%	0%	0%	6.3%	0%

direction of the current.

6.9 Conclusions

The path-following precision achieved by this work can have profound impacts for the research, emergency services, and exploration communities. The ability to provide bathymetric surveying and mapping capabilities to remote areas with highly dynamic currents will enable researchers to expand the boundary between known and unknown environments.

To improve the robustness of the control augmentation presented, future work should include two areas. First, the addition of providing the same precision path following for a Dubins vehicle, such as the Jetyak will require additional methods to handle deliberate turns in planned missions. Second, another desirable expansion of this work will include changing from intermediate way-point augmentation to a lower level control of the linear and angular velocities (v, ω). Such an approach may produce more concise countermeasures to further reduce the path tracking error.

Given the proof-of-concept and the validation of our implementation in the chap-

ter, we will move to the conclusion and present some closing thoughts for this venture.

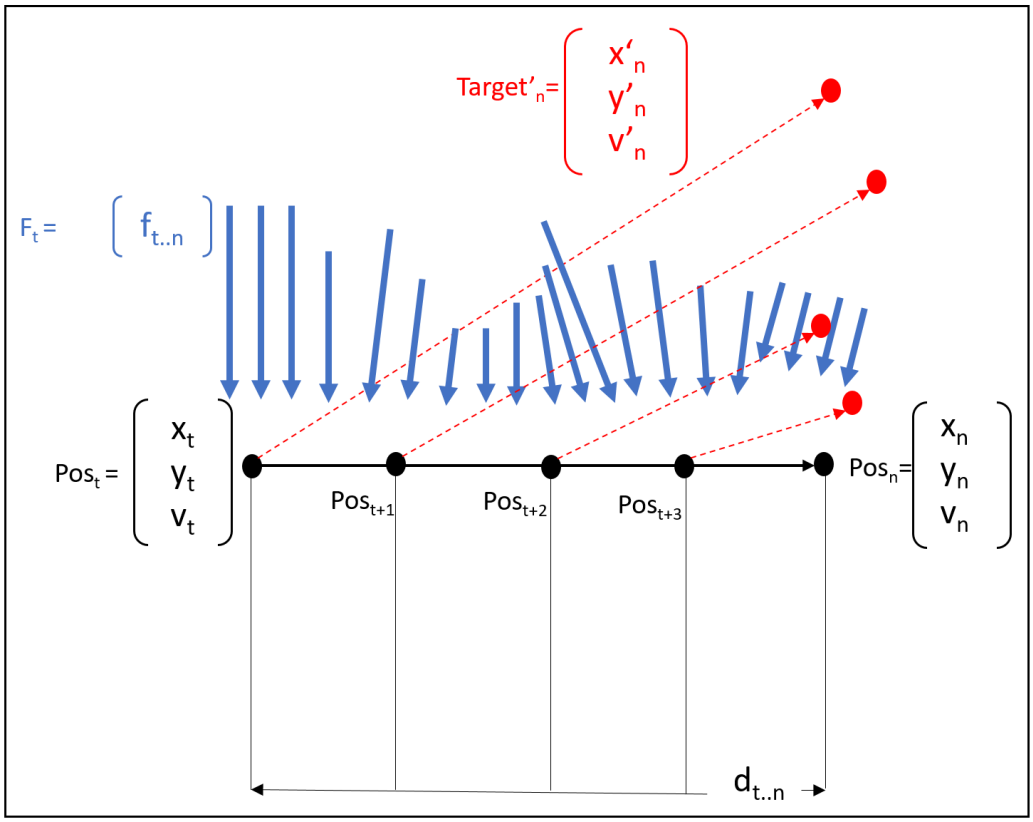


Figure 6.14: High-level illustration of way-point navigation augmentation method. Black solid line and position points denote the path we wish to maintain. Blue arrows represent the external force vector acting on the ASV, which are wind and current in our setup. Red points and arrows represent the intermediate way-points provided to the Pixhawk navigator and their associated target headings.

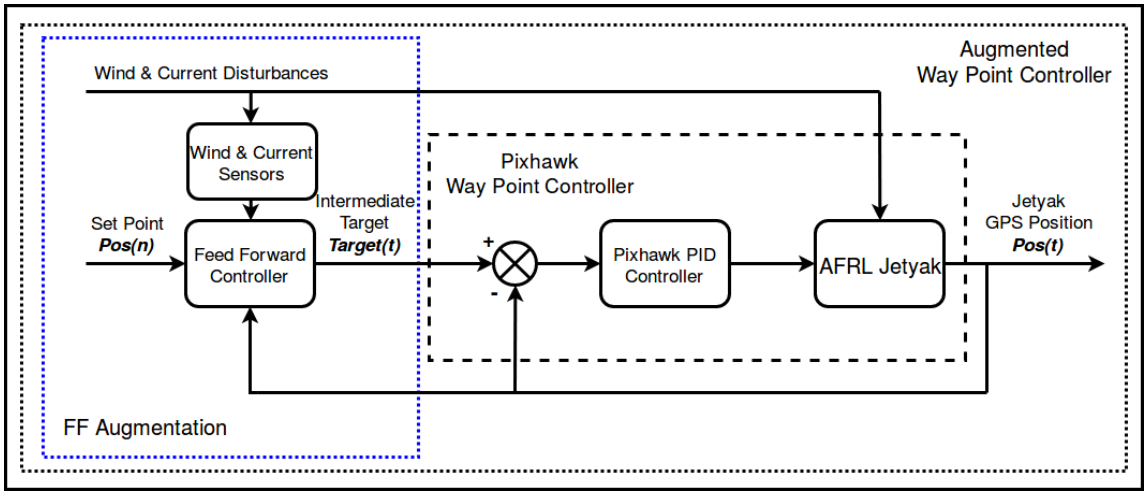


Figure 6.15: The way-point navigation PID controller used in the Pixhawk PX4 augmented by our intermediate way-point offset generator.

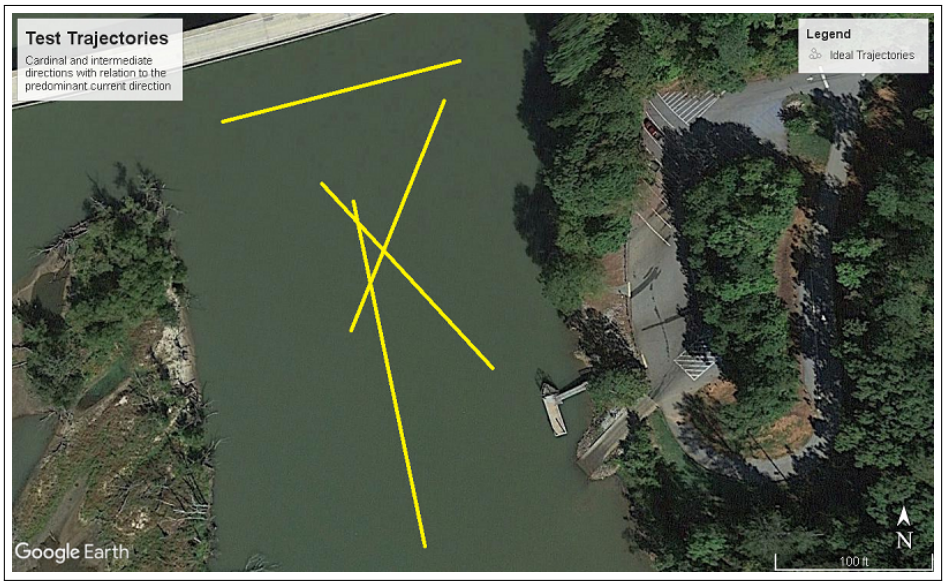


Figure 6.16: Test patterns run in both directions to establish a control baseline for performance evaluation in currents of less than 1 m/s, depending on location of the ASV in the Saluda River's cross-section.

Figure 6.17: Pixhawk PID controlled way-point navigator tracking in slow currents with the ASV travelling mainly (a) against the predominant direction of the current; (b) with the predominant direction of the current – white line: target trajectory, red line: actual executed trajectory.

Figure 6.18: Augmented Pixhawk way-point navigator tracking in slow currents with the ASV travelling mainly (a) against the predominant direction of the current; (b) with the predominant direction of the current – white line: target trajectory, yellow line: actual executed trajectory.

Chapter 7

CONCLUSION

> A pessimist sees the difficulty in every opportunity; an optimist sees the opportunity in every difficulty.
>
> Winston S. Churchill

In this dissertation, we have described a novel solution for building an ASV for long-term deployment capable of mapping, predicting, exploring, and safely controlling itself in highly dynamic environments. We have described several alternative approaches that are used in larger environments, on larger vessels, or in unrelated disciplines.

Our approach is unique in its capability of deploying in narrow, shallow, and fast-moving currents, which up till now, have required manual and closely supervised collection techniques. Since our approach is developed on a small, lightweight, agile platform, our ability to access remote areas as well as highly disturbed areas safely improves on previous work. Our selection of sensors, meticulous reverse engineering, and calibration of each sensor results in an inexpensive array of accurate sensors capable of measuring the phenomena of a dynamic environment used to enable our latter contributions.

Our latter contributions consist of leveraging Gaussian Processes to infer the unvisited portions of our environment and mapping them for future use. In addition, our approach of considering wind, water current and depth as our input space to the GP allows us to extend these mappings and inferences to produce an efficient algo-

rithm for exploration in highly disturbed environments during the very early stages of deployment.

Finally, our method for adapting the ASV controls using real-time measurement results in the end-state ability of the ASV to be aware of the changing dynamics and make in-course adjustments to remain on task and complete its mission, safely, upto and, with future strategies, beyond the physical limitations of the ASV.

7.1 Future Work

The problem of autonomous exploration and mapping of unknown environments, especially marine environments, presents several challenges to maintaining absolute control. In this case, our future work includes adapting the high-level control algorithm we have presented and validated in this document into the top level architecture of the ASV controller. This lower-level control could have two positive impacts. First, it can greatly reduce the requirement for online transformations and calculations of intermediate way-points. In addition, the ability to directly change the linear and angular velocities (v, ω) presents the opportunity for even greater granular control. Further collaboration within the mechanical and computer engineering communities is required to ensure this seamless integration. Once lower-level control augmentation is available, we believe the requirement to produce precise and deliberate turning capability will be enabled.

Another challenge to be considered for future research is the communication architecture required to support a team of ASVs operating in concert. This dissertation illustrates a limitation of two nodes operating within a two kilometer proximity as our soft bound for number of nodes per ground station. Robust communication for fleets greater than two ASVs per ground station is possible as illustrated in the Sharaf *et al.* paper[42] denoting node and distance limitations, while targeting the lowest cost. Advanced technologies such as WIMAX exist, but the spectrum leasing costs, avail-

ability, and required equipment present concerns that each researcher must weigh.

Expanding deployment areas, the addition of the mapping, prediction, and controls to include coastline surf conditions would be an interesting research area. The feasibility of this small vehicle negotiating such a surface would be the first question for a researcher to deal with. If feasible, then the integration of IMU readings into an algorithm that senses surf and deploys countermeasures to negotiate swells and breaks is likely possible.

Finally, the integration of object recognition, avoidance, and Coast Guard Regulation adherence as proposed by Benjamin *et al.* [12] will provide the ultimate, truly autonomous capability. Sensor and physical requirements to accomplish this task include adding Lidar and vision based detection capabilities to the ASV presented in this thesis. These additions will complete the robust capabilities useful in a single or fleet of Jetyaks platforms operating in highly dynamic conditions.

7.2 Final Words

In total, the research and implementation goals realized in this dissertation are essential building blocks for the field robotics community. The trials, methods, lessons learned, and informative results presented serve as a solid start point for roboticists seeking to push the boundary between explored areas and previously inaccessible waterways.

Bibliography

[1] Ardupilot rover. http://ardupilot.org/rover/. Accessed: 2016-09-13.

[2] Mokai Manufacturing Inc. http://www.mokai.com/. Accessed: 11 August 2018.

[3] Ping DSP Inc. http://www.pingdsp.com/3DSS-DX-450. [Accessed: 2018-05-11].

[4] NuttX Real-Time Operating System — NuttX Real-Time Operating System, 2018. URL http://nuttx.org/doku.php?id=nuttx&rev=1531430789. Accessed: 14-July-2018.

[5] AFRL. Jetyak Documentation. https://afrl.cse.sc.edu/afrl/resources/JetyakWiki/index.html, 2018. [Online; accessed 19 July 2018].

[6] W. H. Al-Sabban, L. F. Gonzalez, and R. N. Smith. Wind-energy based path planning for unmanned aerial vehicles using Markov decision processes. pages 784–789, 2013.

[7] K. J. Åström and T. Hägglund. *PID controllers: theory, design, and tuning*, volume 2. Instrument society of America Research Triangle Park, NC, 1995.

[8] K. J. Åström and T. Hägglund. The future of pid control. *Control engineering practice*, 9(11):1163–1175, 2001.

[9] D. P. Atherton. Pid controller tuning. *Computing & control engineering journal*, 10(2):44–50, 1999.

[10] J. Banfi, A. Quattrini Li, I. Rekleitis, F. Amigoni, and N. Basilico. Strategies for coordinated multirobot exploration with recurrent connectivity constraints. *Autonomous Robots*, 42:875–894, 2018. doi: 10.1007/s10514-017-9652-y.

[11] N. Basilico and F. Amigoni. Exploration strategies based on multi-criteria decision making for searching environments in rescue operations. 31(4):401, 2011.

[12] M. R. Benjamin and J. A. Curcio. Colregs-based navigation of autonomous marine vehicles. In *Autonomous Underwater Vehicles, 2004 IEEE/OES*, pages 32–39. IEEE, 2004.

[13] K. Betke. The nmea 0183 protocol. *Standard for Interfacing Marine Electronics Devices, National Marine Electronics Association, Severna Park, Maryland, USA*, 2001.

[14] S. G. B. F. S. Boericke. Paddlewheel speed correction, 10 2002. URL `https://patents.google.com/patent/US7110908B2/`.

[15] T. Buchanan and W. Somers. Discharge measurements at gaging stations: U.s. geological survey techniques of water-resources investigations. 1969. URL `https://pubs.usgs.gov/twri/twri3a8/`. Accessed: 2017-08-10.

[16] J. Candela, R. C. Beardsley, and R. Limeburner. Separation of tidal and subtidal currents in ship-mounted acoustic doppler current profiler observations. *Journal of Geophysical Research: Oceans*, 97(C1):769–788, 1992.

[17] H. Choset. Coverage for robotics – a survey of recent results. *Annals of Mathematics and AI*, 31(1-4):113–126, 2001. ISSN 1012-2443.

[18] P. Coulibaly, F. Anctil, and B. Bobee. Multivariate reservoir inflow forecasting using temporal neural networks. *Journal of Hydrologic Engineering*, 6(5):367–376, 2001.

[19] J. Curcio, J. Leonard, and A. Patrikalakis. SCOUT-a low cost autonomous surface platform for research in cooperative autonomy. In *MTS/IEEE OCEANS*, pages 725–729, 2005.

[20] P. Encarnaçao and A. Pascoal. Combined trajectory tracking and path following: an application to the coordinated control of autonomous marine craft. volume 1, pages 964–969. IEEE, 2001.

[21] J. Fraga, J. Sousa, G. Cabrita, P. Coimbra, and L. Marques. Squirtle: An ASV for inland water environmental monitoring. In *ROBOT2013: First Iberian Robotics Conference*, pages 33–39. Springer, 2014.

[22] E. Galceran and M. Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258 – 1276, 2013. ISSN 0921-8890.

[23] Y. Girdhar, A. Xu, B. B. Dey, M. Meghjani, F. Shkurti, I. Rekleitis, and G. Dudek. MARE: Marine Autonomous Robotic Explorer. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5048–5053, 2011.

[24] J. Han. From pid to active disturbance rejection control. *IEEE transactions on Industrial Electronics*, 56(3):900–906, 2009.

[25] Q. Hernandez-Escobedo, F. Manzano-Agugliaro, J. A. Gazquez-Parra, and A. Zapata-Sierra. Is the wind a periodical phenomenon? the case of mexico. *Renewable and Sustainable Energy Reviews*, 15(1):721–728, 2011.

[26] M. A. Hsieh, E. Forgoston, T. W. Mather, and I. B. Schwartz. Robotic manifold tracking of coherent structures in flows. In *IEEE International Conference on Robotics and automation (ICRA)*, pages 4242–4247, 2012.

[27] M. A. Hsieh, H. Hajieghrary, D. Kularatne, C. R. Heckman, E. Forgoston, I. B. Schwartz, and P. A. Yecko. Small and adrift with self-control: using the environment to improve autonomy. In *Robotics Research*, pages 387–402. Springer, 2018.

[28] V. T. Huynh, M. Dunbabin, and R. N. Smith. Predictive motion planning for AUVs subject to strong time-varying currents and forecasting uncertainties. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2015.

[29] N. Karapetyan, J. Moulton, J. S. Lewis, A. Quattrini Li, J. M. O'Kane, and I. M. Rekleitis. Multi-robot dubins coverage with autonomous surface vehicles. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pages 2373–2379, 2018.

[30] N. Karapetyan, A. Braude, J. Moulton, J. A. Burstein, S. Whitea, J. M. O'Kane, and I. Rekleitis. Riverine coverage with an autonomous surface vehicle over known environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019 *submitted*.

[31] R. Kelly and J. Moreno. Learning pid structures in an introductory course of automatic control. 44(4):373–376, 2001.

[32] P. Kimball, J. Bailey, S. Das, R. Geyer, T. Harrison, C. Kunz, K. Manganini, K. Mankoff, K. Samuelson, T. Sayre-McCord, F. Straneo, P. Traykovski, and H. Singh. The WHOI Jetyak: An autonomous surface vehicle for oceanographic research in shallow or dangerous waters. In *Proc. AUV*, pages 1–7, 2014. doi: 10.1109/AUV.2014.7054430.

[33] R. Kostaschuk, J. Best, P. Villard, J. Peakall, and M. Franklin. Measuring

flow velocity and sediment transport with an acoustic doppler current profiler. *Geomorphology*, 68(1-2):25–37, 2005.

[34] D. Kularatne, S. Bhattacharya, and M. A. Hsieh. Optimal path planning in time-varying flows using adaptive discretization. 3(1):458–465, 2018.

[35] L. Landberg. Short-term prediction of the power production from wind farms. *Journal of Wind Engineering and Industrial Aerodynamics*, 80(1-2):207–220, 1999.

[36] H. S. Lee and M. Tomizuka. Robust motion controller design for high-accuracy positioning systems. *IEEE Transactions on Industrial Electronics*, 43(1):48–55, 1996.

[37] Y. Lee, S. Park, and M. Lee. Pid controller tuning to obtain desired closed loop responses for cascade control systems. *Industrial & engineering chemistry research*, 37(5):1859–1865, 1998.

[38] J. S. Lewis, W. Edwards, K. Benson, I. Rekleitis, and J. M. O'Kane. Semi-boustrophedon coverage with a dubins vehicle. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5630–5637. IEEE, 2017.

[39] A. Q. Li, I. Rekleitis, S. Manjanna, N. Kakodkar, J. Hansen, G. Dudek, L. Bobadilla, J. Anderson, and R. N. Smith. Data correlation and comparison from multiple sensors over a coral reef with a team of heterogeneous aquatic robots. In *International Symposium of Experimental Robotics (ISER)*, Tokyo, Japan, Mar. 2016.

[40] R. Mackay. Ardupilot. `https://github.com/ArduPilot/ardupilot_wiki/blob/master/rover/source/docs/gettit.rst`, 2016.

[41] P. Mahacek, T. Berk, A. Casanova, C. Kitts, W. Kirkwood, and G. Wheat. Development and initial testing of a swath boat for shallow-water bathymetry. In *OCEANS 2008*, pages 1–6. IEEE, 2008.

[42] S. Malebary, J. Moulton, A. Quattrini Li, and I. Rekleitis. Experimental analysis of radio communication capabilities of multiple autonomous surface vehicles. In *OCEANS 2018 MTS/IEEE Charleston*, pages 1–6. IEEE, 2018.

[43] S. Manjanna, A. Q. Li, R. N. Smith, I. Rekleitis, and G. Dudek. Heterogeneous Multirobot System for Exploration and Strategic Water Sampling. In *IEEE International Conference on Robotics and Automation*, pages 4873–4880, Brisbane, Australia, May 2018.

[44] A. Matos, A. Martins, A. Dias, B. Ferreira, J. M. Almeida, H. Ferreira, G. Amaral, A. Figueiredo, R. Almeida, and F. Silva. Multiple robot operations for maritime search and rescue in eurathlon 2015 competition. In *MTS/IEEE OCEANS 2016-Shanghai*, pages 1–7, 2016.

[45] D. W. Meals and S. Dressing. Surface water flow measurement for water quality monitoring projects. *Tech Notes*, 3, 2008.

[46] M. Michini, M. A. Hsieh, E. Forgoston, and I. B. Schwartz. Robotic tracking of coherent structures in flows. 30(3):593–603, 2014.

[47] M. S. Miranda and R. W. Dunn. One-hour-ahead wind speed prediction using a bayesian methodology. In *Power Engineering Society General Meeting, 2006. IEEE*, pages 6–pp. IEEE, 2006.

[48] J. Moulton, N. Karapetyan, S. Bukhsbaum, C. McKinney, S. Malebary, G. Sophocleous, A. Quattrini Li, and I. Rekleitis. An autonomous surface vehicle for long term operations. In *OCEANS 2018 MTS/IEEE Charleston*, pages 1–10. IEEE, 2018.

[49] J. Moulton, N. Karapetyan, A. Q. Li, and I. Rekleitis. External force field modeling for autonomous surface vehicles. *arXiv preprint arXiv:1809.02958*, 2018.

[50] J. Moulton, N. Karapetyan, M. Kalaitzakis, A. Quattrini Li, N. Vitzilaios, and I. Rekleitis. Dynamic autonomous surface vehicle controls, aiding disaster response and enabling responsible port growths. In *Field and Service Robotics (FSR)*, 2019 *submitted*.

[51] J. Moulton, N. Karapetyan, M. Kalaitzakis, A. Quattrini Li, N. Vitzilaios, and I. Rekleitis. Effects modeling – enabling proactive autonomous surface vehicle control in dynamic environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019 *submitted*.

[52] R. R. Murphy, K. L. Dreger, S. Newsome, J. Rodocker, E. Steimle, T. Kimura, K. Makabe, F. Matsuno, S. Tadokoro, and K. Kon. Use of remotely operated marine vehicles at minamisanriku and rikuzentakata japan for disaster recovery. In *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 19–25, 2011.

[53] A. O'Dwyer. *Handbook of PI and PID controller tuning rules*. Imperial College Press, 2009.

[54] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[55] P. K. Penumarthi, A. Quattrini Li, J. Banfi, N. Basilico, F. Amigoni, I. Rekleitis, J. M. O'Kane, and S. Nelakuditi. Multirobot exploration for building

communication maps with prior from communication models. In *International Symposium on Multi-Robot and Multi-Agent Systems*, pages 90–96, 2017.

[56] A. Pereira, J. Das, and G. S. Sukhatme. An experimental study of station keeping on an underactuated asv. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 3164–3171. IEEE, 2008.

[57] A. A. Pereira, J. Binney, G. A. Hollinger, and G. S. Sukhatme. Risk-aware path planning for autonomous underwater vehicles using predictive ocean models. 30 (5):741–762, 2013.

[58] A. J. Plueddemann and R. A. Weller. . current and water property measurements in the coastal ocean. *The Global Coastal Ocean-Processes and Methods*, 10:367, 2005.

[59] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. ROS: an open-source Robot Operating System. In *ICRA workshop on open source software*, volume 3.2, page 5. Kobe, Japan, 2009.

[60] E. Ramsden. *Hall-effect sensors: theory and application*. Elsevier, 2011.

[61] K. Rasal. Navigation & control of an automated swath surface vessel for bathymetric mapping. *Santa Clara University*, 2013. URL `https://scholarcommons.scu.edu/mech_mstr/2`. Accessed: 2017-10-18.

[62] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[63] D. Rodriquez, M. Franklin, and C. Byrne. A study of the feasibility of autonomous surface vehicles. *Worcester Polytechnic Institute*, 2012. URL `https://web.wpi.edu/Pubs/E-project/Available/E-project-121212-135500/unrestricted/ASV_IQP.pdf`.

[64] K. Savla, F. Bullo, and E. Frazzoli. The coverage problem for loitering Dubins vehicles. In *Decision and Control, 2007 46th IEEE Conference on*, pages 1398–1403, Dec 2007.

[65] T. K. Schroeder. Vessel hull transducer modular mounting system, 06 2007. URL `https://patents.google.com/patent/US7236427B1/`.

[66] P. Segui-Gasco, H.-S. Shin, A. Tsourdos, and V. J. Seguí. Decentralised submodular multi-robot task allocation. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2829–2834, Sept 2015. doi: 10.1109/IROS.2015.7353766.

[67] F. Shkurti, A. Xu, M. Meghjani, J. C. G. Higuera, y. Girdhar, P. Giguere, B. B. Dey, J. Li, A. Kalmbach, C. Prahacs, K. Turgeon, I. Rekleitis, and G. Dudek. Multi-domain monitoring of marine environments using a heterogeneous robot team. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1447–1753, 2012.

[68] S. Skogestad. Simple analytic rules for model reduction and pid controller tuning. *Journal of process control*, 13(4):291–309, 2003.

[69] C. A. Smith and A. B. Corripio. *Principles and practice of automatic process control*, volume 2. Wiley New York, 1985.

[70] S. S. Soman, H. Zareipour, O. Malik, and P. Mandal. A review of wind power and wind speed forecasting methods with different time horizons. In *North American power symposium (NAPS), 2010*, pages 1–8. IEEE, 2010.

[71] N. Stanfelj, T. E. Marlin, and J. F. MacGregor. Monitoring and diagnosing process control performance: the single-loop case. *Industrial & Engineering Chemistry Research*, 32(2):301–314, 1993.

[72] T.-C. Tsao. Optimal feed-forward digital tracking controller design. *Journal of dynamic systems, measurement, and control*, 116(4):583–592, 1994.

[73] A. Visioli. *Practical PID control*. Springer Science & Business Media, 2006.

[74] S. Waharte, N. Trigoni, and S. Julier. Coordinated search with a swarm of UAVs. In *IEEE Annual Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops*, 2009. doi: 10.1109/SAHCNW.2009.5172925.

[75] Y. Yang, J. Du, H. Liu, C. Guo, and A. Abraham. A trajectory tracking robust controller of surface vessels with disturbance uncertainties. 22(4):1511–1518, 2014.

[76] T. H. Yorke and K. A. Oberg. Measuring river velocity and discharge with acoustic doppler profilers. *Flow Measurement and Instrumentation*, 13(5-6):191–195, 2002.

Appendix A

Multi-node Communications

A.1 Introduction

The rapid advancement in sensor modalities enables fleets of robots to carryout their missions autonomously and efficiently by maintaining reliable communication links between them and base-stations [74]. Robots nowadays are more sophisticated in terms of exploration capability (drive, fly, and dive autonomously) based on sensory data. However, monitoring exploring robots during a mission is still crucial to minimize potential loss (financially or injury in case of accidents). Hence, it is essential to provide low latency, reliable, and robust communication channels to ensure continuous and effective monitoring of autonomous robots during missions. The desirable range along with the number of nodes are the key factors that define the frequency band (VHF, UHF, SHF, etc.) to be used in the radio spectrum for communications. Several other factors contribute to degrading the quality of communications, such as, but not limited to, environmental noise and weather outdoors, and walls, obstacles, and spectrum overlap indoors. Various technologies are widely used nowadays and have proven their effectiveness in 900+ MHz communications. Some examples of these types of communications that can be seen in our daily lives are, Wi-Fi, Wimax, Zigbee, Bluetooth, etc. Although most of these wireless technologies allow bi-directional communications, they differ in several technical aspects such as communication range, bandwidth, data rate, latency, and are prone to noise. Additionally, an important aspect to consider is the cost factor. Therefore, driven by these observations, our

Figure A.1: Jetyaks equipped with RFD900+ modems.

work was done in an effort to use cheap off-the-shelf 900 MHz modems to test their capabilities for long range communications among a fleet of autonomous vehicles. It is worth mentioning that there exists 900 MHz industrial solution for long distance communication, e.g., AWK-3191 Series[1]. The aforementioned series can cover up to 30 km in line of sight (LOS) at 6 Mbps for point-to-point connection and 5 km in LOS for point-to-multi-point at the same rate. The cost for such a device ranges between $1999 - $2199.

This portion of the thesis presents a performance evaluation that can be used as a guide to understand the capability and reliability of long range communications using inexpensive 900MHz modems. Such a study can then be used to establish a better network for a team consisting of multiple robots in marine environments, where network infrastructure might not be available and long range distance communication is necessary. We focus on communicating in the ISM Band (900 MHz) when experimenting indoors and outdoors, because of the low-cost/weight of the hardware, and the potential to cover longer ranges with better penetration through obstacles than higher frequencies. In particular, we use cheap, off-the-shelf Radio Frequency (RF) modems – Open Source RFD900+[2] (widely used for peer-to-peer telemetry communications).

[1] `https://www.moxa.com/doc/brochures/Brochure-AWK-3191.pdf/`

[2] `http://store.rfdesign.com.au/rfd-900p-modem/`

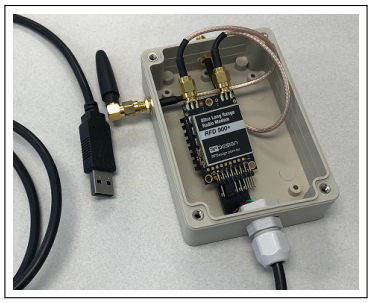


Figure A.2: RFD900+ modem installed in a waterproof box, connected by FTDI cable.

Several indoor and outdoor experiments show how different network configurations affect the quality of communication in terms of latency, range, data rate and RSSI (Received Signal Strength Indicator) value. Specifically, in indoor environments we assume stationary robots due to limited space. In outdoor environments, experiments were conducted by mounting RFD900+ hardware on a fleet of Autonomous Surface Vehicles (ASV) masts (expanding on the WHOI Jetyak [32]). Different experimental dimensions, including the number of robots and the network topology with a base station, have been evaluated. The main contribution of this paper is to give an insight of the different setups that can be easily adopted when monitoring autonomous vehicles using basic hardware and how to optimize and tune parameters to achieve higher throughput and range. We also provide a fruitful comparison between the quality of communications in LOS vs. N-LOS in point-to-point and multi-points connections.

A.2 Experimental Setup

To evaluate the capabilities of long range communications using 900 MHz band in both point-to-point and multi-point scenarios, we first need to take a look at the various parameters that can be adjusted by the user. These parameters are closely related and can greatly affect the performance of the communication. Table A.1 lists the available and configurable parameters in the RFD900+ modems as well as the default (out of box), minimum, and maximum value for each parameter.

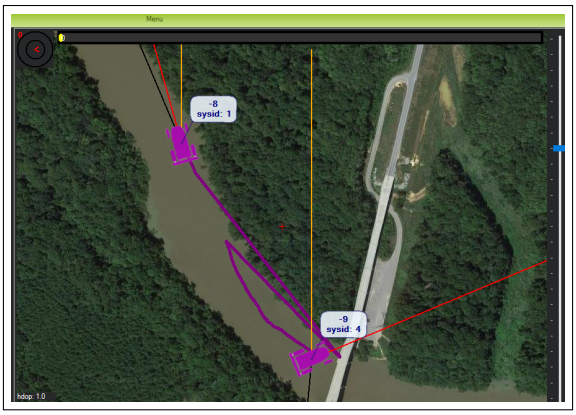


Figure A.3: Live monitoring of 2 ASVs communicating with 1 GCS at the Congaree river in South Carolina using Mission Planner (Ground Control Software).

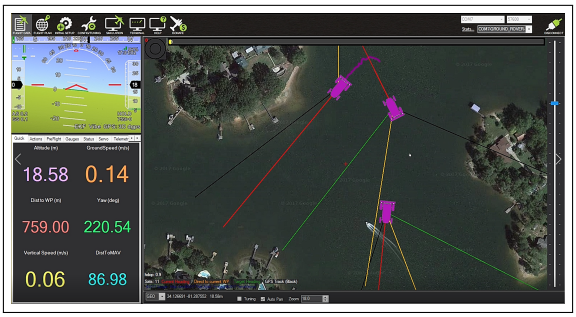


Figure A.4: Live monitoring of 3 ASVs deployed at Lake Murray. Data was collected from various experiments where each ASV was connected to a separate GCS operating on different Net IDs, and this illustration where all were monitored by a single GCS. All BSs were connected to laptops stationed at shore running Mission Planner software for monitoring and data collection.

A.2.1 Platforms

We adopt an ASV from the WHOI project [32] custom-modified in our lab to serve as remote autonomous nodes. The RFD900+ modems were mounted on the mast (Fig. A.1) of each ASV to minimize any interference that may be caused by other on-board electrical and electronic components, *e.g.*, GPS module, Gyroscope, Compass, Accelerometer Sensors.

Table A.1: RFD900+ configurable parameters with their minimum, maximum and default (out of box) values

Parameter	**Description**	**Default**	**Max**	**Min**
Format	EEPROM Version	—	—	—
Serial Speed	Serial data rate (unit: kB)	57	115	2
Air Speed	Data rate (unit: kB)	64	250	2
Net ID	Network ID	25	499	0
Tx Power	unit: dBm	20	30	0
ECC	Error correction code	0	1	0
Mavlink	Mavlink frame & report	0	1	0
Op Resend	Opportunistic resend	0	1	0
Min Freq	In kHz	915	927	902
Max Freq	In kHz	928	928	903
Num Channel	Frequency hopping channels	20	50	5
Duty Cycle	Percentage of transmission	100	100	10
LBT RSSI	Listen before talk	0	1	0
Manchester	Manchester encoding	0	1	0
RTS/CTS	Request/Clear to send	0	1	0
Node ID	Unique ID for each node	2	29	0
Node Dest	Remote ID	65535	29	0
Sync Any	Broadcast feature	0	1	0
Node Count	Total number of nodes	2	30	2

A.2.2 Dimensions

Many factors are crucial to take into consideration when evaluating wireless communications. The desired range, number of nodes, surrounding noise, and obstacles between transmitters and receivers are the most common and widely used to analyze communication quality. The aforementioned factors can be considered environmental. In the following, we detail our experimental setup and the consideration we took for running our tests.

Hardware

We employ half wave dipole antennas approximately 3 dBi gain with omni directional radiation pattern. It is reported on the RFD900p manufacturer data-sheet that depending on the antennas installed, communications can be carried for up to 40km in

Figure A.5: Electronics box of the ASV, which contains controllers and sensors, including GPS and Arduino. Each of the components can be a source of noise.

case of LOS[3]. Hence, we adopt such high gain antenna to be used in BSs and ASVs to validate this claim.

Number of nodes

We can summarize our experiments into two main categories: A point-to-point and multi-point scenarios. The first scenario consists of one ASV and one Base Station (BS), equipped with the same RFD900+ modem, connected to a laptop through a serial to USB cable (see Fig. A.2). In the multi-point scenario, several setups are tested, where one ASV broadcasts to two separate BSs to emulate multiple GCSs monitoring the same vehicle (Fig. A.3). Another setup consists of multiple ASVs monitored by a single BS at the same time (Fig. A.4).

LOS vs. Non-line of sight (N-LOS)

We explicitly consider scenarios under LOS and N-LOS as experimental dimension, to see how robust the quality of communication is.

[3] http://files.rfdesign.com.au/Files/documents/RFD900/20DataSheet.pdf/

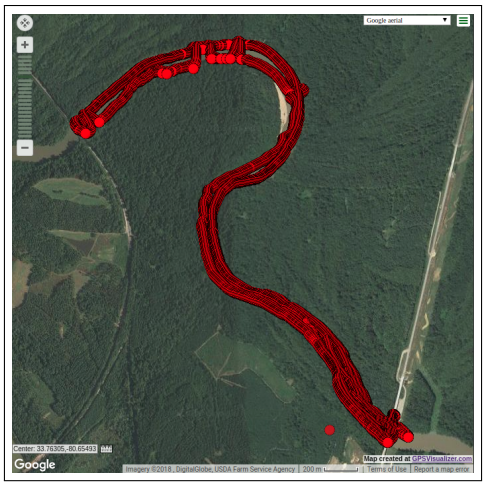


Figure A.6: GPS traces of the four ASVs during a deployment at the Congaree river in N-LOS.

Figure A.7: GPS trace-path of deployed ASV on Lake Murray to evaluate communication between BS and ASV in LOS. Yellow traces represent low latency, orange represent ASV executing way-point missions, and blue traces evaluate the quality of communication and range limitation.

A.2.3 Metrics

We collect two sets of data, telemetry logs (T-logs) and binary (Bin) logs. Both of these types of logs contain mostly the same data from the ASV sensors and modules. However T-logs gets streamed live to the GCS/BS using the RFD900+ modems and the Bin logs get stored locally for collection later. Therefore, different information can be extracted from these logs depending on the type of evaluations. For instance, T-logs provide channel metrics such as RSSI, Noise, and receiving error for Local and Remote nodes Figures A.3 and A.4, which define the Link quality between transmit-

ters and receivers, while Bin logs are more reliable for obtaining sensitive data (*i.e.*, GPS traces) due to local logging in the ASVs on-board storage Figures A.6 and A.7.

Noise

Several types of noise are well known to have a detrimental impact on the quality of the wireless communications, degrading the quality of the communication link: data transfer rate as well as communication range can dramatically drop, especially when operating outdoors. Examples of common types of noise are environmental noise – *e.g.*, weather – and hardware – *e.g.*, thermal, noise from the ASV's engine magneto. Also, in case of multiple nodes communicating at close distance, each node can be considered a noise source to its neighbor. For simplicity, we categorize the noise measured at GCS and ASV into *local noise* and *remote noise*.

RSSI

A positive value that represents the strength of the signal. It is different than Received Signal Strength (RSS) which defines the actual strength value of the signal represented by a negative value.

Rx error

Represents the error rate in receiving data over air, *i.e.*, packets that didn't pass CRC check.

Distance

The distance between transmitter and receiver which can be ASV or BS.

A.2.4 Scenarios

In this section, we describe the scenarios to collect data, according to the dimensions just mentioned. All scenarios can be categorized based on the location where the experiments were conducted. As said, a key factor when dealing with wireless communication that have a great impact on the link-quality is the presence of LOS between communicating nodes. Therefore, we pick two locations namely Lake Murray and Congaree River, to represent communicating in LOS and in N-LOS respectively as follows:

Lake Murray (LOS)

Lake Murray is about 50,000 acres of open space with minimal obstacles. Therefore, we choose to conduct experiments at this location due to convenience (close to our lab) and optimal conditions (minimal obstacles). Here we setup our scenarios to evaluate the maximum range that can be covered when deploying ASVs in LOS environment, while maintaining reliable monitoring. In particular, we run the following experiments:

1. A pair of one ASV and one BS.
2. Two pairs of one ASV and one BS.
3. Three pairs of one ASV and one BS.
4. Two ASVs connected to one BS, and one ASV connected to a separate BS with different Network-ID.

Congaree River (N-LOS)

We nominate the Congaree River to run and collect data in a N-LOS scenarios. The winding path of the river and bushy surroundings make this an optimal location for

testing and evaluating the quality of communication when there is no visible path between the BS and ASVs (Fig. A.6). We conduct several experiments that are different than the ones at the lake. More complicated scenarios were run at the river to verify the accuracy of the results. Intuitively one can expect to run smoother experiments at the lake than at the river due to space limitation and surrounding obstacles. Hence, we start with one pair of BS and ASV as a point-to-point scenario. Then, we introduce several (up to three) ASVs to the network totaling four ASVs and three BS connected to two laptops as follows:

1. One ASV and One BS forming a point-to-point scenario.
2. One ASV broadcasting to two BSs at the same time, where each BS is connected to separate GCS representing an ASV being monitored by multiple GCSs.
3. Two ASVs connected to one BS.
4. Two pairs of one ASV and one BS, where each pair is assigned with different Network-ID.
5. Two ASVs connected to one BS, and one ASV connected to a separate BS at the same time but with different Network-ID.
6. Two pairs of one ASV connected to one BS, and two ASVs connected to one BS totaling four ASVs in the water and three BSs connected to two GCSs (laptops).

In the aforementioned experiments, T-logs were collected at the corresponding GCS as they were live-streamed. The Bin logs were setup to be stored locally at each ASV and were collected from a 4 GB SD-card located in Pixhawk boxes (Fig. A.5) upon the end of all experiments. Several key observations and notes regarding the results are discussed next.

A.3 Summary of Results and Discussion

In this section, we provide an inclusive summary of all results obtained from conducting experiments on Lake Murray and Congaree river.

It is worth mentioning that it takes a significant amount of time to carry such field experiments due to loading and hauling ASVs, setting up and configuring parameters, unexpected technical issues, collecting data and perform analysis. Here, we highlight the main observations from the analysis and provide a discussion that guides future setups. Figures A.8 and A.9 show an analysis of one scenario at the lake and the river respectively.

A.3.1 Impact of number of ASVs

By setting up a point-to-point communication, *i.e.*, one ASV and one BS, we observed a low latency and reliable communication in the lake and the river. We configured the nodes to communicate at the maximum available data rate *i.e.*, 250 kbps. As for the duty cycle, we kept the ASV at 100% transmission cycle since we cared about monitoring the ASV. The BS was given the node ID 0, the ASV node ID 1, and they were both configured to communicate with each other by setting up the node destination variable Table A.2. When another ASV joins the network (with the same configuration), a drastic change to RSSI values is observed, as shown for example in Figure A.10. Nodes can no longer communicate with BS or among themselves. We started from the maximum values that allowed by the modems then worked our way down until we were able to upload missions and monitor all ASVs at the BSs. For instance, we altered the data-rate to be at 128 kbps and cut down duty cycle to 40% for each ASV and 20% for the BS. Additionally, we configured all the node to communicate in a broadcasting behavior by assigning the following values: node destination = 65535, SYNC any = 1, and RTS/CTS = 1. Fig. A.11 shows the RSSI and Noise values of the configuration that gave the best results in our experiments

in point-to-point and multi-point cases.

A.3.2 Impact on range

Experiments conducted at the river, in a N-LOS environment, showed a reliable monitoring of multiple ASVs (up to 3, monitored by 2 BSs) for up to 1 km range. The same configuration provided a range up to 5 km at Lake Murray in a LOS environment. Also, as expected, the range extends with the fewer number of ASVs in the network due to the absence of noise created by neighboring nodes. We also noticed decreasing data rate results in an increase of the range but at the cost of the amount of data to be exchanged.

A.3.3 Discussion

Although RSSI, from the physical layer perspective, is one of the most valuable metrics that can define the quality of communication, several other metrics have to be considered when evaluating the quality of communications. For instance, associating more nodes to the network may increase the value of RSS due to signals colliding. Another important metric is the distance that can be covered. From our experiments, four remote ASVs can be monitored in an open large area more reliably than in a smaller area due to the noise created from neighboring ASVs. Also, it is worth mentioning that trial and error method might be the most effective way when configuring modems for the following reasons. The number of ASVs changes based on the exploring area, environmental factors – *e.g.*, obstacles – other miscellaneous reasons – *e.g.*, antennas type, length, and placement, etc. Therefore, it is hard to find an optimal configuration that can be generalized. On the other hand, we observed an inverse relationship between the number of ASVs vs. maximum possible data rate. Based on the results and analysis, this relation can be defined as follow where n is the number

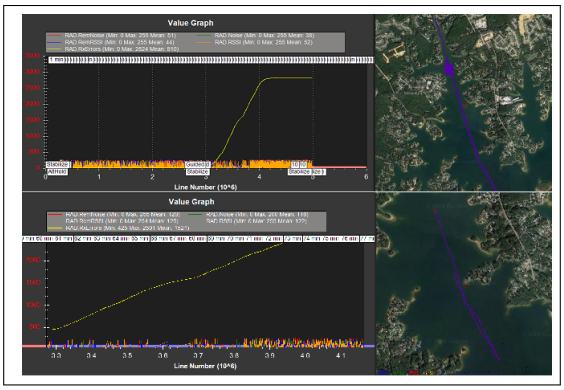


Figure A.8: Analysis example of local and remote RSSI (blue and orange), noise (red and green), and receiving error (yellow) values over distance of a remote (ASV) and a local node (BS) deployed at Lake Murray in LOS. Top map shows the full path and corresponding analysis on the left side of the map. Bottom shows a zoomed in view (segment of path and plot) when accumulated receiving error go beyond 20%.

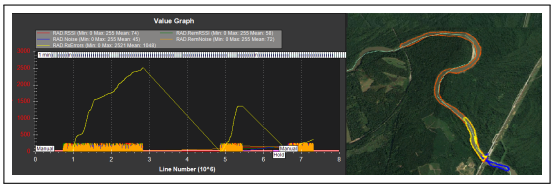


Figure A.9: Analysis of 3 ASVs at the Congaree river with two BSs. Map on the right shows three paths of ASVs. Path colors correspond to quality of communication: Blue-receiving error of less than 20%, yellow-up to 49%, orange- above 50% which considered unreliable for monitoring.

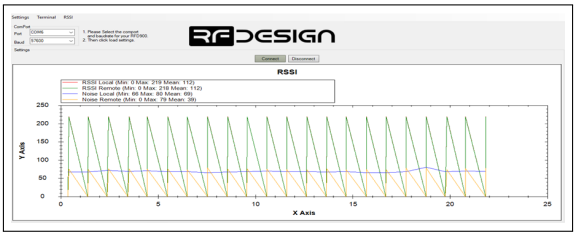


Figure A.10: Initial plot of RSSI and Noise values (Y-Axis) over Time (X-Axis) when a third node (BS or ASV) was added. We can observe the impact of introducing a new node on the quality of communication (RSSI and noise).

of remote ASVs:

$$\text{Maximum possible Data rate} = 250kbps/(2^n - 1) \tag{A.1}$$

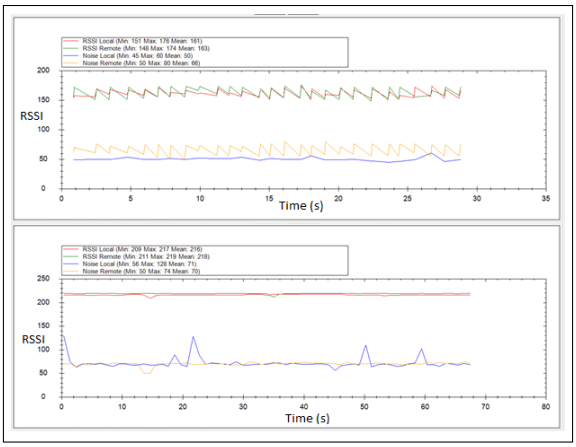


Figure A.11: Plot showing optimal RSSI Vs. Noise values (Y-Axis) over Time (X-Axis) of local and remote nodes (BS and ASV respectively) when communicating in point-to-point (bottom) and multi-point (top) scenarios

.

A.4 Conclusions and Future Work

This paper provides an insight of the different configurations that can be used for communicating over 900 MHz band. We showed how we can utilize off the shelf models, namely RFD900+, to provide bi-directional communications in point-to-point and multi-point setups. Based on the results, a key observation is that using these types of affordable modems can provide long range communications with limited data-rates. It is important to mention that we observed a significant drop in the data-rate when switching form point-to-point communication to multi-point by adding an extra ASV. We observed higher noise that forced us to cut the air-data value by half what it was in a point-to-point configuration. In summary, we tested the capabilities and performance of long range communications using cheap off the shelf modems (approx. $70 per modem), which with some tuning can provide an affordable solution. The viability of this solution depends on the required range and data rate. We showed how the number of actively communicating nodes have the greatest impact to the reliability and quality of communication in terms of the data rate, the receive error rate, and the noise introduced by joining more ASVs to the network. We suggest using

Table A.2: RFD900+ configuration for point-to-point communication scenarios when conducting experiments at the lake and river.

Local Node setting -BS	**Remote node setting -ASV**
S0: FORMAT=27	S0: FORMAT=27
S1: SERIAL SPEED=57	S1: SERIAL SPEED=57
S2: AIR SPEED=250	S2: AIR SPEED=250
S3: NETID=36	S3: NETID=36
S4: TXPOWER=30	S4: TXPOWER=30
S5: ECC=0	S5: ECC=0
S6: MAVLINK=1	S6: MAVLINK=1
S7: OPPRESEND=0	S7: OPPRESEND=0
S8: MIN FREQ=915000	S8: MIN FREQ=915000
S9: MAX FREQ=928000	S9: MAX FREQ=928000
S10: NUM CHANNELS=50	S10: NUM CHANNELS=50
S11: DUTY CYCLE=50	S11: DUTY CYCLE=100
S12: LBT RSSI=0	S12: LBT RSSI=0
S13: MANCHESTER=0	S13: MANCHESTER=0
S14: RTSCTS=0	S14: RTSCTS=0
S15: NODEID=0	S15: NODEID=1
S16: NODEDESTINATION=1	S16: NODEDESTINATION=0
S17: SYNCANY=0	S17: SYNCANY=0
S18: NODECOUNT=2	S18: NODECOUNT=2

a separate network ID when possible to minimize the unintentional noise produced by a neighboring ASV's modem.

Our future research will seek to improve our current communication system. We will consider the construction of a communication map [55] in order to control the ASVs facilitating a communication link to the Ground Control Station while exploring areas larger than the communication ranges [10]. We will also research and experimentally evaluate the maximum number of nodes that can be deployed at once forming a mesh-like network as well as the impact on the range and data rate. Additionally, we will employ and use drones as a communication bridge to extend ASV range from the BS.

Appendix B

SENSOR MICRO-CONTROLLER CODE

While the rest of the code for subscribing and publishing ROS topics and completing the Gaussian Process functions, transforms, and control algorithm are interesting, there are several methods to implement the code that are functionally equivalent. For this reason, the implementation code for these functions and calculations is not listed in the Appendices. However, the driver code for Arudiono micro-controllers is unique onto itself due to the novel nature of using analog, inexpensive sensors for our measurements. For this reason, the following modules are the drivers for the anemometer and current sensor array, and emergency master kill switch respectively.

B.1 Current Sensor Arduino Code

The code below provides the hardware driver required to measure and convert four individual current measurements to strings for usage in creating ROS topics. Frequencies and conversion to differing desired units are carried out here. Target device is the Arduino Mega.

```
//Adapted from hall effect sensor example originally from code by Crenn
   from http://thebestcasescenario.com

#include <Wire.h>

//Varibles used for calculations
```

```
volatile byte ticks_star_fwd = 0;
float Speed_star_fwd = 0;
const byte hallsensor_star_fwd = 2; //The Hall effect sensor (HES) output
    of current sensor connected to pin no 2 of Arduino MEGA
volatile byte ticks_port_fwd = 0;
float Speed_port_fwd = 0;
const byte hallsensor_port_fwd = 3; //The Hall effect sensor (HES) output
    of current sensor connected to pin no 3 of Arduino MEGA
volatile byte ticks_port_rear = 0;
float Speed_port_rear = 0;
const byte hallsensor_port_rear = 19; //The Hall effect sensor (HES)
    output of current sensor connected to pin no 19 of Arduino MEGA
volatile byte ticks_star_rear = 0;
float Speed_star_rear = 0;
const byte hallsensor_star_rear = 21; //The Hall effect sensor (HES)
    output of current sensor connected to pin no 21 of Arduino MEGA

//If changing velocities or units, modify the following equation as well
    as the mps_factor constant for converting knots to unit of choice
float wcurrent_vel=4.8; //current sensor frequency is 4.8Hz so 1 knot =
    4.8 ticks;
float mps_factor = .51444444; //1 knot = 0.51444444 m/s

//This is the setup function where the serial port is initialised, and the
    interrupt is attached
void setup(){
pinMode(hallsensor_star_fwd, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(hallsensor_star_fwd),
    revolutions_star_fwd, CHANGE);
```

```
pinMode(hallsensor_port_fwd, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(hallsensor_port_fwd),
    revolutions_port_fwd, CHANGE);
pinMode(hallsensor_port_rear, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(hallsensor_port_rear),
    revolutions_port_rear, CHANGE);
pinMode(hallsensor_star_rear, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(hallsensor_star_rear),
    revolutions_star_rear, CHANGE);
Serial.begin(9600);
}

void loop () {
ticks_star_fwd = 0; // Make ticks zero before starting interrupts.
ticks_port_fwd = 0; // Make ticks zero before starting interrupts.
ticks_port_rear = 0; // Make ticks zero before starting interrupts.
ticks_star_rear = 0; // Make ticks zero before starting interrupts.
delay (1000);   //Wait 1 second, change frequency here.

//Simply divide number of ticks by the constant required for the desired
    units times the sensor frequency
//uncomment following two lines to calculate current speed in meters per
    second
Speed_star_fwd = ((ticks_star_fwd*mps_factor)/wcurrent_vel); //change to
    meters per second (*.51444444) and divide by frequency given on airmar
    website (/4.8)
Speed_port_fwd = ((ticks_port_fwd*mps_factor)/wcurrent_vel); //change to
    meters per second (*.51444444) and divide by frequency given on airmar
    website (/4.8)
```

```
Speed_port_rear = ((ticks_port_rear*mps_factor)/wcurrent_vel); //change to
    meters per second (*.51444444) and divide by frequency given on airmar
    website (/4.8)
Speed_star_rear = ((ticks_star_rear*mps_factor)/wcurrent_vel); //change to
    meters per second (*.51444444) and divide by frequency given on airmar
    website (/4.8)

//Print calculated Speed at the serial port
  Serial.print("$CS,");
  Serial.print(Speed_star_fwd);
  Serial.print(",");
  Serial.print(Speed_port_fwd);
  Serial.print(",");
  Serial.print(Speed_port_rear);
  Serial.print(",");
  Serial.print(Speed_star_rear);
  Serial.println(",");
}  // end of loop function

void revolutions_star_fwd () {
//This is the interrupt subroutine that increments ticks counts for each
    HES response.
ticks_star_fwd++;
}
void revolutions_port_fwd () {
//This is the interrupt subroutine that increments ticks counts for each
    HES response.
ticks_port_fwd++;
}
```

```
void revolutions_port_rear () {
//This is the interrupt subroutine that increments ticks counts for each
   HES response.
ticks_port_rear++;
}
void revolutions_star_rear () {
//This is the interrupt subroutine that increments ticks counts for each
   HES response.
ticks_star_rear++;
}
```

B.2 Wind Sensor Arduino Code

The code below provides the hardware driver required to measure and convert wind speed, wind direction and rain gauge measurements to strings for usage in creating ROS topics. Frequencies and conversion to differing desired units are carried out here.

```
/*
 Weather Shield Example
 By: Nathan Seidle
 SparkFun Electronics
 Date: November 16th, 2013
 License: This code is public domain but you buy me a beer if you use this
    and we meet someday (Beerware license).
 Much of this is based on Mike Grusin's USB Weather Board code:
    https://www.sparkfun.com/products/10586
 This is a more advanced example of how to utilize every aspect of the
    weather shield. See the basic example if you're just getting started.
```

```
 This code reads all the various sensors (wind speed, direction, rain
    gauge, humidity, pressure, light, batt_lvl) and reports it over the
    serial comm port. This can be easily routed to a datalogger (such as
    OpenLog) or a wireless transmitter (such as Electric Imp).
 Measurements are reported once a second but windspeed and rain gauge are
    tied to interrupts that are calculated at each report.

 This example code assumes the GPS module is not used.
  Updated by Joel Bartlett
  03/02/2017
  Removed HTU21D code and replaced with Si7021
  Updated by Jason Moulton
  12/07/2017
  Altered output for use as topic in ROS node.
 */

#include <Wire.h> //I2C needed for sensors
#include "SparkFunMPL3115A2.h" //Pressure sensor - Search "SparkFun
   MPL3115" and install from Library Manager
#include "SparkFun_Si7021_Breakout_Library.h" //Humidity sensor - Search
   "SparkFun Si7021" and install from Library Manager

MPL3115A2 myPressure; //Create an instance of the pressure sensor
Weather myHumidity;//Create an instance of the humidity sensor

//Hardware pin definitions
//-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
// digital I/O pins
const byte WSPEED = 3;
```

```
const byte RAIN = 2;
const byte STAT1 = 7;
const byte STAT2 = 8;
// analog I/O pins
const byte REFERENCE_3V3 = A3;
const byte LIGHT = A1;
const byte BATT = A2;
const byte WDIR = A0;
//-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
//Global Variables
//-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
long lastSecond; //The millis counter to see when a second rolls by
byte seconds; //When it hits 60, increase the current minute
byte seconds_2m; //Keeps track of the "wind speed/dir avg" over last 2
   minutes array of data
byte minutes; //Keeps track of where we are in various arrays of data
byte minutes_10m; //Keeps track of where we are in wind gust/dir over last
   10 minutes array of data
long lastWindCheck = 0;
volatile long lastWindIRQ = 0;
volatile byte windClicks = 0;

//We need to keep track of the following variables:
//Wind speed/dir each update (no storage)
//Wind gust/dir over the day (no storage)
//Wind speed/dir, avg over 2 minutes (store 1 per second)
//Wind gust/dir over last 10 minutes (store 1 per minute)
//Rain over the past hour (store 1 per minute)
//Total rain over date (store one per day)
```

```
byte windspdavg[120]; //120 bytes to keep track of 2 minute average

#define WIND_DIR_AVG_SIZE 120
int winddiravg[WIND_DIR_AVG_SIZE]; //120 ints to keep track of 2 minute
    average
float windgust_10m[10]; //10 floats to keep track of 10 minute max
int windgustdirection_10m[10]; //10 ints to keep track of 10 minute max
volatile float rainHour[60]; //60 floating numbers to keep track of 60
    minutes of rain

//These are all the weather values that wunderground expects:
int winddir = 0; // [0-360 instantaneous wind direction]
float windspeedmph = 0; // [mph instantaneous wind speed]
float windgustmph = 0; // [mph current wind gust, using software specific
    time period]
int windgustdir = 0; // [0-360 using software specific time period]
float windspdmph_avg2m = 0; // [mph 2 minute average wind speed mph]
int winddir_avg2m = 0; // [0-360 2 minute average wind direction]
float windgustmph_10m = 0; // [mph past 10 minutes wind gust mph ]
int windgustdir_10m = 0; // [0-360 past 10 minutes wind gust direction]
float humidity = 0; // [%]
float tempf = 0; // [temperature F]
float rainin = 0; // [rain inches over the past hour)] -- the accumulated
    rainfall in the past 60 min
volatile float dailyrainin = 0; // [rain inches so far today in local time]
//float baromin = 30.03;// [barom in] - It's hard to calculate baromin
    locally, do this in the agent
float pressure = 0;
float batt_lvl = 11.8; //[analog value from 0 to 1023]
```

```
float light_lvl = 455; //[analog value from 0 to 1023]

// volatiles are subject to modification by IRQs
volatile unsigned long raintime, rainlast, raininterval, rain;
//-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
//Interrupt routines (these are called by the hardware interrupts, not by
   the main code)
//-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
void rainIRQ()
// Count rain gauge bucket tips as they occur
// Activated by the magnet and reed switch in the rain gauge, attached to
   input D2
{
   raintime = millis(); // grab current time
   raininterval = raintime - rainlast; // calculate interval between this
      and last event
   if (raininterval > 10) // ignore switch-bounce glitches less than 10mS
      after initial edge
   {
      dailyrainin += 0.011; //Each dump is 0.011" of water
      rainHour[minutes] += 0.011; //Increase this minute's amount of rain
      rainlast = raintime; // set up for next event
   }
}
void wspeedIRQ()
// Activated by the magnet in the anemometer (2 ticks per rotation),
   attached to input D3
{
   if (millis() - lastWindIRQ > 10) // Ignore switch-bounce glitches less
```

```
        than 10ms (142MPH max reading) after the reed switch closes
    {
        lastWindIRQ = millis(); //Grab the current time
        windClicks++; //There is 1.492MPH for each click per second.
    }
}

void setup()
{
    Serial.begin(9600);
    //Serial.println("Weather Shield Example");
    pinMode(STAT1, OUTPUT); //Status LED Blue
    pinMode(STAT2, OUTPUT); //Status LED Green
    pinMode(WSPEED, INPUT_PULLUP); // input from wind meters windspeed
        sensor
    pinMode(RAIN, INPUT_PULLUP); // input from wind meters rain gauge
        sensor
    pinMode(REFERENCE_3V3, INPUT);
    pinMode(LIGHT, INPUT);

    //Configure the pressure sensor
    myPressure.begin(); // Get sensor online
    myPressure.setModeBarometer(); // Measure pressure in Pascals from 20
        to 110 kPa
    myPressure.setOversampleRate(7); // Set Oversample to the recommended
        128
    myPressure.enableEventFlags(); // Enable all three pressure and temp
        event flags
```

```
    //Configure the humidity sensor
    myHumidity.begin();

    seconds = 0;
    lastSecond = millis();

    // attach external interrupt pins to IRQ functions
    attachInterrupt(0, rainIRQ, FALLING);
    attachInterrupt(1, wspeedIRQ, FALLING);

    // turn on interrupts
    interrupts();
}

void loop()
{
    //Keep track of which minute it is
  if(millis() - lastSecond >= 1000)
    {
        digitalWrite(STAT1, HIGH); //Blink stat LED
    lastSecond += 1000;

        //Take a speed and direction reading every second for 2 minute
            average
        if(++seconds_2m > 119) seconds_2m = 0;

        //Calc the wind speed and direction every second for 120 second to
            get 2 minute average
        float currentSpeed = get_wind_speed();
```

```
windspeedmph = currentSpeed; //jason added this. THis works, remove
    to go back to orginal code.
//float currentSpeed = random(5); //For testing
int currentDirection = get_wind_direction();
windspdavg[seconds_2m] = (int)currentSpeed;
winddiravg[seconds_2m] = currentDirection;
//if(seconds_2m % 10 == 0) displayArrays(); //For testing

//Check to see if this is a gust for the minute
if(currentSpeed > windgust_10m[minutes_10m])
{
    windgust_10m[minutes_10m] = currentSpeed;
    windgustdirection_10m[minutes_10m] = currentDirection;
}

//Check to see if this is a gust for the day
if(currentSpeed > windgustmph)
{
    windgustmph = currentSpeed;
    windgustdir = currentDirection;
}
if(++seconds > 59)
{
    seconds = 0;
    if(++minutes > 59) minutes = 0;
    if(++minutes_10m > 9) minutes_10m = 0;
    rainHour[minutes] = 0; //Zero out this minute's rainfall amount
    windgust_10m[minutes_10m] = 0; //Zero out this minute's gust
}
```

```
        //Report all readings every second
        printWeather();
        digitalWrite(STAT1, LOW); //Turn off stat LED
    }
  delay(100);
}

//Calculates each of the variables that wunderground is expecting
void calcWeather()
{
    //Calc winddir
    winddir = get_wind_direction();
    //Calc windspeed
    //windspeedmph = get_wind_speed(); //This is calculated in the main
        loop
    //Calc windgustmph
    //Calc windgustdir
    //These are calculated in the main loop
    //Calc windspdmph_avg2m
    float temp = 0;
    for(int i = 0 ; i < 120 ; i++)
        temp += windspdavg[i];
    temp /= 120.0;
    windspdmph_avg2m = temp;

    //Calc winddir_avg2m, Wind Direction
    //You can't just take the average. Google "mean of circular
        quantities" for more info
```

```
//We will use the Mitsuta method because it doesn't require trig
    functions
//And because it sounds cool.
//Based on: http://abelian.org/vlf/bearings.html
//Based on:
    http://stackoverflow.com/questions/1813483/averaging-angles-again
long sum = winddiravg[0];
int D = winddiravg[0];
for(int i = 1 ; i < WIND_DIR_AVG_SIZE ; i++)
{
    int delta = winddiravg[i] - D;
    if(delta < -180)
        D += delta + 360;
    else if(delta > 180)
        D += delta - 360;
    else
        D += delta;
    sum += D;
}
winddir_avg2m = sum / WIND_DIR_AVG_SIZE;
if(winddir_avg2m >= 360) winddir_avg2m -= 360;
if(winddir_avg2m < 0) winddir_avg2m += 360;

//Calc windgustmph_10m
//Calc windgustdir_10m
//Find the largest windgust in the last 10 minutes
windgustmph_10m = 0;
windgustdir_10m = 0;
//Step through the 10 minutes
```

```
for(int i = 0; i < 10 ; i++)
{
    if(windgust_10m[i] > windgustmph_10m)
    {
        windgustmph_10m = windgust_10m[i];
        windgustdir_10m = windgustdirection_10m[i];
    }
}
//Calc humidity
humidity = myHumidity.getRH();
//float temp_h = myHumidity.readTemperature();
//Serial.print(" TempH:");
//Serial.print(temp_h, 2);

//Calc tempf from pressure sensor
tempf = myPressure.readTempF();
//Serial.print(" TempP:");
//Serial.print(tempf, 2);

//Total rainfall for the day is calculated within the interrupt
//Calculate amount of rainfall for the last 60 minutes
rainin = 0;
for(int i = 0 ; i < 60 ; i++)
    rainin += rainHour[i];

//Calc pressure
pressure = myPressure.readPressure();

//Calc light level
```

```
    light_lvl = get_light_level();

    //Calc battery level
    batt_lvl = get_battery_level();
}

//Returns the voltage of the light sensor based on the 3.3V rail
//This allows us to ignore what VCC might be (an Arduino plugged into USB
    has VCC of 4.5 to 5.2V)
float get_light_level()
{
    float operatingVoltage = analogRead(REFERENCE_3V3);
    float lightSensor = analogRead(LIGHT);
    operatingVoltage = 3.3 / operatingVoltage; //The reference voltage is
        3.3V
    lightSensor = operatingVoltage * lightSensor;
    return(lightSensor);
}

//Returns the voltage of the raw pin based on the 3.3V rail
//This allows us to ignore what VCC might be (an Arduino plugged into USB
    has VCC of 4.5 to 5.2V)
//Battery level is connected to the RAW pin on Arduino and is fed through
    two 5% resistors:
//3.9K on the high side (R1), and 1K on the low side (R2)
float get_battery_level()
{
    float operatingVoltage = analogRead(REFERENCE_3V3);
    float rawVoltage = analogRead(BATT);
```

```
    operatingVoltage = 3.30 / operatingVoltage; //The reference voltage is
        3.3V
    rawVoltage = operatingVoltage * rawVoltage; //Convert the 0 to 1023
        int to actual voltage on BATT pin

    rawVoltage *= 4.90; //(3.9k+1k)/1k - multiple BATT voltage by the
        voltage divider to get actual system voltage
    return(rawVoltage);
}

//Returns the instataneous wind speed
float get_wind_speed()
{
    float deltaTime = millis() - lastWindCheck; //750ms
    deltaTime /= 1000.0; //Covert to seconds
    float windSpeed = (float)windClicks / deltaTime; //3 / 0.750s = 4
    windClicks = 0; //Reset and start watching for new wind
    lastWindCheck = millis();

    windSpeed *= 0.6669; // was 1.492 for mph units; //4 * 1.492 =
        5.968MPH 4*.6669 = 2.668 m/s
    return(windSpeed);
}

//Read the wind direction sensor, return heading in degrees
int get_wind_direction()
{
    unsigned int adc;
    adc = analogRead(WDIR); // get the current reading from the sensor
```

```
    // The following table is ADC readings for the wind direction sensor
       output, sorted from low to high.
    // Each threshold is the midpoint between adjacent headings. The
       output is degrees for that ADC reading.
    // Note that these are not in compass degree order! See Weather Meters
       datasheet for more information.
    if (adc < 380) return (203);
    if (adc < 393) return (158);
    if (adc < 414) return (180);
    if (adc < 456) return (248);
    if (adc < 508) return (225);
    if (adc < 551) return (293);
    if (adc < 615) return (270);
    if (adc < 680) return (113);
    if (adc < 746) return (135);
    if (adc < 801) return (338);
    if (adc < 833) return (315);
    if (adc < 878) return (68);
    if (adc < 913) return (90);
    if (adc < 940) return (23);
    if (adc < 967) return (45);
    if (adc < 990) return (0);
    return (-1); // error, disconnected?
}
//Prints the various variables directly to the port
//I don't like the way this function is written but Arduino doesn't
   support floats under sprintf
void printWeather()
```

```
{
    calcWeather(); //Go calc all the various sensors
    Serial.print("$WDS,");
    Serial.print(winddir);
    Serial.print(",");
    Serial.print(windspeedmph, 1); //windspeedmph was here, but not
        updated in code. changed to reflect correct output of windspeed
        function.
    Serial.println(",");
}
```

B.3 Remote Kill Switch Arduino Code

The code below provides the hardware driver required to provide an active-low remote safety kill switch. This is desgined to be an active low system, so that if any component of the PixHawk, Arduino lose power, the relay will complete the kill circuit that is included in the Jetyak OEM design. In addition, this will function as a remote kill switch on channel 6 from Taranas RC to PixHawk to Arduino to the relay using its normally closed pin to complete the kill circuit. Normal operation occurs when relay is in normally open mode, which requires coil to be energized from the Arduino.

```
/*
  The circuit:
 * Relay Coil attached from pin 13 to ground
 * Pixhawk AUX output signal attached to pin 2 (again +5 -> Normal
    Operation; 0 -> Kill circuit
 * Note: Also attach LED from pin 13 to ground as an operational
    indicator. Light on = normal operation; light off = kill circuit
```

```
 *
 created July 7, 2017
 by Jason Moulton
 */

// constants won't change. They're used here to set pin numbers
// the number of the pixhawk kill pin
unsigned long pixhawkkillStatusduration;
const int coilPin = 13;    // the number of the relay coil pin

// variable will change: deprecated, was used in original active high
    version of kill switch.
int pixhawkkillStatus = 0;      // variable for reading the pixhawk kill
    pin status

void setup() {
  // initialize the relay coil (+) pin as an output:
  pinMode(coilPin, OUTPUT);
  // initialize the pixhawk kill signal pin as an input:
  pinMode(pixhawkkillPin, INPUT);
  // set the coilPin high 5v for normal operation to begin
  digitalWrite(coilPin, HIGH);
}

void loop() {
  // read the state of the pixhawk kill value:
  pixhawkkillStatusduration = pulseIn(pixhawkkillPin, HIGH);
  // check how long the pixhawkkill Status duration is low.
  // if it is over threshold X, then kill the boat by pixhawkkillStatus
```

```
    duration is HIGH:
  if (pixhawkkillStatusduration <= 1800) {
    // keep energize coil pin high 5v:
    digitalWrite(coilPin, HIGH);
  } else {
    // set deenergize coil pin low 0v :
    digitalWrite(coilPin, LOW);
  }
}
```
