Formation Fidelity of Simulated Unmanned Autonomous Vehicles through Periodic Communication

Jeffrey Newman Twigg

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science
In
Engineering Mechanics

Shane D. Ross, Chairman
L. Glenn Kraige
Craig A. Woolsey

(November 20, 2009)
(Blacksburg Virginia)

Keywords: Leaderless Formation Control, Communication Error

Formation Fidelity of Simulated Unmanned Autonomous Vehicles through Periodic Communication

by

Jeffrey Newman Twigg

Abstract

Controlling a formation of unmanned autonomous vehicles is a daunting prospect even when the formation operates under ideal conditions. When communication between vehicles is limited, maintaining a formation becomes difficult. In some cases the formation may become unstable. While a control law may stabilize a formation of vehicles with good communication, it may not be able to do so with poor communication. The resulting lack of formation stability affects the level of fidelity the formation has to the original control law. Formation fidelity is the degree to which the vehicles in a formation follow the trajectories prescribed by a control law. Many formation control laws assume certain conditions. Perfect formation fidelity is not guaranteed when the vehicles in a formation are no longer operated under those conditions.

We seek to mitigate the detrimental effects of poor communication and other real-world phenomena on formation fidelity. Through simulation we test the effectiveness of a new way to implement an existing formation control law. Real-world conditions such as rigid-body motion, swarm dynamics, poor communication, and other phenomena are assessed and discussed. It is concluded through testing in simulation that it is possible to control a formation of boats by directing each boat with a unique set of waypoints in simulation. While these waypoints do not lead to perfect formation behavior, testing shows that implementing this control law using these waypoints allows the formation to be more robust to reduced communication.

Preface

This thesis was submitted at the *Virginia Polytechnic Institute and State University* on November 20, 2009, as a partial fulfillment of the requirements for the degree of Master of Science in Engineering Mechanics.

This thesis details the research, design, implementation, and testing of a system to enable an unmanned autonomous swarm of boats to move through a planned course with limited communication.

Acknowledgments

I would like to begin by thanking all the people who have taken time to listen to my ideas, many of which are half-baked longshots from left field. I think this recognition encompasses my parents and most of my family, friends, and nearby inanimate objects.

I would also like to thank Roger Cortesi, Dr. Eric Justh, Daniel Robinson, Kevin Galloway, and Patricia Hirsch for their advice and explanations involving a wide range of controls, Java, and Cybele Pro related issues.

In addition, I would like to thank Vincent Wong and Nick Olsen for the work that they did in helping me to integrate the hardware autopilot into the Java framework.

Finally I would like to thank Dr. Shane Ross, Dr. Craig Woolsey, and Dr. Glenn Kraige for their advice and guidance through this project.

Contents

List of Figures

All figures listed are the original work of Jeffrey Newman Twigg

Chapter 1

Introduction: Background and Overview

1.1 Premise

Controlling a formation of unmanned autonomous vehicles is a daunting prospect even when there is perfect communication between vehicles. When communication between vehicles is limited, maintaining a formation becomes difficult, and in some cases the formation may become unstable. An unstable formation is characterized by vehicle collision and/or dispersion of the formation. While a particular control law may stabilize a formation of vehicles with good communication, it may not be able to do so with poor communication. Formation fidelity is the degree to which the vehicles in a realistic implementation follow the trajectories described by the control laws under ideal conditions. The formation's resulting fidelity to the original control law is an indicator of the stability of the formation. Consequently, formation fidelity can be used to determine the implementability of a formation control law. We seek to mitigate the detrimental effects of poor communication on formation fidelity by developing a new system to use an existing formation control law.

There are many applications for formation control. However, the context of immediate interest in this thesis is the ability to control a formation of unmanned autonomous boats in U.S. naval operations. Maintaining a formation is a concern for any mission planner who wants to control more than one vehicle through a set of commands. This sort of coordination allows for the vehicles in a formation to be both unmanned and autonomous. Unmanned vehicles can perform missions that are otherwise considered too dangerous for people. In addition, autonomous vehicles do not require constant supervision. Formation, or "swarm" control, allows for a large number of vehicles to coordinate their movements and actions with minimal user input.

1.2 Major Issues

Engineers working for the United States Navy have developed software to simulate and display rigid-body motion (Cortesi et al. [2006]). Within this simulation framework it is necessary to try to capture the most important aspects of real-world phenomena that a controlled group of boats would encounter. Rigid-body dynamics, less-than-perfect communication, and other maritime phenomena such as wind and ocean waves are important considerations for any realistic boat formation simulation.

The control laws developed by Justh and Krishnaprasad [2003] are a promising method of control in this context because of the control law's success in other domains (Zhang et al. [2004]). One of the main indicators of the success of this thesis research is the extent to which the formation maintains high fidelity to the results of simulations which are performed under ideal conditions. Perfect fidelity indicates that the control law implementation is robust to non-ideal conditions. Many control laws are modified in order to make them more robust to communication problems as well as other real-world constraints. However, it is also possible to implement the control laws so as to mitigate the problems caused by real-world phenomena. When a control law is made more robust through a different implementation we will assess its effectiveness in terms of its fidelity. Perfect fidelity is achieved when the swarm control implementation allows each vehicle to follow the control law's exact path specified. In addition, perfect fidelity would indicate that the swarm control implementation would inherit all of the theoretical qualities and attributes of the original control laws. High fidelity would not prove that the swarm control-law implementation would inherit all of the theoretical attributes of the control-law, but it would indicate that these attributes could still be present to a high degree.

We submit that the lack of complete communication between boats serves as the main obstacle to perfect formation fidelity. Distance between boats combined with multipath interference and other sources of communication error make simulating poor communication a very important problem.

Finally, commercially available autopilots provide a readily available means of compensating for maritime phenomena that would occur in a real-world setting. As a result, our control-law implementation is designed to accommodate a software autopilot which constrains the implementation of the control law.

There are several important issues that need to be discussed, in order to appreciate the problems with realistic formation control. It is important to understand how the control laws prescribe where each boat will go. In addition, the communication error is very important because it closely

affects the implementability of the control laws. Also, understanding the simulation environment is necessary because it is the basis of every test and corresponding result. Finally, understanding hardware autopilots is also important because it will allow the entire control-law implementation to move from simulation to real-world testing.

1.3 Contributions Described in this Thesis

The research described in this thesis motivates, develops and analyzes a new control-law implementation. This research consists of several components. The notion of changing the control law output from rudder control into waypoints is motivated and developed. The new control-law implementation is incorporated into an existing rigid-body simulation environment. This environment also acts as a foundation for simulating communication between boats. Numerous classes and functions are written to simulate and analyze the developed control-law implementation. Other classes are designed to control the flow of data between simulated boats. Once these programs are integrated, the control-law implementation is tuned through experimentation and analysis. Experimental routes are chosen and simulated to determine the success of this control-law implementation. The resulting data is used to tune and then analyze the effectiveness of the resulting control implementation. Finally, formation fidelity is developed as a concept and related to the simulation results.

1.4 Overview of Successive Chapters

Chapters 2 through 4 address three of the major components that exist as a part of the simulated testing environment. The control laws are covered in Chapter 2. The method of introducing communication error into the simulation is described in Chapter 3. The rigid-body dynamics equations used to simulate the vehicles in formation are described in Chapter 4.

The next group of chapters describe the development and testing of the control-law implementation. In Chapter 5, the basic control-law implementation is described, as it has been coded into the existing simulation environment. Chapter 6 describes how parameters have to be tuned in order to get the best results from the newly developed control-law implementation. The evaluation of the control-law implementation through varying levels of communication between vehicles is discussed in Chapter 7. Finally conculsions are drawn in Chapter 8 based on the results of the testing and tuning of the control-law implementation.

Chapter 2

Gyroscopic Control Laws

The United States Navy has an interest in controlling a group of small autonomous boats which move a constant speed. Control laws that can be categorized as gyroscopic control laws are of particular interest for controlling small boats. These control laws govern vehicles controlled solely through steering and maintain a constant speed. In addition, some gyroscopic control laws are an implementation of geometric theory which is used to design and analyze nonlinear feedback systems (Wang and Krishanprasad [1992]). This is particularly useful because of the nonlinear nature of the control of real-world boats. There are several forms of gyroscopic control that have been developed. One specific set of control laws developed by Zhang and Leonard [2007] have been demonstrated to stabilize any number of vehicles into a holding pattern. While these control laws are relevant to controlling unmanned vehicles, it is uncertain as to how dependent the control laws are on initial conditions and under certain conditions collision avoidance is not always possible. Another set of control laws proposed by Justh and Krishnaprasad [2003] are of particular interest to this thesis research because of the successful implementations in other projects controlling unmanned autonomous vehicle formations (Zhang et al. [2004]). After examining the available control laws the ones developed by Justh and Krisnaprasad seemed to suit our needs the best.

The applicability of Justh and Krishnaprasad's control law led to its use in the control-law implementation. These control laws also fall into the category of control laws that implement waypoint driven formation control. These control laws are specifically formulated for point-vehicles. Simplified kinetic and kinematic properties are used to describe point-vehicles. In addition, these vehicles move with constant speed and are constrained to move in a single plane. Each point-vehicle is controlled solely through changing its heading. The resulting vehicles are actuated through a single degree of freedom, and have three degrees of freedom in which they move. This creates an under-actuated system. However, these control laws are still able to direct each point-vehicle

towards an equilibrium within the vehicle formation. This remains true given almost any initial position and heading.

In fact, these point vehicles are similar in configuration to certain kinds of watercraft. Just as the point-vehicles directly control their heading, many boats are controlled through a single rudder to adjust their heading. While the point-vehicles do not have the inertia that a boat would, it is easy to approximate inertia by limiting the degree to which the heading can be changed over a unit of time. As a result, it is a good assumption that these point-vehicles behave in ways similar to rigid-body boats. Because of this correspondence with real-world boat control, and their rigorously demonstrated stability properties, the control laws developed by Justh and Krishnaprasad are a focus of this thesis.

Justh and Krishnaprasad [2003] describe the configuration of n point-vehicles using a special notation which we describe with respect to a fixed origin. The non-dimensional displacement of the jth vehicle is described by the vector $\mathbf{r}_j$. Nonzero values of a scalar variable u_j change the rate of heading change of the jth vehicle. The governing equations for these point-vehicles are shown in Equation 2.1. In order to simulate the evolution of the system of point-vehicles the heading angle θ_j and the position $\mathbf{r}_j$ are numerically integrated. This method of stepping the point-vehicles forward in time is implemented for any point-vehicle simulation mentioned in this thesis.

$$\dot{\mathbf{r}}_j = \begin{pmatrix} \cos\theta_j \\ \sin\theta_j \end{pmatrix}, \dot{\theta}_j = u_j, j = 1, 2..., n \tag{2.1}$$

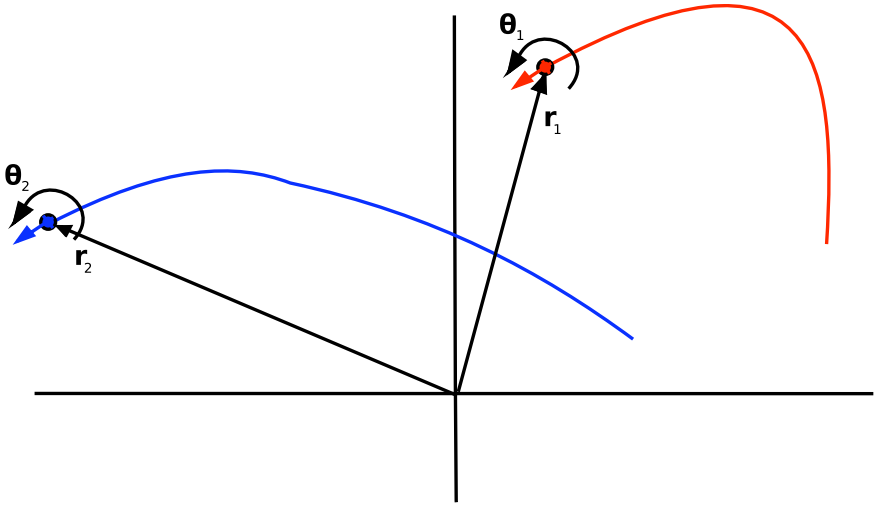
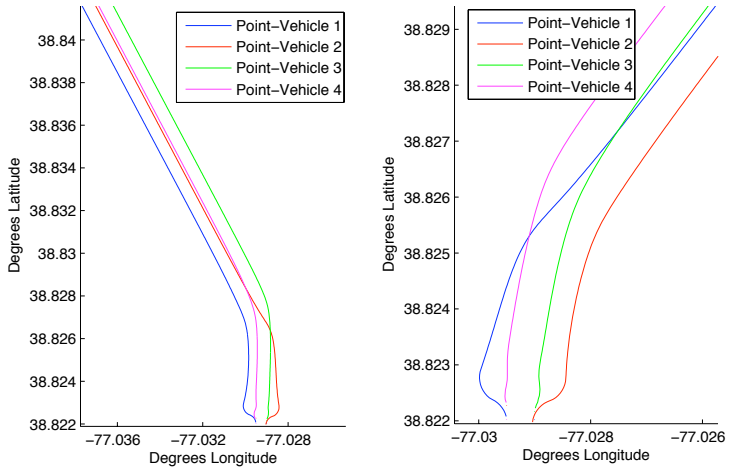
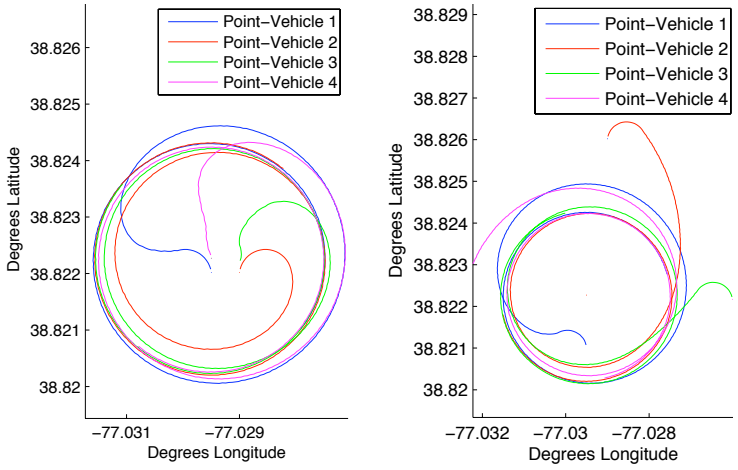


Figure 2.1: Particle Trajectories for Two Point-Vehicles



(a) (b)

Figure 2.2: Rectilinear Control Law for Two Sets ((a) and (b)) of Initial Conditions



(a) (b)

Figure 2.3: Circular Control Law for Two Sets ((a) and (b)) of Initial Conditions

The choice of u_j depends on the on the desired control behavior. In order to move the formation along a straight path, the rectilinear control law is implemented, given by

$$u_j^{rect} = \frac{1}{n}\sum_{k\neq j}\left((-\eta)\overbrace{\left(\frac{\mathbf{r}_{jk}}{|\mathbf{r}_{jk}|}\cdot\mathbf{x}_j\right)\left(\frac{\mathbf{r}_{jk}}{|\mathbf{r}_{jk}|}\cdot\mathbf{y}_j\right)}^{f(\mathbf{r},\mathbf{x},\mathbf{y})} - \alpha\overbrace{\left(1-\left(\frac{r_0}{|\mathbf{r}_{jk}|}\right)^2\right)\left(\frac{\mathbf{r}_{jk}}{|\mathbf{r}_{jk}|}\cdot\mathbf{y}_j\right)}^{g(r_0,\mathbf{r},\mathbf{x},\mathbf{y})} + \mu\overbrace{\mathbf{x}_k\cdot\mathbf{y}_j}^{h(\mathbf{r},\mathbf{x},\mathbf{y})}\right) \quad (2.2)$$

Alternatively, if it is desirable for the formation to circle a point then the u_j is calculated as,

$$u_j^{circ} = \frac{1}{n}\sum_{k\neq j}\left(-\eta\overbrace{\left(\frac{\mathbf{r}_{jk}}{|\mathbf{r}_{jk}|}\cdot\mathbf{x}_j\right)}^{f(\mathbf{r},\mathbf{x},\mathbf{y})} - \alpha\overbrace{\left(1-\left(\frac{r_0}{|\mathbf{r}_{jk}|}\right)^2\right)\left(\frac{\mathbf{r}_{jk}}{|\mathbf{r}_{jk}|}\cdot\mathbf{x}_j\right)}^{g(r_0,\mathbf{r},\mathbf{x},\mathbf{y})}\right) \tag{2.3}$$

The parameter n corresponds to the number of vehicles in Equations 2.2 and 2.3. There are several vectors which are calculated before either of the control calculations is performed. Equations 2.4, 2.5, and 2.6 are substituted into the control calculations.

$$\mathbf{r}_{jk} = \mathbf{r}_j - \mathbf{r}_k \tag{2.4}$$

$$\mathbf{x}_j = \begin{bmatrix}\cos\theta_j \\ \sin\theta_j\end{bmatrix} \tag{2.5}$$

$$\mathbf{y}_j = \begin{bmatrix}-\sin\theta_j \\ \cos\theta_j\end{bmatrix} \tag{2.6}$$

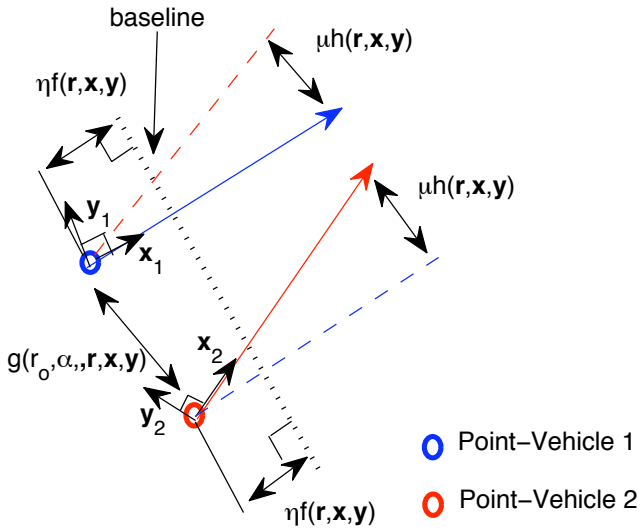


Figure 2.4: Equation Term Decomposition: terms are shown near the distances/angles they influence

The terms of these equations in Figure 2.4 are the terms used in Equations 2.2 and 2.3. These

terms require the positions and headings of the entire formation to calculate each u_j value. Both control laws are proven to stabilize a formation of two point-vehicles from almost any set of initial conditions (Justh and Krishnaprasad [2003]). The rectilinear control law in Equation 2.2 groups the vehicles together so that they move as a swarm in an arbitrary direction. If one of the vehicles is externally controlled, then the formation will follow this vehicle. Otherwise the swarm follows a repeatable, yet arbitrary trajectory. The circular control law in Equation 2.3 moves the vehicles into a circular holding pattern. To illustrate the control laws Figures 2.2 and 2.3 show the paths of a vehicle swarm with random initial positions using rectilinear and circular control laws respectively.

The μ parameter helps align the heading of the vehicles. The current heading of the jth vehicle is represented by the colored arrows. The red and blue dashed lines show heading of the swarm as determined by the blue and red vehicles respectively. The α and r_0 parameters allow the vehicles to position themselves an appropriate distance from each other. If the vehicles are too far apart, they steer towards each other. If the vehicles are too close together they steer away from each other. Finally, the η parameter drives the vehicles to stabilize their formation such that the vehicles are forced closer to the "baseline" of the swarm. The baseline is the line perpendicular to the mean heading of the swarm and intersects the bow of the furthest forward boat. This "baseline" is represented by the black dashed line.

When the point-vehicles move into a circular formation, the η term also spaces the point-vehicles at opposite ends of the circle. The α and r_0 parameters still control the spacing between vehicles. The μ term does not exist in the circular control law because the vehicles in the swarm do not need to align their headings. Even though the parameters are described using the same parameter symbols in Equations 2.2 and 2.3 they may not be the same for both rectilinear and circular control. However, the similarity between the two depends on the nature of the vehicles being controlled.

2.1 Important Control Law Aspects

While these control laws have been developed and tested (Zhang et al. [2004]), the control laws developed by Justh and Krishnaprasad [2003] have not been made robust to less-than-perfect communication between vehicles. As described in (Justh and Krishnaprasad [2003]), these control laws have many advantages when implemented with the idealized point-vehicles described in Equation 2.1 under perfect communication conditions. The formation can accomodate any number of vehicles, the vehicle formation should always stabilize, and the formation remains deterministic. In a deterministic world the same set of initial conditions invariably yields the same outcome. In

this case, when the vehicles have the same initial positions and headings, they will have the same trajectories. It is also important to stress that a stable formation does not imply a high fidelity one. A *stable formation* control law will force the vehicles to reach an equilibrium state where they are a specified distance apart and heading in the same direction. A high fidelity control-law implementation will mirror the trajectories of the control law it is implementing. For example, if all the vehicles in a given formation were to disperse or collide, this problem could be caused by low fidelity to a stable formation control law, or to some degree of fidelity to a poorly designed control law. While in some circumstances fidelity and stability may seem similar, there is a subtle difference between these two concepts. Since the control laws we use are stable, any apparent instability we observe must be due to the implementation lacking fidelity. We can adjust the implementation to improve fidelity, but this does not alter the stability properties of the underlying control laws.

2.2 Swarm Waypoints

Another useful aspect of the Justh-Krishnaprasad control laws is that the swarm can be guided to waypoints by creating an imaginary vehicle in the swarm affixed at the desired location of the waypoint. This waypoint-vehicle is different from the other vehicles in several important ways. The waypoint-vehicle has zero velocity and Equations 2.2 and 2.3 are not performed for the waypoint-vehicle during the simulation. In addition the waypoint is attributed a higher weight, replacing the $\frac{1}{n}$ value with with a higher value when other vehicles consider it in their control calculations. This change forces the point-vehicles to follow the waypoint more aggressively than the other vehicles. Finally the waypoint-vehicle is oriented to the average of the heading required by each vehicle to point towards the waypoint-vehicle. Once the vehicles reach the waypoint, the waypoint-vehicle can be taken out of the control calculation and a new waypoint-vehicle can be put in its place. This method allows the formation to follow a series of waypoints.

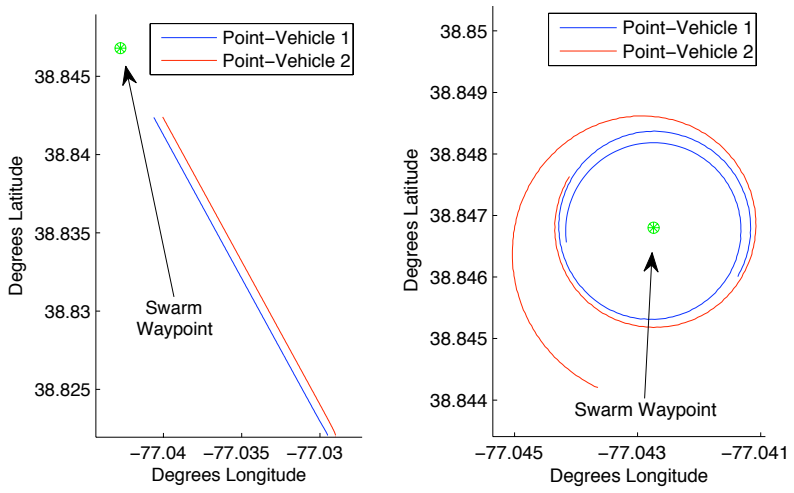


Figure 2.5: (a) Boats Moving Towards Swarm Waypoint Using Rectilinear Control Law. (b) Boats Circling a Swarm Waypoint Using Circular Control Law.

2.3 Swarm Waypoint Following with Control Mixing

The simulation starts by calculating u_j. The rectilinear and circular control laws are used to compute a rectilinear and circular control option. These two scalars are combined to update the heading of each vehicle. When the vehicles are far away from a waypoint, only the rectilinear control is used. When the vehicles are close to a waypoint, they use the circular control. When the vehicles are in between these two phases, they use a combination of both controls. As the vehicles get closer to a waypoint, the control transfers from being dominated by rectilinear control to the circular control using Equation 2.7.

$$u_j = u_j^{circ}\epsilon + u_j^{rect}(1-\epsilon) \tag{2.7}$$

The function $\epsilon(t)$ is a percentage and a function of time. If the formation is transitioning from rectilinear from to circular control, the initial time is set to zero. If the formation is transitioning from the circular to rectilinear control, the initial time is set to one. This ϵ value included in Equation 2.7 is the percentage of that duration value. When the control law is required to transition from rectilinear to circular control, $\epsilon(t)$ increases with time. When transitioning from circular to rectilinear control $\epsilon(t)$ decreases with time. When the control law circles a swarm waypoint, there is a finite duration associated with that waypoint. As time steps forward, this duration is counted down to zero. Once zero is reached the next waypoint is assigned.

This mixed control increases the number of control possibilities from two to four. In order for the formation to operate using different control options , the vehicles use a process to determine when the vehicles change control options.

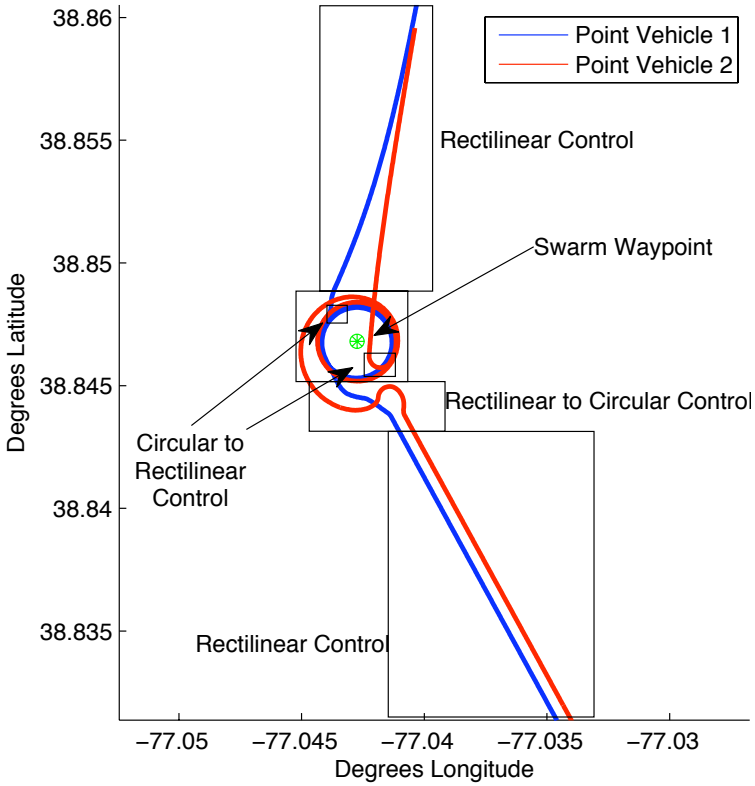


Figure 2.6: Point-Vehicles on Full Course

The vehicles are able to determine when to use the mixed rectilinear to circular control using Equations 2.8 and 2.9.

$$\mathbf{r}_T = \frac{1}{n}\sum_{j=1}^{n}\mathbf{r}_j \tag{2.8}$$

$$d_{Arr} \geq |\mathbf{r}_w - \mathbf{r}_T| \tag{2.9}$$

Equation 2.8 determines the center of the formation ($\mathbf{r}_T$). Even though the waypoint-vehicle is part of the swarm for the control calculation, it is not considered part of the formation in (2.8). This value is used to determine how far away the formation is from the waypoint the swarm is following. This value is compared with a chosen waypoint arrival distance (d_{Arr}), determined in Equation 2.9. The duration of the circling control is used is determind by a timer. After the rectilinear to circular control is finished, a new timer is started with a prescribed circling time.

When this value is counted down to zero, the circular to rectilinear control is started. If there is not another waypoint to follow, the swarm will circle the last waypoint indefinitely.

Once the point-vehicle simulation, steering control laws, and control law mixing are combined, a full path can be generated as seen in Figure 2.6. In this example, the vehicles start at the lower right corner using the rectilinear control law. The stationary point-vehicle at the initial swarm waypoint pulls the other two vehicles towards it. Once one of the point-vehicles is within the swarm waypoint radius, the point-vehicles transition to the circling control law. After the point-vehicles have circled the waypoint for a specified amount of time a new swarm waypoint is assigned. In Figure 2.6 the vehicles move to a second waypoint that is outside of the plot boundaries. The point-vehicles transition from circling the first waypoint to following the second waypoint, as described by Equation 2.7.

2.4 Swarm Control with Poor Communication

Controlling a formation of vehicles through various communication situations is a problem that has been examined by other researchers as well. Theoretical work performed by Sepulchre et al. [2007] shows that control laws such as those proposed by Justh and Krishnaprasad [2003] "have almost global convergence properties". However, this does not address how to make formation control laws more robust to poor communication. In multiagent rendezvous problems proposed by Cortés et al. [2006], agreement between vehicles is achieved even in spite of communication failures between individual vehicles. While Cortés et al. [2006] discuss complex communication networks they do not address how to deal with poor communication between all of the vehicles in the formation. Even though there is research to make formation control laws more robust to certain communication situations, periodic communication between vehicles is still a valid problem for a formation of vehicles implementing a control law proposed by Justh and Krishnaprasad [2003]. As a result, this thesis research seeks to achieve resistance to real-world conditions by implementing the control law in a different way.

When poor communication is introduced into the point-vehicle simulation there is a noticable loss of fidelity to the path that Figure 2.6 lays out. Poor communication in Figure 2.7 is simulated as an interval between communication. The choice of communication error simulation is further explained Chapter 3. While the point-vehicles are initially able to follow the swarm waypoint, they are unable to effectively circle it. This inability of the point-vehicles to follow the swarm waypoint is problematic even for a two vehicle formation. However the overlap of paths does not necessarily

indicate that the vehicles collide with each other. The vehicles may cross paths at different points in time. The low fidelity of the formation to the original path, however, indicates that the control law is no longer being followed. As a result the likelihood of collision increases.

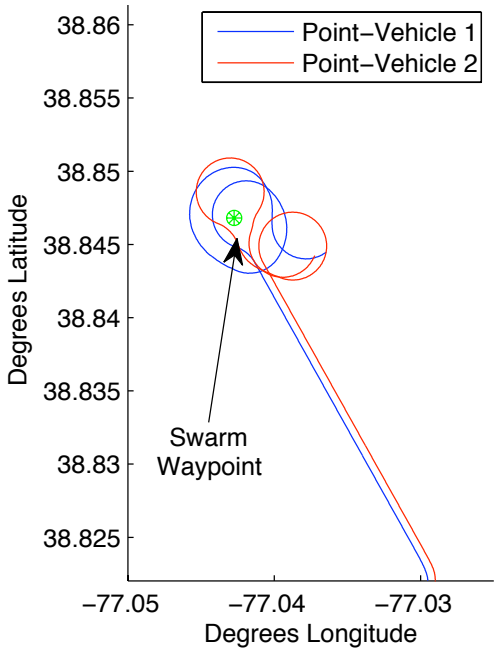


Figure 2.7: Point-Vehicle Simulation with Poor Communication

Chapter 3

Communication Error in Swarm Control

3.1 Inter-Vehicle Communication

A formation of vehicles can be leaderless, leader-follower, or exist as a more complex configuration. A leader-follower formation exists when every vehicle follows a single leader vehicle. The leader vehicle exists physically, as opposed to a waypoint-vehicle which does not. While the leader follower configuration may be easier to implement in a network, it suffers from a number of inherent problems. First, if the leader is disabled, the entire formation destabilizes because the vehicles have nothing to follow. Second, leader-follower systems respond very poorly to communication problems, as shown by Fax and Murray [2002]. In the simulation described in later chapters, the formation is a leaderless waypoint-driven formation.

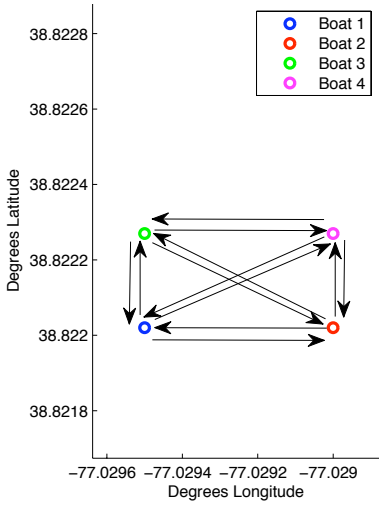


Figure 3.1: Complete Graph

There are a number of ways to approach communication problems in this situation. One popular

way to address communication problems is to examine how well each vehicle communicates with each of the other vehicles. This question of *consensus* is addressed in Fax and Murray [2002] and Moreau [2005]. Consensus is achieved when each vehicle receives accurate information in order to compute its subsequent control. This view of communication is understood using graph theory. Both Fax and Murray [2002] and Moreau [2005] describe each vehicle as a node and the communication link from one vehicle to another is considered a directed vertex. The control laws stated by Justh and Krishnaprasad [2003] form a *complete graph* seen in Figure 3.1 as defined by Fax and Murray [2002]. In this idealized state every vehicle has bidirectional communication with every other vehicle. In a complete graph situation, stabilizing the vehicles is only a matter of stabilizing each individual vehicle. However, this assumption is true for only fixed time delays in communication (Fax and Murray [2002]). In this thesis, bidirectional communication is considered to exist when information is received and sent by each vehicle.

Another notion developed by Fax and Murray [2002] and Moreau [2005] is that of a centroid. The *centroid* or *formation center* is an imaginary vehicle that is created through the control laws implemented by each vehicle. Each vehicle follows this imaginary vehicle as if it were the physical leader. When each vehicle has the same centroid, the entire formation appears stable; however, this stability is not proven for all control laws. As vehicles have an increasing discrepancy in their relative centroids, the entire formation becomes less stable. This discrepancy results in a reduction in the probability that the vehicles will converge to a specific distance without encountering problems. While a formation may have varying degrees of communication problems, poor communication is only a problem if the vehicles have different formation centers. *Convexity* is another way to look at the stability of a formation controlled system (Moreau [2005]). Convexity is a notion introduced to multi-agent system communication by Moreau. The notion of convexity is established using a combination of system and graph theory tools. Convexity results from research showing that when a directed graph is arranged a certain way, there is the possibility for instability in the system. For example, if a vehicle receives a data packet that is from a different time step, this data reception could destabilize the formation. The results of Moreau's study show that bidirectional communication between each vehicle for every time interval will allow for the vehicles to converge to a certain value. However, it is also shown that in certain situations more information exchanged between vehicles can cause a loss of convergence. We suspect that this loss of convergence is very similar to a decrease in formation fidelity.

3.2 Types of Interference

When developing a communication model, it is also important to understand the sources of communication error. According to Takai et al. [2001] and Zang et al. [2005] there are a number of potential sources involving communication error for vehicles that communicate using a wireless interface. Different scenarios exist for *line of sight* (LoS) and *no line of sight* (NLoS) situations. In the small vehicle formation scenario described in this thesis, the NLoS scenario should be considered. The vehicles will be situated at various distances from each other with the possibility of obstacles in between the antennas. It would seem that the greatest source of communication loss is due to distance or multipath interference. While a distance-dependent attenuation model for multipath is easy to understand and simulate, other multipath effects are more difficult to assess.

Multipath interference is a problem caused by a signal that is sent or received after reflection off of adjacent objects. This interference can cause fading as described by Arrendondo et al. [1973]. This fading is caused when two different waves interfere with each other in either a destructive way. This fading is evaluated in Zang et al. [2005] as *signal to interference and noise ratio* (SINR), which is described by Equation 3.1.

$$SINR = \frac{C}{\sum (I + N)} \tag{3.1}$$

The value C is the signal level expected and the terms I and N inside the summation are due to the different signals causing the interference. Interference can come from background noise or from other packets of information that are not intended to be received. In this model, once the cumulative noise level exceeds the signal strength, the received packet is dropped. A dropped packet is the loss of a unit of information needed for the control calculation. The result is either a loss or delay in communication depending on how the system deals with a dropped packet.

3.3 Research to Improve Formation Control During Poor Communication

One paper has addressed how to compensate for some degree of communication error by altering the formation control laws (Pongpunwattana et al. [2007]). The control law examined by Pongpunwattana et al. [2007] was originally proposed by Klein and Morgansen [2006]. While the control laws are not the same as the ones used by Justh and Krishnaprasad [2003], they are similar in

nature in that the control law is broken up into a term that matches the velocity of the group, a term that moves the vehicle towards the centroid, and a term that regulates the distance between each vehicle. Communication error is modeled as a delay destabilizing the original control laws. Changes to the vehicle spacing term of the control law helped to add a degree of robustness to the control laws.

While the control laws used by Pongpunwattana et al. [2007] differ in design to some degree from the Justh-Krishnaprasad control laws, the equivalent coefficient r_0 in Equations 2.2 and 2.3 could be made more complex to account for communication problems. This advocates careful tuning of the Justh-Krishnaprasad control law parameters in order to obtain the best possible formation control.

3.4 Relevance to Chosen Control Laws

Several different ideas about the effects of communication problems on formation control have been discussed in this chapter. Graph theory and notions of convexity suggest that it is prudent to attempt communication between the entire formation in the same time step. Since the control laws stated in Equations 2.2 and 2.3 use a complete graph, the formation should be stabilized with poor communication as long as vehicles use information from the same time step. When vehicles in other communication configurations were destabilized, it was because current state information arrived at later time steps as it was passed around. Since a data transmission between boats includes the time of data transmission, data can be organized and dropped if it is from the wrong time step. Data is transferred between vehicles until the entire group agrees on where all of the vehicles were located at a specific time. Multipath fading and interference can cause two different problems for communication between boats. This fading can cause the distortion or loss of a data packet. Since distortion of a data packet can be detected with checksums, it is appropriate to assume that the loss of data will result from poor or distorted communication. This means that poor communication will result in less data transfered between boats which is comparable to the signal to noise ratio described by Zang et al. [2005]. Signal to noise ratio can be implemented as a percentage of randomly timed good communications or as an interval between communications. While it is possible to randomly allow communication between vehicles to simulate poor communication, this causes problems with analyzing the data. If the vehicles randomly communicate with varying degrees along a prescribed course and there is unusual behavior, it will be difficult to determine the cause of this behavior. Therefore it is appropriate to model poor communication as a periodic

interval of successful communications. The severity of the communication problems is indicated by the length of this interval. The details of how this communication interval is implemented is discussed in Chapter 5.

Determining how to mitigate the effects of poor communication is more challenging. It would seem that changing the separation terms in the control laws stated in Equations 2.2 and 2.3 would help make their implementation more resistant to communication problems. While this change does reduce the probability of collision, it does not help the vehicles track waypoints more effectively. In Figure 2.7 the increased possibility of collision is caused by poor waypoint circling control, not by an overly tight formation. Consequently, better tuned parameters will not fix some of the larger problems caused by poor communication.

Chapter 4

Rigid-Body Dynamics Simulation

4.1 Existing Simulation Environment

In order to create a framework that can adequately capture the desired complexity of real-world phenomena, a fairly rigorous software approach is required (Cortesi et al. [2006]). The rigid-body simulation and display software developed by the United States Navy was an appropriate choice for modeling this environment. The simulation environment is written in Java with classes extending those of Cybele Pro. Cybele Pro is a third party application that is available for implementing an agent-based framework in Java.

Cybele Pro and Java were chosen because scaling the simulation environment to a greater number of vehicles is facilitated by these frameworks. The control laws stated in Equations 2.2 and 2.3 have been demonstrated to stabilize n vehicles. Thus, the simulation allowed for testing for n vehicles. Java is also conducive to input from multiple developers. This combination of factors made Java the best choice.

Finally Cybele Pro allowed the simulation to accurately keep track of the vast amount of information that is being manipulated throughout the duration of the simulation. It was important to make sure that data from one time step was not used in the calculation of another time step. It was also critical that the flow of information from one vehicle to another was controlled. This facilitated the simulation of communication error; Cybele Pro allowed this strict data management.

4.1.1 Code Execution

Cybele Pro creates an environment within a "container" at the start of every simulation. In the simulated environment, the container class contains all of the other classes that simulate the physical world in which the rigid-body boats exist. This world includes the boats, autopilots controlling the

boats, and communications between these simulated boats.

Within each container there are agent classes. Agents can be any number of independent entities. In the rigid-body simulation environment, boats are agents. The observers of these simulations are also agents. Agents are created by the container or by other agents. Agents communicate by sending and receiving messages. The programmer is able to choose if and how often each agent receives information from other agents. For example, the boats can broadcast their positions and headings. However only the agents which are told to listen to the specific broadcasting agent ever receive that information. If information is not received, then old or default information is used instead.

Activity classes in the rigid-body simulation environment can be thought of as components of the agent in question. For example each boat agent has an activity which recieves information from the other boats and uses it to calculate the desired waypoint to which the autopilot should head. Therefore this activity acts as the communication component. Once initialized, activities execute at a specific frequency.

4.1.2 Pre-existing Code

A significant portion of the rigid-body simulation environment was already in place even before this research had started. Roger Cortesi along with Eric Justh, Kevin Galloway, and Wil Selby at the Naval Research Laboratory had already begun different avenues of research with this code. At the beginning of this research effort the code was designed to use rigid-body dynamics to characterize the motion of various rudder controlled, constant speed vehicles.

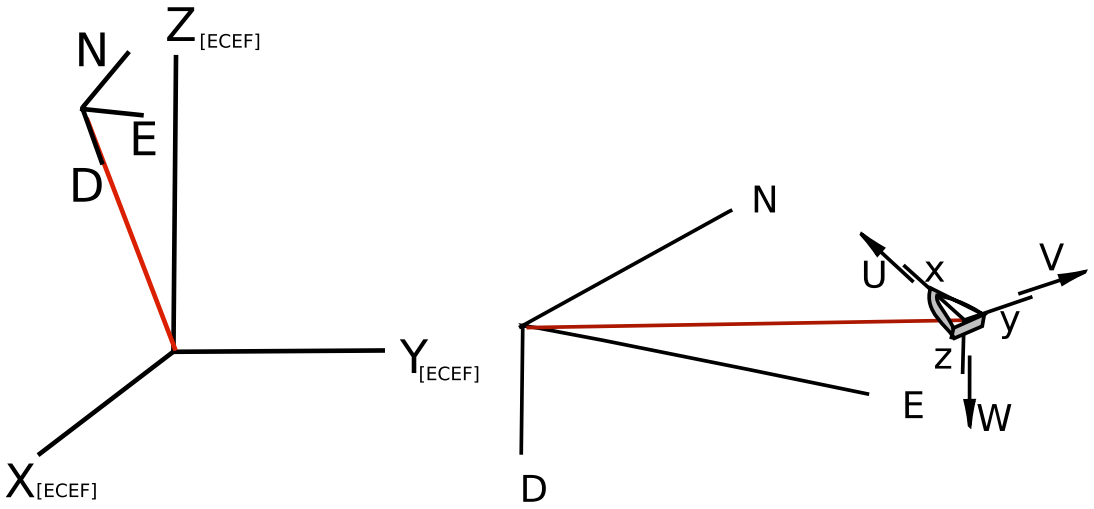
At the start of every simulation, two configuration files are accessed in order to create a simulation container. These files determine which and how many agents are going to run. These files give each created boat agent its initial location, meta-centric heights, drag coefficients, and its moments of inertia. These files also determine how the results of the simulation will be displayed through the creation of different agents.

When a vehicle agent is created, each agent creates its own instance of all the classes that it will access. Although two agents may access the same classes or methods, the data used and generated by that method is not shared between the two agents because, the execution of the method is unique to each instance of the agent class. Once the simulation is started, the container waits for all of the agents to be initialized. The simulation then moves forward in time. After the simulation is finished, the trajectory data can be accessed and analyzed.

4.2 Rigid-Body Simulation

While there are many calculations that the simulation performs, one of the more complex calculations is that of the rigid-body dynamics simulation. The existing program uses a combination of Newton's Second Law and Euler's laws for rigid-body rotation. Once each of these calculations is performed the software uses the Improved Euler method to numerically integrate forward in time. Finally the values are converted into different coordinate systems for the display software. The program performs these time stepping calculations with three vectors and a quaternion.

The first vector is the [**ECEF**] vector which is in the earth centered, earth fixed frame. It describes the position of the boat relative to the earth. The [**UVW**] vector describes the inertial velocity of the boat written in terms of the body-fixed coordinate system. These two coordinate



(a) (b)

Figure 4.1: (a) [ECEF] to [NED] Coordinate Transformation (b) [NEC] to [UVW] Coordinate Transformation

systems are related through the north-east-down coordinate system as shown in Figure 4.1. The north-east-down coordinate frame was chosen because it is commonly used by autopilots and approximates the surface of the earth as a plane.

Equation 4.1 describes the translational acceleration that the vehicle experiences using Newton's second law,

$$[\mathbf{U\dot{V}W}] = [\mathbf{g}] + \frac{1}{\mathrm{m}}[\mathbf{F}] \tag{4.1}$$

Where [**UVW**] are velocities of the body-fixed frame with respect to the north-east-down frame.

The value "m" is the mass of the boat. The acceleration of gravity is represented by the $\mathbf{g}$ vector. In the north-east-down coordinate frame gravity is positive. In addition, it is assumed that gravity only acts in the z-direction of the body-fixed frame shown in Figure 4.1 (b). Since the boat will only experience minor pitches and rolls, this is a good assumption. However, this force balance shows that the speed is not always constant. In order to meet the conditions required by the Justh-Krishnaprasad control laws, it is important to maintain constant speed. Luckily, the speed only varies for the brief period when the simulation first starts, although the boats remain near the same speed for the rest of the simulation. If a constant speed is not maintained, vehicles with crossing paths may collide. In this simulated environment there is no wind or any other force that would push the boat off course, so constant speed remains a good assumption. The forces that factor into this force balance are summed below.

$$[\mathbf{F}] = \mathbf{F}_{\text{Buoyancy}} + \mathbf{F}_{\text{Drag}} + \mathbf{F}_{\text{Propulsion}} \tag{4.2}$$

It is assumed that the boat floats on a flat, calm surface. The acceleration of the boat is determined by several forces as shown in Equation 4.2. The buoyancy force is given by,

$$\mathbf{F}_{\text{Buoyancy}} = \begin{bmatrix} 0 \\ 0 \\ -F_{Buoyancy-z} \end{bmatrix}, \mathrm{F}_{\text{Buoyancy-z}} = \begin{cases} 0 & \text{if } \rho(d_h - A)Lwg \leq 0 \\ \rho(d_h - A)Lwg & \text{if } 2mg > (d_h - A)Lw \\ 2mg & \text{if } \rho(d_h - A)Lwg \geq 2mg \end{cases} \tag{4.3}$$

The buoyancy force in Equation 4.3 is also assumed to always act in the z-direction for the same reasons as gravity. The force acting in the z-direction of the body-fixed frame is a product of the density (ρ), submerged volume, and the acceleration of gravity (g). The submerged volume is calculated using the length (L), width (w), and submerged depth ($d_h - A$). This last term uses the altitude (A) and the draft of the boat (d_h) to determine the submerged depth. These values are used to calculate a force which is part of a larger piecewise function which gives the buoyancy force maximum and minimum values over a certain range.

The drag in Equation 4.4 is assumed to be linearly related to the magnitude of the velocity.

$$\mathbf{F}_{\text{Drag}} = \begin{bmatrix} -C_{d-x}U \\ -C_{d-y}V \\ -C_{d-z}W \end{bmatrix} \tag{4.4}$$

The C_d values are coefficients of drag. The components U,V, and W are the velocities of the body-fixed frame. This rough estimation of drag is realistic since the boats will be moving at constant speed. Therefore it can be assumed drag is linear around the small range of velocities in which the boat operates.

The forces created by the propulsion systems are calculated as,

$$\mathbf{F}_{\text{Propulsion}} = \begin{bmatrix} \cos(-\beta) & \sin(-\alpha)\sin(-\beta) & \cos(-\alpha)\sin(-\beta) \\ 0 & \cos(-\alpha) & -\sin(\alpha) \\ -\sin(-\beta) & -\sin(-\alpha)\cos(-\beta) & \cos(-\alpha)\cos(-\beta) \end{bmatrix} \begin{bmatrix} \mathbb{T}(A,\theta,T) \\ 0 \\ 0 \end{bmatrix} \tag{4.5}$$

In Equation 4.5 α is the rudder angle, β is the trim angle, and $\mathbb{T}$ is the thrust exerted by the water on the boat. Thrust is a function of altitude (A), the pitch (θ), and the engine torque (T). Altitude and pitch are used to determine whether or not the propellor is submerged. If the propeller is not fully submerged, this will affect the thrust adversely.

Equation 4.6 describes the translational velocity of the vehicles. Rotation matrices are used to convert the body-fixed velocities into earth-centered, earth-fixed velocities. Again [**ECEF**] is the boat position in the earth-centered, earth-fixed frame. The rotation matrix $\left[\mathbf{C_{e/n}}\right]$ transforms coordinates from the north-east-down frame to the (ECEF) frame. The $\left[\mathbf{C_{n/b}}\right]$ matrix transforms the frame from the body-fixed coordinate system to the north-east-down coordinate system.

$$[\mathbf{E\dot{C}EF}] = \left[\mathbf{C_{e/n}}\right]\left[\mathbf{C_{n/b}}\right][\mathbf{UVW}] \tag{4.6}$$

The body-fixed angular rates are the third vector which is integrated. It is described by the $\boldsymbol{\omega}^b$ vector.

$$\boldsymbol{\omega}^b = \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \tag{4.7}$$

The $\boldsymbol{\omega}^b$ vector is broken up into individual components. The component P is the roll rate, Q is the pitch rate, and R is the yaw rate. The rotational acceleration of the boat is calculated as,

$$\begin{aligned}
\dot{P} &= \frac{1}{I_{xx}I_{zz}}\left(\left(I_{zz}\left(I_{yy}-I_{zz}\right)\right)QR+I_{zz}M_x\right) \\
\dot{Q} &= \frac{1}{I_{yy}}\left(\left(I_{zz}-I_{xx}\right)PR+M_y\right) \\
\dot{R} &= \frac{1}{I_{xx}I_{zz}}\left(\left(\left(I_{xx}-I_{yy}\right)I_{xx}\right)PQ+I_{xx}M_z\right)
\end{aligned} \tag{4.8}$$

Equation 4.8 is Euler's equation for rigid-body rotation (Greenwood [1998]). These equations are evaluated from the center of gravity of the boat along the principal axes, where the body fixed frame is located. As a result, the moments of inertia I_{xx}, I_{yy}, and I_{zz} are the only values required to describe the rotational inertia of the rigid-body boat. When the simulation is initialized the values of $\boldsymbol{\omega}^b$ values are zero. During the simulation the moments cause the boat to rotate.

The moments that affect the boat are calculated as,

$$\mathbf{M} = \mathbf{M}_{\text{Buoyancy}} + \mathbf{M}_{\text{Drag}} + \mathbf{M}_{\text{Propulsion}} \tag{4.9}$$

Where buoyancy, drag, and propulsion cause the rotation of the boat as shown in Equation 4.9.

$$\mathbf{M}_{\text{Buoyancy}} = \begin{bmatrix} GM_xF_{Buoyancy-z}\phi \\ GM_yF_{Buoyancy-z}\theta \\ 0 \end{bmatrix}, \tag{4.10}$$

The buoyancy moments in Equation 4.10 also act to stabilize the rotation of the boat. The metacentric heights (GM_x) and (GM_y) are constants that dictate the the rolling and pitching stability of the boat. The magnitude of this stabilizing force is indicated in Equation 4.3. The angle ϕ is the roll and θ is the pitch of the boat.

$$\mathbf{M}_{\text{Drag}} = \begin{bmatrix} -C_{dr-x}P \\ -C_{dr-y}Q \\ -C_{dr-z}R \end{bmatrix} \tag{4.11}$$

The moments due to drag shown in Figure 4.11 are also linearly related by coefficients to the rotation rates of the boat. The (C_{dr}) values are the coefficients of drag due to rotation. The components P, Q, and R are the rotational rates of the boat.

$$\mathbf{M}_{\text{Propulsion}} = [\mathbf{r}_{\text{Body/Prop}}] \times [\mathbf{F}_{\text{Propulsion}}] \tag{4.12}$$

The propulsive moments as shown in Figure 4.12 are a cross product of the vector $\mathbf{r}_{Body/Prop}$ by propulsive force. This force vector is described in Equation 4.5. This vector originates at the center of gravity of the boat and terminates at the propeller.

The boat is steered by the changing the the propulsive moment. The equations also assume that the mass of the boats remains constant throughout the simulation. This means that neither linear nor angular momentum is transfered between the water near the boat and the boat. At the same time there are not any simulated water currents which would perturb the boat's trajectory.

Once the body-fixed angular rates are calculated, they are used to determine the attitude rates. These rates are described by the derivative of the quaternion [**q**] as seen in (4.13) from Stevens and Lewis [2003].

$$\begin{bmatrix} \dot{q_0} \\ \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \end{bmatrix} = \begin{bmatrix} 0 & -P & -Q & -R \\ P & 0 & R & -Q \\ Q & -R & 0 & P \\ R & Q & -P & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{4.13}$$

Once this calculation is performed it is possible to step the [**ECEF**], [**UVW**], $\boldsymbol{\omega}^b$, and [**q**] terms forward in time using the Improved Euler method.

$$y_{n+1} = y_n + hf\left(t_n + \frac{h}{2}, y_n + \frac{h}{2} f\left(t_n, y_n\right)\right), n = 0, 1, 2, \cdots, N-1 \tag{4.14}$$

Equation 4.14 is used to step several values forward in time (Kohler and Johnson [2004]). These values are subtituted in place of y. The value t corresponds to time in the simulation. The integer n indicates which time step Equation 4.14 is operating on. While Improved Euler is only a second-order approximation, the calculations are performed with a very small time step h. It would seem that this low-order integration is admissible since the results converge towards a solution as the time step is decreased.

Once the rigid-body calculations have been performed and the simulation has moved forward in time, the position and heading is converted into the geodetic (WGS 84) coordinate system. The geodetic coordinate system is more commonly known as the latitude and longitude system. This conversion makes hardware autopilot integration possible. The quaternion [**q**] is used to determine

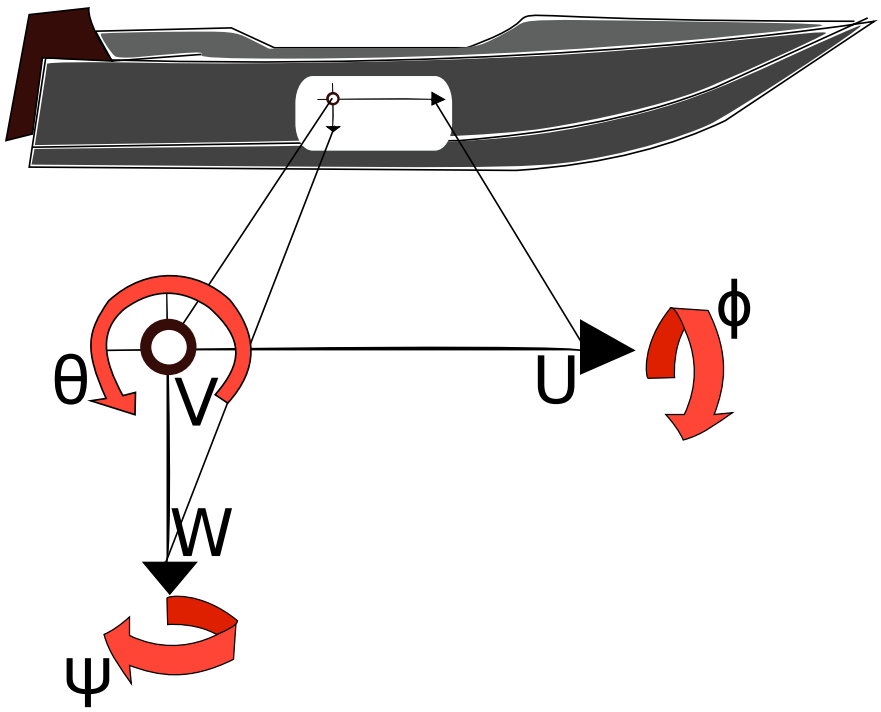


Figure 4.2: Euler Angles and Body Fixed Coordinate System

the Euler angles as seen in Figure 4.2 using Equation 4.15. The body-fixed yaw, pitch, and roll (3-2-1) Euler angles (ψ, θ, ϕ) are defined by Stevens and Lewis [2003].

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \tan^{-1}\left(\frac{2(q_2q_3+q_0q_1)}{q_0^2-q_1^2-q_2^2+q_3^2}\right) \\ \sin^{-1}\left(2(q_1q_3-q_0q_2)\right) \\ \tan^{-1}\left(\frac{2*(q_1q_2+q_0q_3}{q_0^2+q_1^2-q_2^2-q_3^2}\right) \end{bmatrix} \tag{4.15}$$

After this coordinate transformation the control calculation can be performed in order to obtain the rudder angle. Once the rudder is calculated the values are calculated again. This process continues indefinitely until the simulation is stopped.

Chapter 5

Developing a Control Law Implementation

5.1 Initial Setup

5.1.1 Control Implementation Selection

Several control methods were considered when devising how to control the rigid-body boats. The most obvious control method would be to control the rigid-body boats in the same way that point-vehicles are controlled. Each boat could calculate its rudder angle from positions and headings of all the other boats in the swarm. While this may be easy to implement, testing shows it is very susceptible to communication problems and would not work well with a hardware autopilot. A hardware autopilot tries to achieve waypoints, not rudder angles. In addition, whenever the rigid-body boat receives an inaccurate or incomplete information packet, the control law will calculate a yaw rate that will cause the rigid-body boat to go off course.

Another way to control the boat is to try to generate paths for each rigid-body boat using the most recent and complete data from each of the rigid-body boats. This data could be used to compute paths forward using the control law implemented with the point-vehicles. This is the method that was chosen to add resistance to communication error and to aid in autopilot integration.

We submit that computing the paths forward in time reduces the effects of poor communication. It is necessary to investigate how and when the boats will use this path as a means of navigation. Since the hardware autopilot is set up to receive waypoints, it seems natural to control the rigid-body boats through waypoints. This is why waypoints that are sent to an autopilot are called *control waypoints*. This distinction is important because the control law is designed such that the vehicle swarm can move towards and circle a vehicle-waypoint. These waypoints will be called *swarm waypoints*.

5.1.2 Simulating Communication

Simulating communication error can be a complex problem as discussed in Chapter 3. There are many complicated ways to simulate multipath interference and other phenomenon that would cause data inaccuracy. However, most systems would be able to identify inaccuracies caused by communication error and would be designed to drop this information. Therefore communicating in intervals is an appropriate way to model communication error.

In the process of simulating communication error, ensuring that the error is a complete communication drop experienced by each vehicle equally is an important issue to consider. Cybele Pro is a vital component for ensuring this drop in the program. Another issue is which information should be dropped during poor communication. While it is clear that the information fromm other vehicles in the swarm should be dropped, information about the vehicle's own position and heading should be dropped as well. The experimental simulations should be conducted in a way that would allow the control implementation to be scaled from two to any number of vehicles. A vehicle knowing its own position at any given point would jeopardize the scalability of the control-law implementation. If there are two vehicles and each vehicle knows its own location and heading, then each vehicle knows half of the formation's positions and headings. If there are ten vehicles, the assumption that a vehicle knows half the swarm's positions and headings is a very poor assumption since each boat in this case can only know a tenth of this information during a communication drop. Allowing the boats to only know all of the positions and headings at specific time intervals makes it so increasing the number of vehicles in the swarm is still an option.

5.1.3 Setting-up Swarm Course

Creating a testing procedure is an important part of this thesis research. It is always the goal to show the new results of the research while still making meaningful connections to the work that has preceded it. In order to capture the richness and versatility of the control laws, two boats will move through a complex prescribed course. Each of the boats will start relatively near the other. The tests will use only two boats even though the control laws have been shown by Justh and Krishnaprasad [2003] to work for n vehicles. Adding more than two vehicles to the simulation will only make the results more difficult to analyze. Since it is important to control a swarm of boats, there will be a two-waypoint course setup. The boats will travel for a short distance using the rectilinear control law. Once the swarm is a certain distance from the first swarm waypoint, the vehicles transition from the rectilinear control law to the circular control law.

This process is the same as the one as described in Section 2.3. Once the vehicles have spent a prescribed amount of time transitioning control laws, the boats should move into a circling pattern which is maintained for a fixed period of time. Afterwards the boats stop circling the first swarm waypoint and then start moving towards the second swarm navigation waypoint using the circular to rectilinear blended control. The simulation ends while the vehicles use the rectilinear control law to approach the second swarm waypoint. The parameters defined in Equations 2.2 and 2.3 have been scaled beforehand for our system of vehicles and course choice. In order to focus on the control-law implementation, only the parameters introduced by the autopilot and simulated communication are changed throughout these tests.

5.2 Software Development of the Control-Law Implementation

Once the communication error, rigid-body simulation, software autopilot, and control laws were understood, the next step was to combine these components. In order to facilitate the explanation of this combination, several flowcharts are employed. The different types of programing elements are described in Figure 5.1.

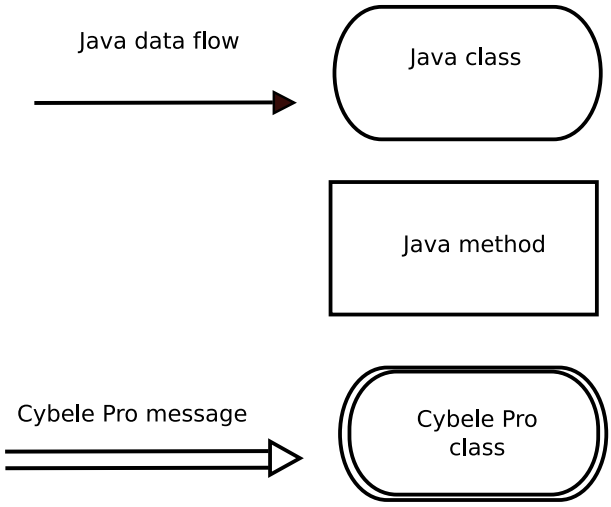


Figure 5.1: Key for the Following Flow Diagrams

The main classes of the control-law implementation are described in this section. This interaction allows the control law, communication simulation, and autopilot to integrate into the rigid-body dynamics environment.

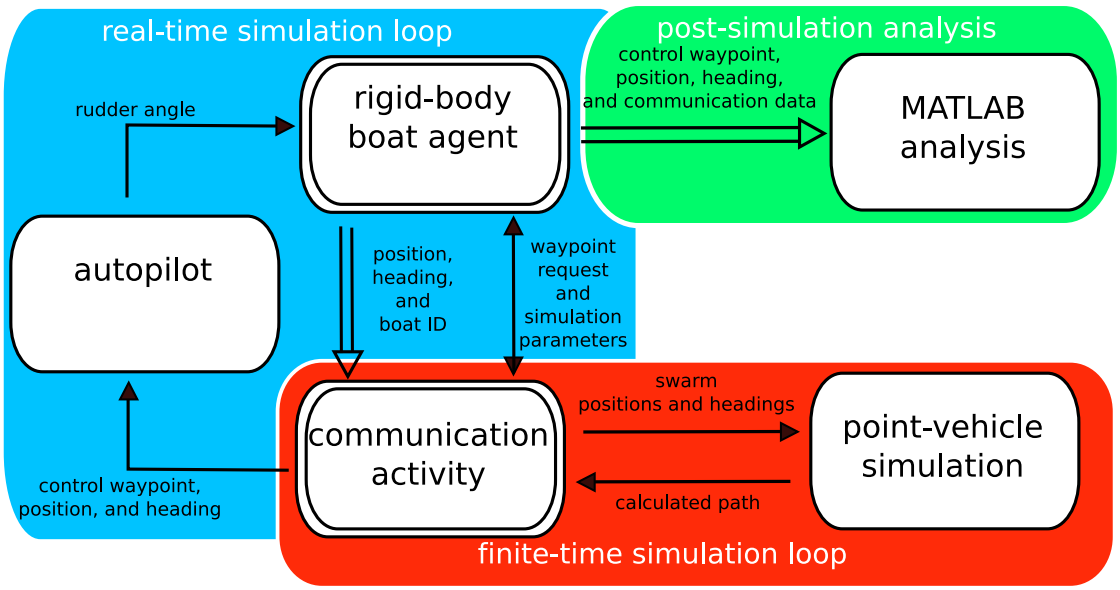


Figure 5.2: Interactions between Control-Law Implementation Programs

Figure 5.2 shows how data is passed between the major classes that determine the trajectory of the rigid-body boat. The real-time simulation loop calculates the values that simulate the boat in the rigid-body dynamics environment. This loop includes the rigid-body boat agent, communication activity, and the autopilot class. In the real-time loop, information is transmitted from the rigid-body boat agent to the communication activity. The communication activity collects the communicated information from all existing rigid-body boat agents along with navigation data from the boat for which it is collecting data.

At specified times the communication activity pauses the real-time simulation loop and starts the finite-time simulation loop. In this loop the communication activity requests a point-vehicle simulation in order to use the control law to determine a new path. This allows the communication activity to determine a corresponding waypoint it should target. The finite-time loop only takes a fraction of a second. The small calculation time allows us to assume that the boat would not have moved during the simulation. This assumption allows us to use the same simulation that was used to show the characteristics of the Justh-Krishnaprasad control law in Chapter 2. Afterwards, the control waypoint is sent to the autopilot class which, in turn, determines the rudder angle. Finally, this rudder angle is sent to the rigid-body boat agent where it is used to determine the new position and heading. Finally, after the simulation ends, the data sent to MATLAB where it can be analyzed and plotted. This analysis is described in Chapters 6 and 7.

5.2.1 Rigid-Body Boat Integration

The rigid-body boat agent is also an extension of the pre-existing rigid-body dynamics simulation. This means that while the class performs all of the rigid body dynamics calculations, it also performs the new functions which are described in this section. As a result this class exists in place of a real boat but also in place of the interphases that connect the other equipment controlling the boat. This means the agent also manages control waypoint requests, and stores the control parameters. While the rigid-body boat agent is constantly saving its course data to a MATLAB program, it is also managing the flow of data between classes.

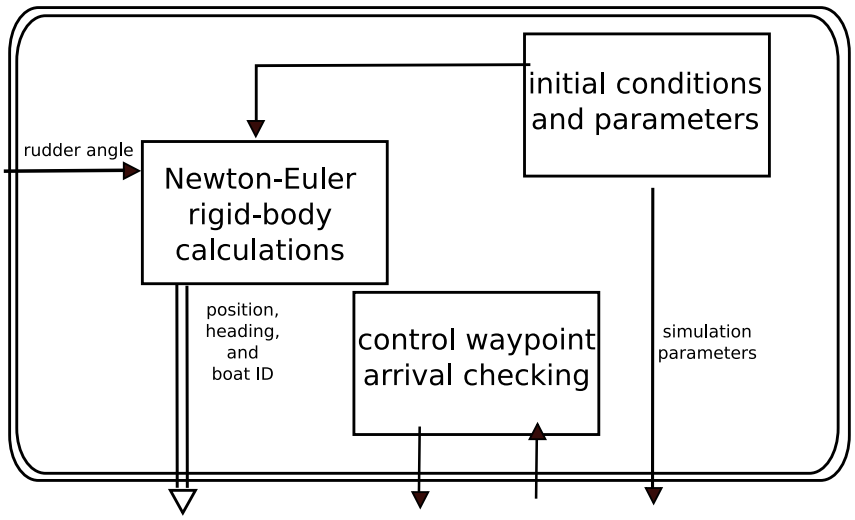


Figure 5.3: Rigid-Body Boat Agent Flowchart

5.2.2 Autopilot Integration

The autopilot class takes the place of a hardware autopilot. Depending on the capabilities of a real autopilot, a real onboard computer might use an instance of this class to calculate the bearing and distance to a waypoint. The autopilot program also takes the place of the hardware autopilot in the simulations. After examining a Simrad autopilot, various methods were written in order to mimic hardware autopilots. The autopilot calculates the bearing to the next control waypoint and then uses that number to determine the ideal rudder angle as seen in Figure 5.4. This ideal rudder angle is compared with the actual rudder angle to determine an appropriate rudder-angle command.

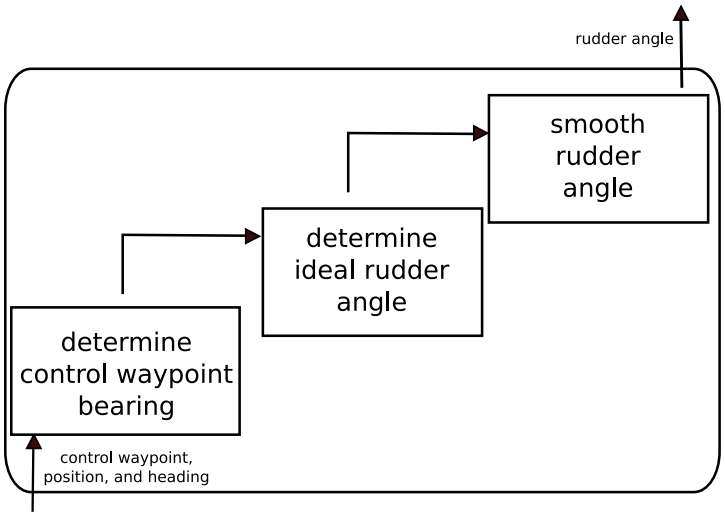


Figure 5.4: Autopilot Simulation Program

The autopilot uses the graphed function shown in Figure 5.5 to determine the appropriate rudder angle. It assumes that the rudder has a maximum displacement of 25 degrees (0.43 rad). It also uses a dead zone to prevent the autopilot from becoming an unstable controller.

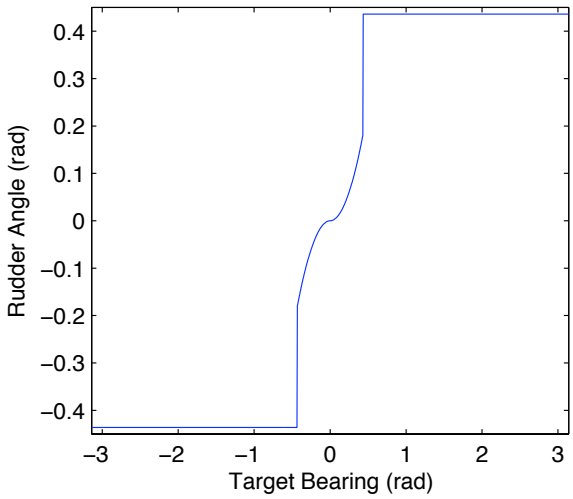


Figure 5.5: Autopilot Rudder Control Response to Corresponding Bearing

Unfortunately this dead zone makes it difficult for the autopilot controlled boat to arrive exactly on a control waypoint. Next, the autopilot smoothes the ideal rudder angle using,

$$\alpha_{n+1} = \alpha_n - \frac{1}{2}(\alpha_n - \gamma) \tag{5.1}$$

This rudder angle smoothing in Equation 5.1 simulates rudder position feedback to the autopilot. The term γ is the ideal rudder angle and α is the actual rudder angle. The rudder angle from the previous time step is used to determine the rudder angle in the next time step. As a result the actual rudder angle can only approach the ideal rudder angle supplied by the autopilot. It does this so that the boat does not turn too quickly, but also because it takes time for a real rudder to move to the desired rudder angle.

5.2.3 Communication Simulation Integration

The communication activity replicates the functions of an onboard communications system for a boat. It includes a number of supporting methods that allow it to manage the heading and position information associated with the rigid-body boat agent which is calling it. The communication

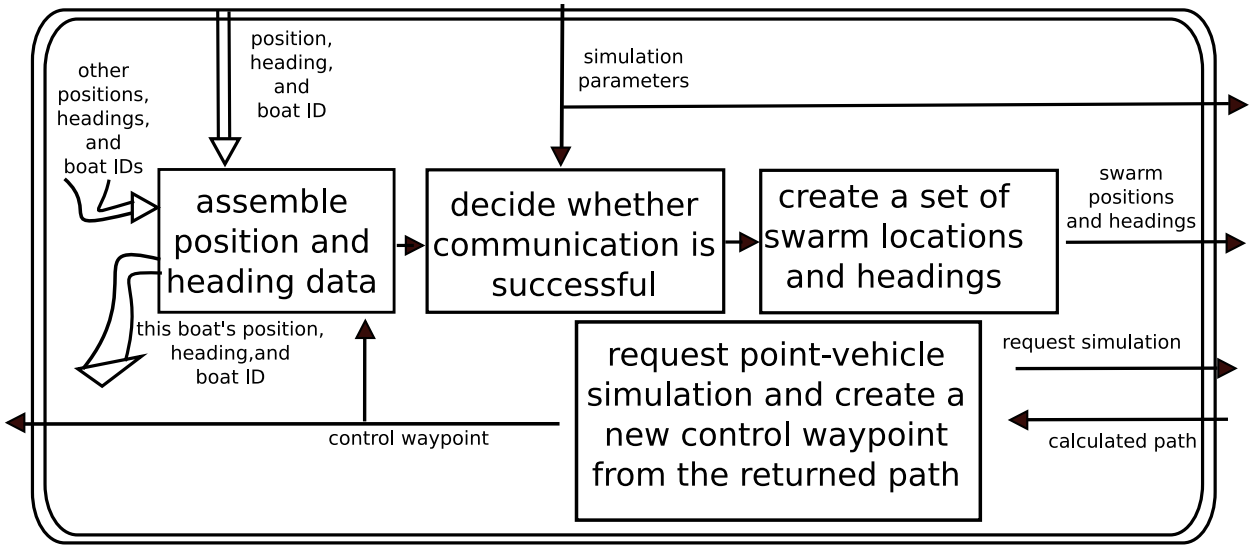


Figure 5.6: Communication Activity

activity as shown in Figure 5.6 sends the known position and heading information to the other instances of the communication activity. The frequency that this method runs establishes the interval between the two communications. This is how the poor communication is simulated. The communication activity listens for this message and other messages on the same channel from other boats' communication activities.

Once the data is received, it is analyzed and assembled in a data buffer. When various methods indicate a new control waypoint is required, the communication activity uses the positions and headings of the swarm that it assembled to request a new point-vehicle simulation. Once the point-

vehicle simulation is complete, the communication activity determines a control waypoint from the path generated by the point-vehicle simulation. The resulting control waypoints are added to the buffer replacing the old swarm position and heading information. These additions to the buffer allow the boat to make new control waypoints without needing a more recent set of swarm positions and headings. If the simulated data was not replacing the old data there would be a swarm with poor fidelity as discussed in Section 6.3.

5.2.4 Point-Vehicle Simulation Integration

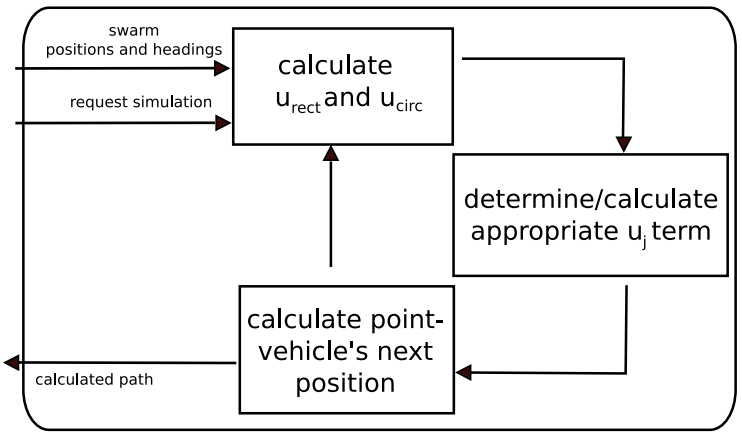


Figure 5.7: Point-Vehicle Simulation

The point-vehicle simulation is used by each of the boats to plot out a future path as seen in Figure 5.7. This program is used to generate the figures seen in Chapter 2. It is also used to make the comparison graphs that are discussed in Section 7.2. In each case the point-vehicle simulation program initializes using positions and headings of the swarm. This simulation always runs using one of the positions of a swarm waypoint. This data is used to calculate the u_j^{circ} and u_j^{rect} controls. These controls are mixed or selected based on which type of control is required. Next, the new positions and headings of the point-vehicles are calculated. At this point the simulation loops. This process is continued until the communication activity receives enough path information to generate new control waypoints. This control waypoint is used by the autopilot to send new rudder commands.

At this point both loops have been through one cycle. These rudder commands are used to calculate the new positions and headings of the rigid body boats. Once the rigid-body boats reach the new control waypoints, the communication activity collects position and heading information

which it uses to run another point-vehicle simulation.

5.3 Resulting Simulation

5.3.1 Understanding Graphs

There are several parameters that result through the addition of the autopilot and communication error simulation. They are displayed below in the format that we have developed in order to represent the paths of the rigid-body boats.

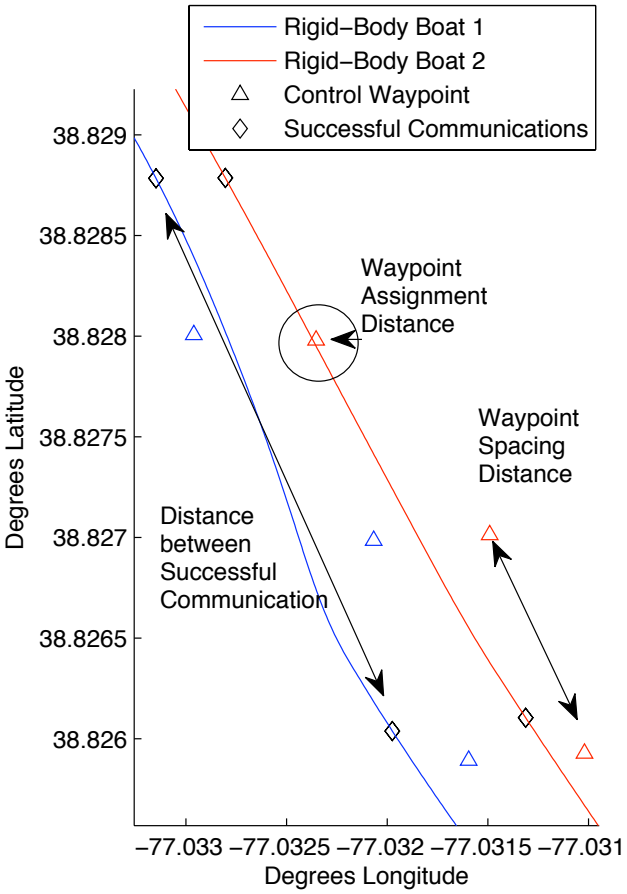


Figure 5.8: Tuned Parameters

Figure 5.8 implements a format that will be used through the rest of this thesis. The x-axis and y-axis are in degrees longitude and latitude respectively. These axes show the scale in which the boats are operating. A waypoint assigned to an individual autopilot is always represented as a triangle. The color of the triangle indicates the boat to which the waypoint is assigned. These way-

points are the control waypoints. A good communication is always represented as a black diamond. Since a good communication occurs across the formation, by definition, the communication is not assigned to a specific boat. In addition, a good communication shows where all of the boats are at a the same time. In Figure 5.8 the path of the boats do not always cross over the control waypoint. Rigid-body motion can make it too difficult for the autopilot to track the control waypoint. It is also possible that the boat has been assigned a new control waypoint which leads the boat on a trajectory away from the original control waypoint. There are many possible situations within this framework which are altered by changing certain values.

The *waypoint assignment distance* is the parameter which determines when a boat receives a new control waypoint. This parameter is the arrival radius of the control waypoints. When a boat reaches the arrival radius, the boat requests a new control waypoint. The distance between control waypoints, the *waypoint spacing distance*, is another carefully tuned parameter. This distance determines how far apart each control waypoint should be. If the distance is small the control waypoints will be closely spaced and better approximate curves. A large value will allow the boats to have more time to correct their course and accurately target the control waypoint. Another parameter, is the distance for good communications between vehicles. This distance is directly related to the *communication interval*. It plays an important role in Chapter 7 where different communication levels are used to test the control-law implementation.

5.3.2 Code Execution

In order to facilitate the understanding of the code, plots in Figure 5.9 show the way the final version of the control-law implementation assigns control waypoints. At the beginning of the simulation every vehicle is aware of the positions and headings of the entire formation. The initial locations are shown in Figure 5.9 (a).

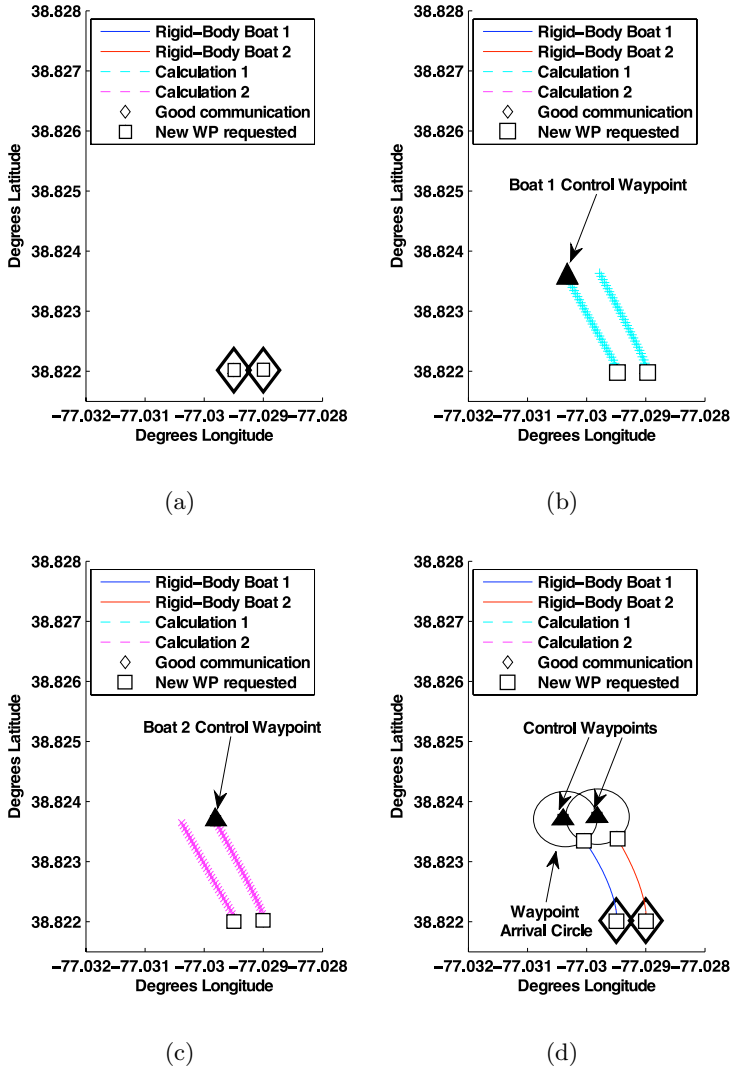


Figure 5.9: Control-Law Implementation Execution

At the start of the simulation the rigid-body boats do not have a control waypoint to follow. Each boat requests a control waypoint from the communication activity. The communication activity uses the point-vehicle simulation program and the most recent set of positions and headings to determine the next control waypoint. Even though each boat does its own control calculation, as shown in Figure 5.9 (b) and (c), both boats are able to determine the control waypoints of the other boats in the formation as well. The reason for having this level of redundancy is described in Section 6.3. Once the boats reach the specified waypoint assignment distance from the control waypoint, the boats request a new control waypoint. In this case the boats have not been able to communicate successfully with each other. As a result, they use their calculated control waypoints as the new positions and headings of the swarm. It turns out that this assumption of swarm

positions and headings leads to a higher fidelity formation.

This is where the process repeats. The boats are programmed to use this method of determining control waypoints through all situations. Communication interval and the choice of rectilinear or circular control do not affect the method of control waypoint assignment.

Chapter 6

Control Law Implementation Tuning and Optimization

6.1 Troubleshooting with Good Communication

There are several ways to check if the control-law implementation is working for a specific situation. The first way to determine if the control-law implementation is working is by visual inspection. While crude, visual inspection is a good heuristic to determine how the formation is behaving and helps determine the source of these behaviors.

6.1.1 Determining Waypoint Assignment Distance

Determining the appropriate waypoint assignment distance is important for any swarm using the control-law implementation discussed in this thesis. Unfortunately, there is not a clear mathematical way to determine an appropriate waypoint assignment distance in advance. This parameter depends on a number of factors. The speed, inertia of the boat, and turning radius of the boat coupled with the aggressiveness of the autopilot are all factors in determining the appropriate waypoint assignment distance. Figure 6.1 shows how the rigid-body boats circle control waypoints which have waypoint arrival distances which are too small. Since the rigid-body boats are not able to move into the arrival radius, they are caught circling their control waypoints. However, the waypoint assignment distance can also be too large as it determines the lower limit of the waypoint spacing distance.

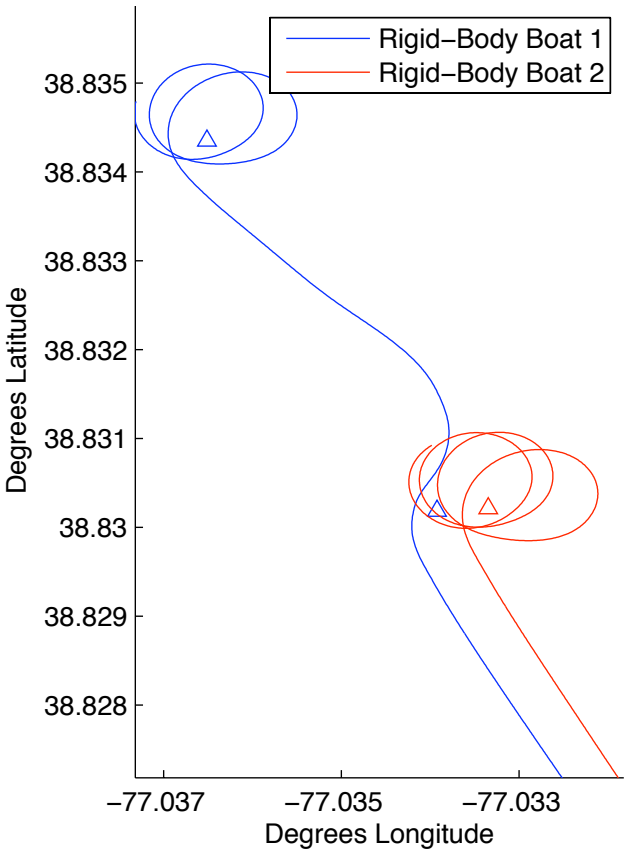


Figure 6.1: Inefficient Course Tracking Using Bearing to Determine Waypoint Spacing

6.1.2 Waypoint Spacing Distances

The next parameter to determine is the appropriate waypoint spacing distance for the rectilinear controlled portion of this course. In this case, it does not seem as if waypoint spacing distance is a major factor for perfect to good communication situations. The error does not vary noticeably over different arbitrary courses. Although it became apparent that waypoint spacing distance would play a large role if it were assigned more extreme values. If the waypoints were spaced closely then the control waypoints should have been better at approximating the curves that the control law produces. However since the autopilot programs have a dead-zone, they do not aggressively target waypoints. This dead-zone makes it so that any advantage gained by approximating the control paths' curves with more waypoints is not effective. Conversely, a large waypoint spacing distance causes the rigid-body boats to be unable to make corrections often enough to their course.

6.1.3 Circling Waypoint Spacing Distance

After running the simulation through the entire course it became apparent that there were some problems with assigning control waypoints once the vehicles started circling the swarm waypoint. This problem is apparent in Figure 6.2. The rigid-body boats were not receiving enough information to approximate a circle. In order to fix this problem, the boats now receive a waypoint at a third the distance of the rectilinear waypoint spacing distance. The new circular waypoint spacing distance is still greater than the waypoint assignment distance.

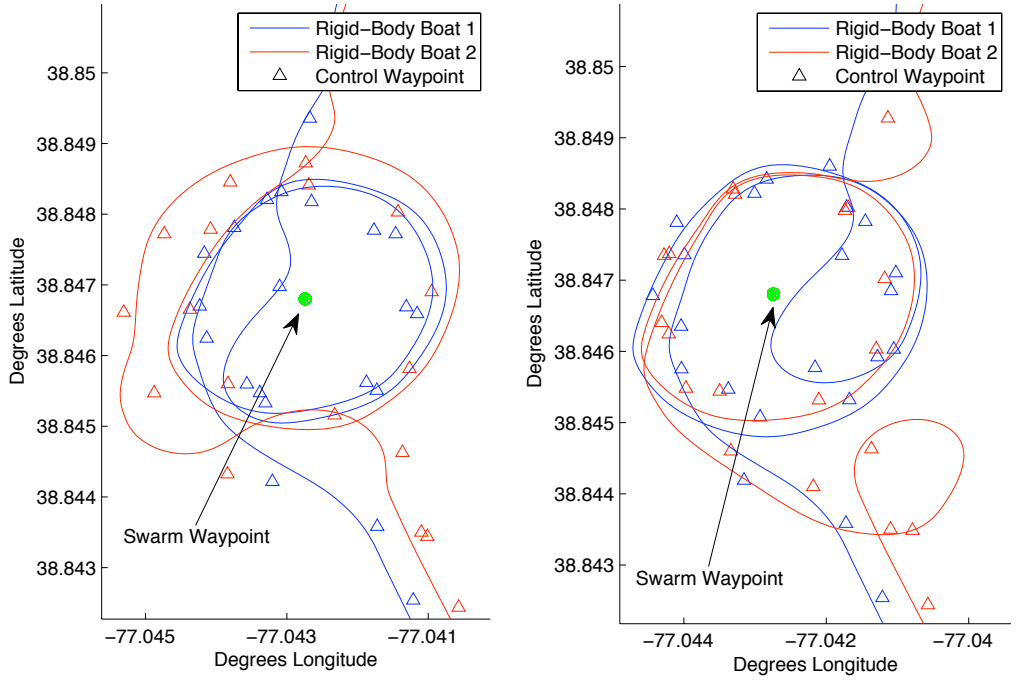


Figure 6.2: (a) Poor Waypoint Circling Caused by Poor Control Waypoint Spacing Distance (b) Better Swarm Waypoint Circling through more Closely Spaced Control Waypoints

6.1.4 Additional Issues

While testing the vehicles through the course, it became apparent that there was some degree of difficulty when pulling out of the waypoint circling mode. It turned out that an easy way to fix this problem was to remove the capability to mix the controls when transitioning from circular to rectilinear control. Since the vehicles need to move to the next swarm waypoints at the same time, it is preferable to remove this capability. While it is unfortunate that this control-law implementation is not able to capture all of the original control law's capabilities, it is certainly not limiting the

controllability of the vehicle swarm.

6.2 Low Communication Conditions

Low communication is loosely defined in this thesis as when there is a good communication between vehicles one to five times every two control waypoints. This communication level regime is also where a number of problems exist for the control-law implementation. Interesting problems are created, such as path creation error, and waypoint spacing issues.

6.2.1 Control Waypoint Spacing Techniques

Before this testing began it was obvious that there would be problems if the vehicles did not have the same technique for determining the appropriate control waypoint spacing. In low communication conditions it is possible that vehicles may have to assume that the old control waypoints are the vehicles' actual positions and headings. If individual vehicles in the swarm determine control waypoints differently from each other, then they will do control calculations differently and the whole system could potentially break down. In order to stop this breakdown from happening, the point-vehicle simulation checks each vehicle in the simulation to see if it has reached the specified waypoint spacing distance from its initial position. Each vehicle is able to use this method to determine which control waypoint it is assigned, but also which control waypoints the other boats are assigned.

6.2.2 Angle-Based Waypoint Spacing

Creating new waypoints based on bearing was another possibility. Bearing is the angular difference between the heading of a vehicle and the heading required to face the waypoint. This method of spacing control waypoints is necessary for creating control waypoints for a hardware autopilot. The hardware autopilot requires that all of the waypoints it receives are limited to only a 30 degree difference from its original waypoint. If the bearing to the next control waypoint is greater than 30 degrees the autopilot requires a user to confirm that this waypoint is correct. While this is good for boat operation safety it does not work well with the control-law implementation that is being developed in this thesis. Figure 6.3 shows that for low communication situations, the bearing waypoint spacing method does not allow the boats to efficiently track the course to which the swarm is assigned.

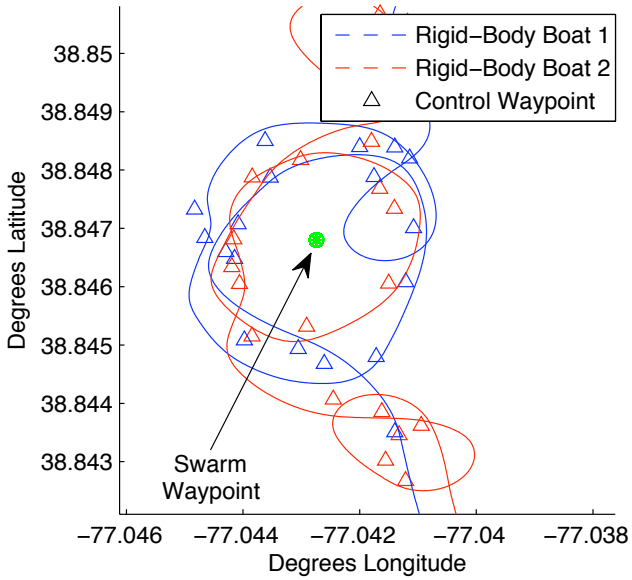


Figure 6.3: Inefficient Course Tracking Using Bearing to Determine Waypoint Spacing

The pretzel shaped smaller circles in Figure 6.3 are not present in other simulations. These smaller circles are created by the autopilot turning a different direction than the simulated point-vehicles do. The inertia in the boat and rudder make it so that the rigid-body boat will try to approximate the sharp turns created by the path differently than those chosen by the point-vehicles described in Equation 2.1. These circles cause the boats to become much further apart such that they are not able to travel as far over the same length of time. In addition, as this formation is scaled with more boats, there will be an increased risk of collision between vehicles because of this low fidelity.

6.2.3 Enforcing Waypoint Assignment Distance

When developing the control-law implementation it was decided that the communication activity would not use any of the position and heading information about its vehicle except at moments of good communication. In the formulation of this experiment it was important that the vehicles only know the positions of all of the vehicles in the swarm at any given time. This was to simplify the system, ensure scalability, and avoid conditions explained in section 5.1.2. However it can be argued that under this specific situation it is possible to give the vehicle's position to the communication activity without causing any of the discussed issues. In Figure 6.4 there is an issue caused by the way that waypoints are assigned in relation to the lack of communication between boats.

In this situation there are only one or two good communications between control waypoints.

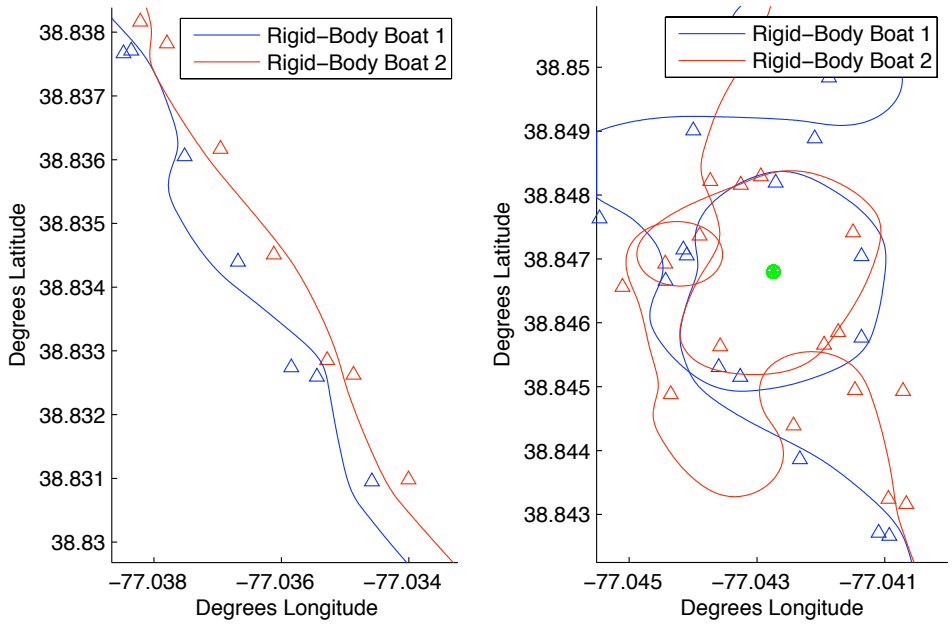


Figure 6.4: (a) Problems with Rectilinear Path Following (b) Problems with Circular Path Following

When the point-vehicle simulation is called to plot the path forward, it assigns a control waypoint that is too close to allow for the autopilot to track. This happens because the point-vehicle simulation does not "know" how close the rigid-body boats are to the locations that it is using to perform the calculations. As a result it stops the point-vehicle simulation once one of the point-vehicles arrives at the predefined waypoint spacing distance from the start of the simulation. In order to fix the problem, the state buffer was modified to allow it to calculate how far the rigid-body boats had traveled since the last good communication. This value is added to the waypoint spacing distance when the point-vehicle simulation runs in order to prevent this situation.

6.3 Severe Communication Problems

For severe communication problems it is possible that there will not be a good communication between vehicles after a given initial waypoint request. There are a couple ways to deal with this problem beyond deciding to stop the simulation. One possibility is to constantly extend the waypoint spacing distance indefinitely until there is another communication. While this seemed like a good idea, there were several problems as seen in Figure 6.5.

These results are actually caused by poor control law utilization. When the point-vehicles, in the point-vehicle simulation, approach the waypoint, there is a change in the mixing of the circular and

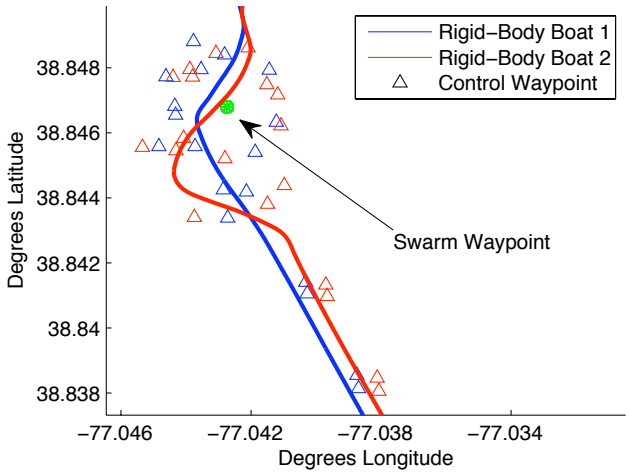


Figure 6.5: Poor Control Fidelity caused by Lack of Location Information.

rectilinear control laws. When the simulation is called again, the boats start from the last positions and headings supplied by the good communication. However, the simulation also starts with the most recent ϵ value generated by the previous point-vehicle simulation. These changes cause the simulated point-vehicles to approach the waypoint in a different way and creates a discontinuity in the path that the point-vehicles are creating. This path is smoothed by the control waypoints, but the result is an inability of the rigid-body boats to effectively circle the swarm waypoints.

In order to solve this problem, the autopilot assumes that the rigid-body boats will arrive at their next control waypoints at approximately the same time. This is a good assumption since each control waypoint is spaced the same distance from the boat's corresponding position. If the rigid-body boats did indeed arrive at their waypoints nearly simultaneously, the control waypoints would be a good approximation of the rigid-body boats' positions and headings. The communication activity uses this assumption by adding the control waypoints generated by the point-vehicle simulation to the communication activity. If a good communication takes place, the good communication data overwrites the control waypoint position and heading just as it would any other position and heading data set.

Chapter 7

Simulation Analysis

7.1 Determining and Quantifying Success

Determining the success of a given approach is the cornerstone of a good experiment and crucial for developing a control-law implementation such as this one. There are several ways to check if the control-law implementation is not working for a specific situation. The first way to determine if the control-law implementation is working is by visual inspection. While crude, visual inspection captures some of the most obvious problems this control-law implementation may experience.

Another way to determine this error is to look at how closely the rigid body simulation follows the paths laid out by the point-vehicle simulation. This error is a strong indicator of formation fidelity since it is effectively a measure of how closely the swarm follows a set path. If the swarm's path is designed to avoid various obstacles, then a high swarm error would indicate that the vehicles in the swarm have the potential of colliding with these obstacles.

$$\delta_{Swarm} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{(x_i - \acute{x_i})^2 + (y_i - \acute{y_i})^2}{r_0^2}\right) \tag{7.1}$$

Swarm deviation is measured during the rectilinear control portion of the course. There are so many ways that the vehicles can circle a waypoint that it is difficult to make sense of the resulting error in circling mode. The values x and y are the positions of the point-vehicles or the rigid body boats as a function of distance from the starting location, whereas the $\acute{x}$ and $\acute{y}$ are the desired path values as a function of distance from the starting location. The r_0 term is the same as the one from Equations 2.2 and 2.3, and N is the number of evenly spaced distances included in this calculation.

Yet another way of measuring error is by looking at how close or far the vehicles are when compared with their separation distance indicated by the point-vehicle simulation. Since the vehi-

cles oscillate around this spacing, vehicles dispersing is just as problematic as vehicles converging because one will inevitably lead to the other. This error will be referred to as *separation error* and is determined by Equation 7.2.

$$\delta_{Sep} = \frac{1}{N}\sum_{i=1}^{N}\left(\left|\frac{\sqrt{(x_{1i}-x_{2i})^2+(y_{1i}-y_{2i})^2}}{r_0}-1\right|\right) \tag{7.2}$$

The separation error average (δ_{Sep}) is defined the average separation of the boats divided by their prescribed separation. The values x_1 and y_1 are the positions of the the first rigid-body boat as a function of distance, whereas the x_2 and y_2 are the positions of the second rigid-body boat as a function of distance. The r_0 term is the same as the one from Equations 2.2 and 2.3, while N is the number of points included in the average.

Another possible error test uses the swarm centroid variation as described by Fax and Murray [2002] and Moreau [2005]. However, this proved a difficult parameter to derive in a meaningful way for several reasons. The inclusion of waypoints and control mixing complicates the process of determining the center of the swarm. The software autopilots' dead-zone makes it so that their transience never dies out so the boats' paths rarely lie on top of the path they are following even in good communication situations.

7.2 Experimental Results

The communication error is simulated by only allowing communication between boats at a specified interval. If these intervals can be related back to the point-vehicle simulation, then a meaningful comparison can be made between the two tests. When the vehicles in the point-vehicle simulation communicate every time step, there is effectively perfect communication in the formation. In between each time step there is an average distance that the point-vehicles travel. If the rigid-body vehicles are also allowed to communicate at an interval in which they cover the same distance, the results from both of these tests should be comparable. While measuring a communication interval in terms of a distance is unusual, it is the most useful standard in this situation, especially since the time step for the point-vehicle and rigid-body simulations are different by two orders of magnitude. Using the same time step interval to compare communications would be meaningless. Since both the point-vehicles and the rigid-body vehicles move with a nearly constant speed and are stepped through discrete time, measuring the distance (or unit steps) between communications is more meaningful. In Figure 7.1 the point-vehicles and the rigid-body boats are simulated for varying

levels of communication. The point-vehicles use the original control law.

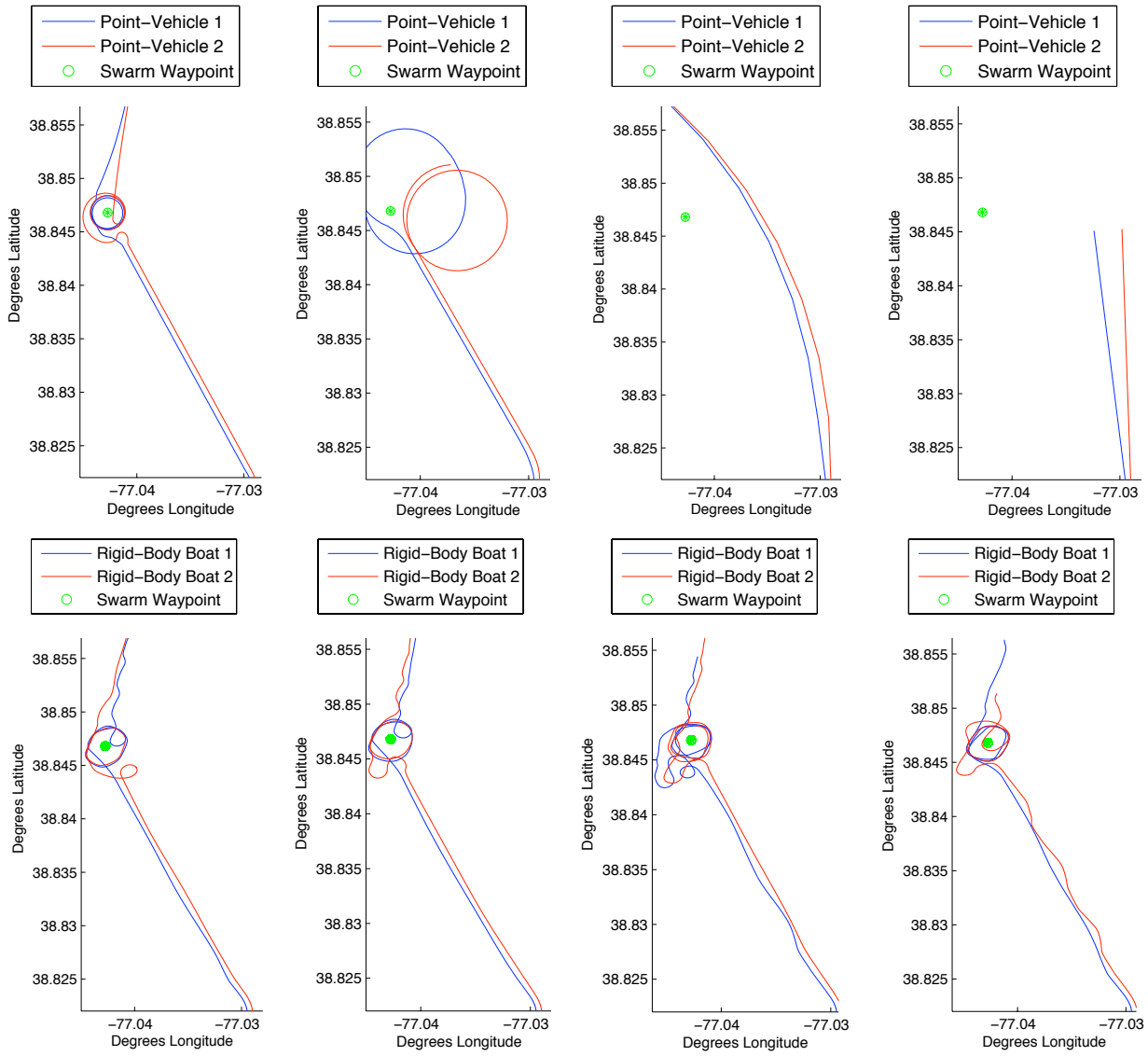


Figure 7.1: (a) One Communication for One Unit Step (b) One Communication for Ten Unit Steps (c) One Communication for One Hundred Unit Steps (d) One Communication for One Thousand Unit Steps

In Figure 7.1 there is a general trend of lower formation fidelity as communication becomes worse. However, the rigid-body simulation with the control-law implementation is not as strongly affected by communication problems as is the original control law. This loss of fidelity should be expected since the point-vehicles need communication in order to determine a new steering command. As there is less communication there are fewer steering commands. As a result the tight curves under perfect communication turn into lines as communication decreases. The decrease in

communication of the control-law implementation does not degrade the performance in a similar way. It would seem intuitive that a progressive decrease in communication would cause a corresponding decrease in fidelity. However, this is not the case. For example at the "one communication for one hundred unit steps" level the rigid-body boats have a large degree of difficulty in circling the swarm waypoint. However at "one communication for one thousand unit steps", the rigid-body boats do not have as much trouble circling the waypoint. What is happening is that the rigid-body boats have to make assumptions about their and the other boats' positions and headings in the swarm for an extended period of time. Once a good communication occurs, the boats have to move along a far different course than the one they were following. As a result it becomes more difficult for them to circle the swarm waypoint at the points desired.

7.2.1 Swarm Deviation

Now that this system has been thoroughly examined and developed, it is possible to perform tests to indicate the effect of communication error accumulated on the rectilinear portion of the course. It is also possible to compare the total swarm deviation occurring with point-vehicle simulation and the control-law implementation.

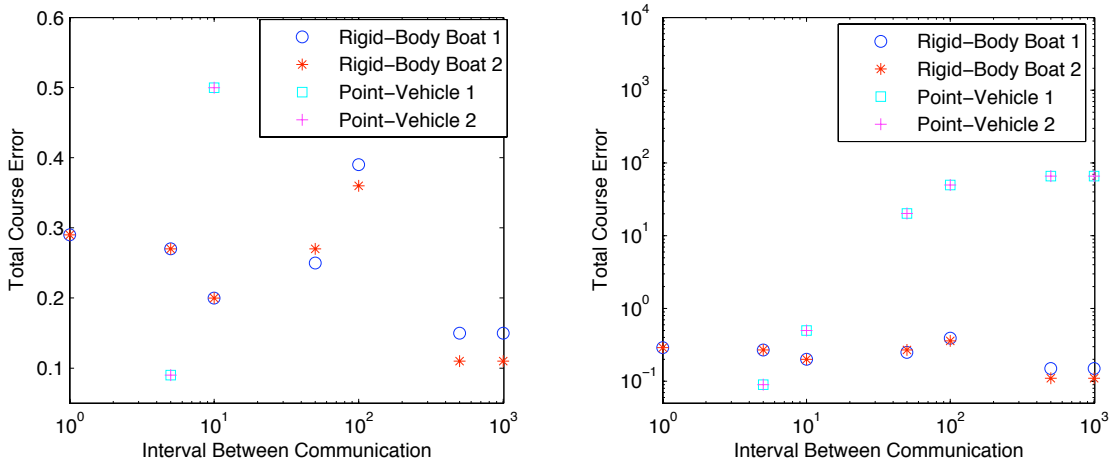


Figure 7.2: (a) Swarm Error Comparison (b) Rescaled Swarm Error Comparison

In Figure 7.2 each point represents one simulation at specified parameter values. Multiple tests with the same values would only yield the same results since there are no random components built into the simulation.

The error associated with the rigid-body boats, seen in Figure 7.2 (a), is caused by how the vehicles navigate the course and the approximatory nature of the control-law implementation. Once again it may seem counterintuitive that the swarm deviation dips down at the end. This should be an expected phenomenon. When vehicles are able to communicate, they are unable to make adjustments to control their spacing. As a result, they can better follow the swarm waypoints. However, these minor adjustments make it so that the rigid-body boats do not follow the course as precisely. When there is no communication, the boats can not make changes to their course path and have to follow the path laid out for them.

Figure 7.2 (b) is on a different scale in order to capture the large errors measured for the low communication situations for the point-vehicle simulation. This large error is caused by the inability of the point vehicles to communicate with each other. This makes it so the point vehicles cannot receive a new rudder command and therefore are controlled less. The strong correlation between the qualitative results and the swarm deviation values suggest that this is a good measure of formation fidelity.

7.2.2 Separation Error

The separation error can also used in the comparison of the point-vehicle simulation and the control-law implementation.

The point-vehicle results show the formation retains a fairly good spacing distance as seen in Figure 7.3. The separation error does not increase several orders of magnitude when the communication interval is increased. However, the error values are still higher than that of the control-law implementation.

The resulting error for the rigid-body boats is far more intuitive than Figure 7.2 (a). As the communication between boats decreases, the separation between the boats varies more. When the boats communicate, some of the communications indicate to the boats that they are too close together and so they move further apart. In addition, without the ability to make changes to their course, the boats are forced to follow the control law paths without compensating for how the software autopilots interpret the control waypoint approximation of those paths. The fact that these boats appear to get closer to each other is not a substantial problem. Increasing the

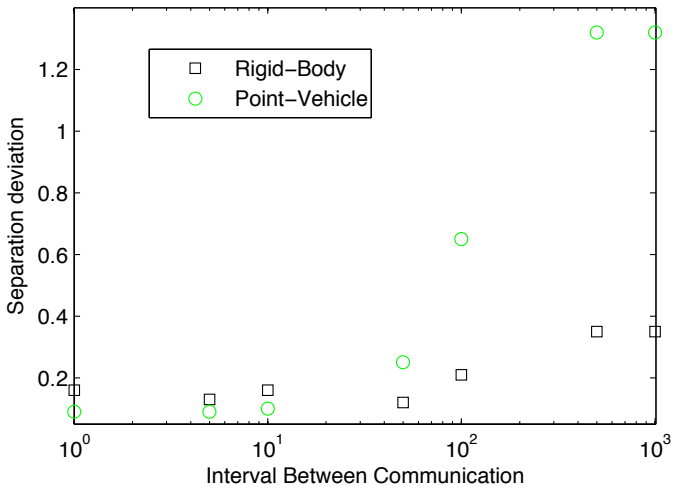


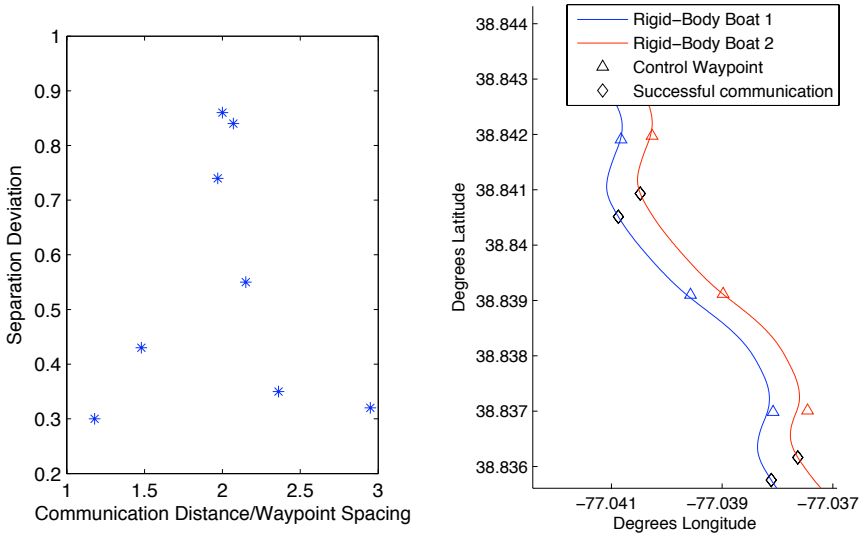
Figure 7.3: Separation Deviation for Rigid-Body Boats and Point-Vehicles

separation parameter r_0 slightly would make the risk of collision far less probable. This is also the conclusion drawn for the formation tested by Pongpunwattana et al. [2007]

7.2.3 Resonance Induced Swarm Deviation

In this low communication regime there are some unexpected and important interactions. After examining the swarm deviation results, it became apparent that one of the error values was particularly high. A plot of this particular simulation showed that the boats' paths were oscillating around the path they were supposed to be following. At the same time, the plot showed that the rigid-body boats were only able to communicate at points far away from their next control waypoints. This indicated that the communication interval and control waypoint spacing were causing some sort of instability in the control-law implementation. In order to investigate this phenomenon the level of communication was varied around the initial communication level that lead to this inquiry

Figure 7.4 (a) shows the resulting plots. This data indicates that the communication is causing some sort of resonance within the control-law implementaion. The simulation with the highest swarm error is shown in Figure 7.4 (b). The vehicles are receiving positions and headings from each other just before the waypoints are requested. This information exchange causes the control path to be generated when the boats are farthest away from the prescribed path. As a result the autopilots have to aggressively track the new control waypoints, which causes them to be far away from the prescribed path when the next good communication occurs and the positions and headings are exchanged.



(a) (b)

Figure 7.4: (a) Unstable Control Situation Created by Resonant Situation (b) Swarm Deviation in Boat 1 Caused by Resonant Situation.

As the ratio of communication to waypoint spacing distances between control waypoints approaches two, the swarm error increases exponentially. This marked increase in swarm error is actually a more complicated phenomenon than that suggested by a simple ratio. The boat paths with respect to the tracked control waypoint actually oscillate while approaching the waypoint from different directions. It actually takes the boats longer to reach half of the waypoints because of the oscillation in which the vehicles are caught.

It is not clear how to avoid this problem. Even if the boats had programs to detect and recognize this phenomenon, it would be difficult for them to avoid this situation. The boat swarm would collectively have to determine a new control waypoint spacing distance. This correction would be difficult to implement because this phenomenon is caused by poor communication. Fortunately, this phenomenon is not likely to occur in real-world testing since it would require precisely periodic communication at a given frequency. In addition the boat separation remains mostly constant so while the boats will not collide with each other they may collide with other unseen obstacles.

Chapter 8

Conclusions

8.1 Review of Material

The material in this body of work describes the process of developing a new control-law implementation for an existing control law. The first few chapters review material that is the foundation of this thesis research effort. In Chapter 2, the chosen control law is characterized and developed for the environment in which it is later tested. Chapter 3 describes previous research efforts to make formations that are resistant to poor communication situations. This research helps to determine a relevant and effective way to model communication error. In Chapter 4, the existing software is described and evaluated.

The successive chapters describe the process developed to control a formation of unmanned autonomous boats. The process for developing the programming environment for the the simulation is discussed in Chapter 5. Parameters emerging from the new system are tested and optimized in Chapter 6. Finally, in Chapter 7, the tuned system is tested and evaluated.

8.2 Success of Project

The control-law implementation appears to be a successful way to make the control laws as described by Justh and Krishnaprasad [2003] resistant to communication error. At the same time the control-law implementation maintains relatively high fidelity through poor communication while the original control law does not. While the control law implementation does not capture the tight curves that the point-vehicle simulation describes, tuning various parameters effectively allows the control-law implementation to capture the more important aspects of the control law. This control-law implementation should certainly not be used when there is perfect, or almost perfect

communication. However, the control-law implementation seems to be a good choice for reducing poor fidelity in all but the most severe communication problems through complex courses. This approach is actually a significant step forward in working towards a real-world implementation of this control law. The complete program is sufficiently compact to run on a small computer that could be placed on a real boat. In addition, the control law implementation still accounts for most of the limitations of a hardware autopilot. It can communicate the control law's path to the autopilot through control waypoints. The waypoints can be sufficiently far apart to allow the autopilot to react and change course. In good communication conditions the bearing check is a valid way to assign control waypoints. While the rigid-body simulation does not account for wind and ocean waves, the incorporation of this autopilot would mitigate the problems these phenomena might cause.

8.3 Future Work

There are a number of avenues of research that can move the work in this thesis forward. One possible next step is to make the vehicles change the way in which they circle a waypoint. Currently, the way that vehicles maneuver in order to be equally spaced around the waypoint is difficult for the control law implementation to approximate. Using a different method to circle a waypoint may be advisable.

Boundaries and other obstacles can be added to subsequent testing scenarios, which would allow the boats to operate in real environments. Coastlines or dangerous areas can be approximated by a series of waypoint-vehicles in the simulation which would be added to the point-vehicle control calculation when the boats are too close to a defined boundary. Once the boats are able to move further away from the boundary, the boundary-vehicles could be removed so that the swarm would not get stuck following the boundary. With a properly tuned separation parameter, the boats would avoid the boundary as they would any other boat. It would be important to properly space the boundary-vehicles such that the boats would not try to move in between the boundary vehicles themselves.

The resonance phenomenon described in section 7.2.3 could be avoided or created in various ways. Changing the shape of the waypoint assignment circle might be enough to dampen this oscillation. The boats could also move through a series of differently spaced control waypoint assignments. This phenomenon could also be used to destabilize an enemy swarm of vehicles. If it were possible to know when the swarm would request a new set of waypoints, the swarm

communications could be jammed by outside sources until that point. This would cause higher swarm error than jamming the boats the entire time.

Random communication is another avenue of potential investigation. Some Research described by Porfiri and Stilwell [2005] suggest that it would be more difficult for vehicles to approach consensus through random communication. It would be interesting to see how the random communication would affect the fidelity of the control-law implementation. While the resulting analysis would be difficult, creating the random communications in the simulation environment would be relatively simple.

There are a number of options available when considering ways to advance the research stated in this thesis. The thesis research has shown that it is possible to implement the control law developed by Justh and Krishnaprasad [2003] with unmanned autonomous boats and maintain high fidelity to the control law. However, there are still steps that need to be taken in order to practically implement this system of boats in real-world scenarios. Optimizing the circling control would make the swarm move between swarm waypoints more efficiently. Adding boundary awareness would allow the boat swarm to avoid obstacles. In addition, further investigation into possible resonant conditions would also allow for more efficient swarm movements. Finally, testing different types of communication situations would facilitate further understanding of the control-law implementation. While there are many steps before this control implementation can be used effectively on actual boats, the research described in this thesis is a large and critical first step.

Bibliography

Arrendondo, G. A., W. H. Chriss, and E. Walker [1973] A multipath fading simulator for mobile radio. *IEEE Transactions on Communications* **21**, 1325– 1328.

Cortés, J., S. Martínez, and F. Bullo [2006] Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. *IEEE Transactions on Automatic Control* **51 No. 8**, 1289–1298.

Cortesi, R. S., K. S. Galloway, and E. W. Justh [2006] A biologically inspired approach to modeling unmanned vehicle teams,. *in Proceedings of SPIE* **6964**.

Fax, A. and R. M. Murray [2002] Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control* **49 No. 9**, 169182.

Greenwood, D. [1998] *Principles of Dynamics 2nd Ed*. Prentice-Hall, Inc.

Justh, E. W. and P. S. Krishnaprasad [2003] Steering laws and continuum models for planar formations. *42nd IEEE Conference on Decision and Control* .

Klein, G. J. and K. A. Morgansen [2006] Controlled collective motion for trajectory tracking. *in Proceedings of the American Control Conference*.

Kohler, W. and L. Johnson [2004] *Elementary Differential Equations with Boundary Value Problems*. Pearson Addison Weasley.

Moreau, L. [2005] Stability of multiagent systems with time-dependent communication links. *IEEE Transactions on Automatic Control* **50 No. 2**, 169182.

Pongpunwattana, A., B. I. Triplett, and K. A. Morgansen [2007] Target tracking control with limited communication and steering dynamics. *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* **Online**.

Porfiri, M. and D. Stilwell [2005] Consensus seeking over random weighted directed graphs. *IEEE Transactions on Automatic Control* **52 No. 9**, 1767–1773.

Sepulchre, R., D. A. Paley, and N. E. Leonard [2007] Stabilization of planar collective motion: All-to-all communication. *IEEE Transactions on Automatic Control* **52 No. 5**, 811–824.

Stevens, B. and F. Lewis [2003] *Aircraft Control and Simulation*. John Wiley & Sons, Inc.

Takai, M., J. Martin, and R. Bagrodia [2001] Effects of wireless physical layer modeling in mobile ad hoc networks. *In Proceedings of MobiHooc 2001, Long Beach* **Oct.**, 87–95.

Wang, L.-S. and P. Krishanprasad [1992] Gyroscopic control and stabilization. *Journal of Nonlinear Science* **2**, 367–415.

Zang, Y., L. Stibor, G. Orfanos, S. Guo, and H. Reumerman [2005] An error model for inter-vehicle communications in highway scenarios at 5.9 GHz. *Proceedings of the 2nd ACM international workshop on performance evaluation of wireless ad hoc, sensor, and ubiquitous networks* page 4956.

Zhang, F., E. W. Justh, and P. Krishanprasad [2004] Boundary following using gyroscopic control. *43rd IEEE conference on decision and control* **1-5**, 5206–5209.

Zhang, F. and N. Leonard [2007] Coordinated patterns of unit speed particles on a closed curve. *Systems and controls letters* **56**, 397–407.