High Performance Path Following for Marine Vehicles Using Azimuthing Podded Propulsion

by

Matthew B. Greytak

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2006

Author .....
Department of Mechanical Engineering
December 20, 2005

Certified by .....
Michael S. Triantafyllou
Professor, Mechanical Engineering
Thesis Supervisor

Certified by .....
Franz S. Hover
Principal Research Engineer, Mechanical Engineering
Thesis Reader

Accepted by .....
Lallit Anand
Chairman, Department Committee on Graduate Students

High Performance Path Following for Marine Vehicles Using Azimuthing Podded Propulsion

by

Matthew B. Greytak

Submitted to the Department of Mechanical Engineering
on December 20, 2005, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

Podded propulsion systems offer greater maneuvering possibilities for marine vehicles than conventional shaft and rudder systems. As the propulsion unit rotates about its vertical axis to a specified azimuth angle, the entire thrust of the propeller contributes to the steering moment without relying on lift generation by a control surface such as a rudder. However, the larger sideforce and moment cause the ship to enter the nonlinear realm sooner than a ruddered vessel. Furthermore if the rudder or azimuthing propulsor is aft of the vessel's center of gravity then the system is non-minimum phase; during a turn the ship center initially moves in the direction opposite the turn. For these reasons it is necessary to design a robust maneuvering control system to set the azimuth angle of the propulsor in an intelligent and stable manner.

This thesis focuses on the path following performance of a vessel with podded propulsion. The enhanced maneuvering abilities of such vessels allow the time constant of cross-track error response to be greatly reduced. Additionally these vessels can follow course changes and waypoints more precisely than ruddered vessels. A simple path following algorithm was developed to achieve this performance; the algorithm uses simulation-based feedforward terms to anticipate the sliding motion of the vessel during a turn. The stability and performance analysis was performed in three domains: linear theory, a nonlinear simulation, and experiments with a 12-foot autonomous surface vessel. Experiments confirmed that path following performance was vastly improved using the feedforward algorithm for waypoints at which the course change angle was large.

Thesis Supervisor: Michael S. Triantafyllou
Title: Professor, Mechanical Engineering

Thesis Reader: Franz S. Hover
Title: Principal Research Engineer, Mechanical Engineering

Acknowledgments

I would first like to thank my advisors, Professor Triantafyllou and Dr. Hover, for their constant support, suggestions, and encouragement. I have great appreciation for the work of Jeff Stettler, whose autonomous kayak was incredibly seaworthy and whose data was invaluable throughout the project. Stephen Licht and Mike Benjamin provided some key tips, advice, and equipment. Finally I would like to thank the MIT Sailing Pavilion staff for their availability, assistance, and considerations: Fran Charles, Mike Kalin, and Dwight Brown.

Contents

Chapter 1

Introduction

Marine surface vessels equipped with azimuthing podded propulsion systems are capable of higher rates of turn and higher sideforces than vessels using the standard fixed propeller and rudder configuration. This allows the vessel to perform faster maneuvers that drive the ship into the nonlinear regime. This thesis explores heading control and path-following control for vessels with azimuthing podded propulsion.

Very precise path following for marine vehicles is necessary for mapping, surveying, and navigating in restricted waterways. When the path is highly demanding the enhanced maneuverability of podded propulsion vessels must be exploited. In this thesis a feedforward algorithm is designed to anticipate the turning distance of the vessel for various course change angles and match the vessel trajectory to the path as closely as possible. In fact the resulting trajectory can be compared to the best possible trajectory given the vessel constraints.

The heading control and path-following control problems are studied with linear theory, nonlinear simulations, and experimental tests with an autonomous surface vessel. For simplicity the vessel is assumed to operate at a constant propeller input torque throughout the maneuvers.

1.1 Previous Work

There has been very little exploration of the maneuvering problem specific to underactuated non-minimum phase surface vessels with podded propulsion. However, pieces of the problem have been studied to various degrees. Podded propulsion research has focused on the

hydrodynamics, drag, and efficiency of azimuthing propulsion units. The unsteady forces and dynamics of podded propulsion units were investigated in [15], but there has been no study of the applicability of linear control techniques to the very nonlinear dynamics caused by these units at high azimuth angles.

The more general heading controller problem for surface vessels has been studied thoroughly, focusing primarily on ships with the standard propeller and rudder configuration because that is the most common arrangement. Linear control is usually sufficient for the heading control problem [5]. However, for more precise control and robustness a variety of nonlinear controllers have been investigated including adaptive control, fuzzy control, and Lyapunov-based controllers.

The path following problem has been met with a similar set of high-level approaches (adaptive, fuzzy, and genetic algorithms), but all of these controllers share the same core: line-of-sight path following. The specifics of how the forward point is chosen and how it is adjusted near waypoints varies slightly from controller to controller but the general concept is well accepted because if its inherent stability properties and ease of implementation.

1.2 New Contributions

This thesis ties together the podded propulsion capabilities, linear and nonlinear heading controllers, and the standard path following algorithm to prove total system stability and demonstrate the heightened performance attained by a vessel equipped with podded propulsion. This performance is demonstrated on straight-line paths and piecewise linear courses in simulation and with experiments using an autonomous podded-propulsion vessel.

The standard line-of-sight algorithm (quasi-P control) is enhanced with trajectory-based derivative and integral terms to create a quasi-PID path following controller. Linear theory is used to verify the stability of the entire system consisting of a PID heading controller surrounded by the quasi-PID path following controller.

The time-based integral term of the quasi-PID path following controller is replaced with a space-based integral term to improve the stability of the system when the vessel slows to well below its operating point or approaches the path leg at a large angle. The space-based integral is related to the area between the vessel trajectory and the desired path, as opposed to the standard time-based integral term that accumulates error in time.

Motivated by the non-minimum phase behavior of the path-following system, a feedforward component is added to the path-following controller near the waypoints. This feedforward term ensures that the vessel trajectory matches the desired path as well as physically possible given the constraint of a non-minimum phase vessel operating at constant thrust.

1.3 Software Tools and Experimental Hardware

All computational work for this project was performed using Mathworks' MATLAB and Simulink software. The Control Systems Toolbox and the Optimization Toolbox received the most heavy use. Experiments were performed using an autonomous surface vessel constructed from a commercial kayak, a trolling motor, an onboard computer, and a variety sensors. MATLAB's xPC Target interface was used to communicate with and control the onboard computer system. Data characterizing the hull and propulsor was available from [15] and used to create a nonlinear model and simulation of the system.

Tests with the autonomous system were performed on the Charles River in Cambridge/Boston MA.

Chapter 2

Background

2.1 Azimuthing Podded Propulsion

For centuries rudders have been used to steer marine vessels of all kinds, from oil tankers to AUV's to dinghies. Rudders are attractive for many reasons: they have very low drag at zero angle of attack, they can generate substantial lift forces if properly sized, and they are relatively easy to actuate. However, they are limited by two factors: they are ineffective at low speeds and they stall for angles of attack greater than around 15°. (The actual stall angle depends on the rudder geometry.) The first factor limits their usefulness in close quarters or docking situations and the second factor limits the turn rate of the vessel.

Both of these limitations are addressed by another form of steering: azimuthing podded propulsion. The propulsion system in a conventional ship consists of an engine, a gearbox, and a straight shaft out to a propeller at the stern of the ship (Figure 2.1.1, top). A generator connected to the main engine, or an auxiliary engine, provides electricity for onboard systems. By contrast an electric ship has its engine connected only to a generator; that generator provides electricity for onboard systems but also supplies current to an electric motor housed in an external pod at the stern of the ship (Figure 2.1.1, bottom). The pod can rotate about its vertical axis in an azimuthing motion. Instead of setting a rudder angle to turn the ship, with podded propulsion one sets the azimuth angle of the entire pod. Electric ship systems can be more efficient than conventional systems because energy losses in the generator and the electric motor can be significantly less than energy losses in a gearbox and shaft bearings. However there are losses associated with the drag of the pod and the extra energy required to rotate the pod in contrast to the energy required

to turn a rudder.

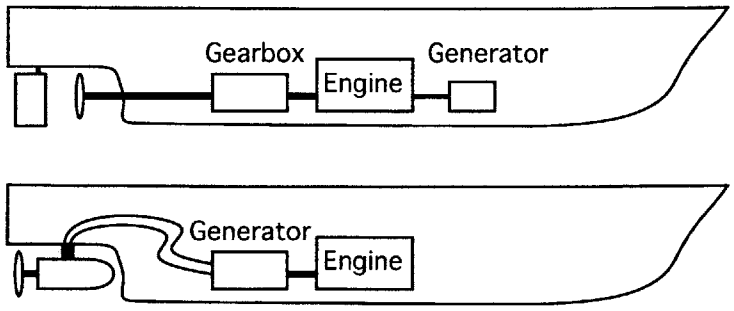

Figure 2.1.1: Top: a conventional ship propulsion system. An engine drives the propeller shaft and a generator for onboard electrical systems. The ship is steered with a rudder. Bottom: an electric ship propulsion system. An engine drives a generator that supplies current to an electric motor housed in an external pod. The ship is steered by rotating the pod about its vertical axis.

This is a very basic view of electric ships; in practice there are multiple engines and generators, electrical isolation between the propulsion current and onboard electrical systems, and more varied arrangements of pods throughout the hull.

2.1.1 Definition

Podded propulsion is defined as a system in which an electric motor directly drives a propeller from inside an external housing, but for the purposes of this research it is defined as any system in which steering is achieved by rotating the propulsion unit rather than using a fixed propeller shaft and a rudder. This expands the term to include azimuthing thrusters whose torque source is inside the hull and even outboard engines and stern drives on small craft.

A rudder is a lifting surface that generates a normal force when inclined at an angle of attack to the incident flow caused by ship motion or propwash. If the vessel is moving straight ahead then the lift force is perpendicular to the main axis of the ship (Figure 2.1.2). A yaw moment is created from two effects: the distance between the rudder sideforce and the ship center of gravity, and the vessel hydrodynamics when the rudder force causes sideslip. The sideforce and moment are proportional to the square of the local fluid velocity relative to the rudder. This means that no force or moment is generated when the ship is stationary. Furthermore there is a stall limit on the angle of attack, typically around 15°

for high aspect ratio foils, past which point the rudder generates far less force.

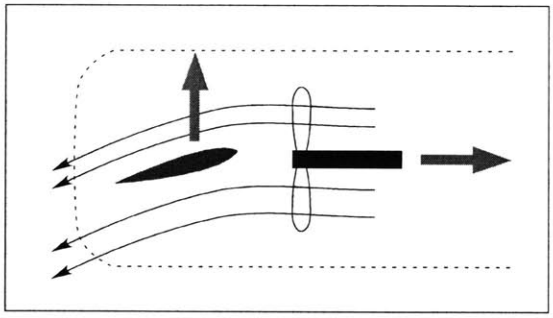

Figure 2.1.2: A rudder generates a lift force perpendicular to the main axis of the ship. This causes the vessel to yaw about the center of gravity.

An azimuthing podded propulsor generates thrust in the direction in which it is pointing (Figure 2.1.3). There is no stall limit so the propulsor can be rotated $\pm 180°$. The maximum sideforce and yaw moment are generated when the propulsor is at an azimuth angle of $\pm 90°$. However the propulsion unit has very high drag when in any orientation except straight ahead. For this reason the propulsor would not typically be turned sideways in normal cruising operation. A propulsion pod may have the propeller at the back of the pod, at the front, or one in each location.

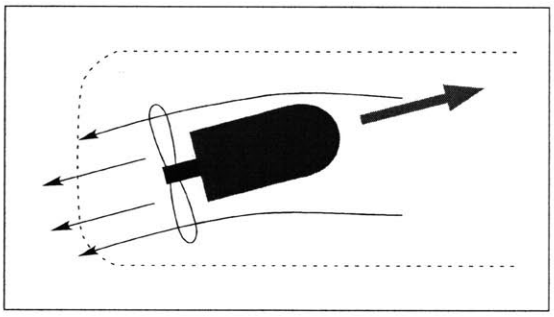

Figure 2.1.3: An azimuthing podded propulsor generates thrust in whichever direction it is pointing, regardless of the ship speed.

There are, of course, variations on both of these configurations. Rudders and other appendages may be placed anywhere on the ship hull, propellers may be ducted, and propulsion pods may have streamlined rudder-like fairings, for example.

2.1.2 Advantages and Limitations

The obvious advantage of azimuthing podded propulsion over a similarly-sized rudder steering system is that a podded propulsion system can generate much higher side forces and yaw moments. When the propulsor is turned completely sideways the entire propeller thrust pushes sideways as well. With higher sideforce comes higher yaw rates which allow for faster and more precise maneuvering in restricted environments. The propulsor can be used for dynamic positioning or to initiate a maneuver from a standstill.

A comparison of Figures 2.1.2 and 2.1.3 reveals that podded propulsors are necessarily larger than rudders and have more drag, especially when oblique to the local flow. While a ship with podded propulsion may be able to turn quickly it will suffer a substantial speed loss due to the lack of forward thrust and increased drag as the propulsor angle gets large. Even for small heading corrections the motion and orientation of the podded propulsor will cause significantly more drag than a rudder. In fact at small angles the rudder may produce more sideforce than the podded propulsor and do so with much less drag. For this reason it has been suggested that rudders be used in normal cruising operation even on vessels with podded propulsion [6].

Yaw moment for steering is created by a sideforce at the rudder or propulsor. The force acts a distance away from the center of gravity so a moment is applied to the ship causing it to turn. However the sideforce also pushes the vessel to the side; this contributes to an additional moment due to the hydrodynamics of the hull. Most vessels have the rudder at the aft end of the ship so that a force to the left must be applied to cause the ship to turn right and vice versa. This means that the ship is actually pushed to the left during a right-hand turn; this "reverse sideslip" is known as non-minimum phase behavior of the system. If the sideforce is larger then then reverse sideslip is larger. Applied to podded propulsion this means that the vessel will skid to the side during high-rate turns. This effect is also present for vessels that steer with rudders but it is less significant because the sideforce is limited by the stall angle of the rudder. A comparison is presented in Figure 2.1.4.

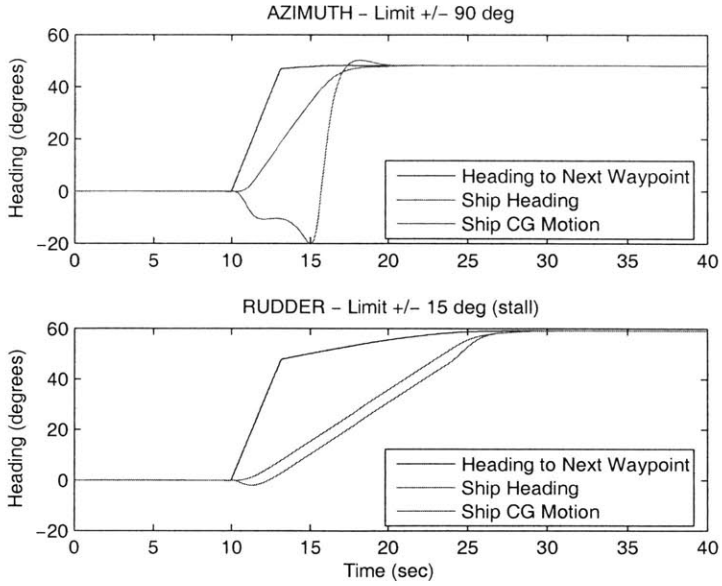

Figure 2.1.4: These plots show the direction of motion of the center of gravity of a ship during a heading change. The center of gravity is pushed in the direction opposite the turn. This effect is much more significant when the sideforce is greater, as with podded propulsion. Note the increased speed of the heading change with podded propulsion.

2.2 Maneuvering Control Systems for Ships

Automatic control of either heading or position falls into the general class of regulation problems that can be dealt with using any one of a number of linear or nonlinear control techniques. Heading controllers have been very well researched, being covered by linear PID control, sliding mode control, nonlinear backstepping from Lyapunov functions [9], adaptive control, fuzzy control [16], and genetic algorithms [11]. Heading controllers are at the core of many path following systems so the depth of their study is warranted.

2.2.1 P / PD / PID Heading Control

Ship heading control is a fairly straightforward regulation problem. Vessel dynamics can be linearized about an operating speed; with linearization comes a large set of powerful analysis and design tools for automatic control systems. The simplest form of linear control is proportional (P) control. With P control the rudder or azimuth angle is set to be proportional to the error between the actual vessel heading and the desired heading. Increasing

the proportional gain increases the speed of response. A proportional controller is often sufficient for this type of system in which the vessel has natural yaw damping (drag). However if the proportional gain is large and there are undesirable oscillations in the response then an additional damping effect can be added using derivative (PD) control. However when the error is small and steady both the proportional and derivative terms drop towards zero and it is difficult to get rid of the error entirely. If there is a steady disturbance then PD control will not be effective in removing its effects. Adding integral control (PID) removes this steady error because the integral of steady error grows large over time.

PID control is one of the most widely used forms of control in all applications. The design task reduces to choosing the three gains of the controller for the particular application. There exist heuristic methods of tuning the gains based on rise time, overshoot, and steady error rejection. PID control is often used to stabilize even nonlinear systems so it is a reasonable choice for heading control of ships, although there are dedicated nonlinear control designs for heading autopilots as well.

2.2.2 Inner Loop / Outer Loop Path Following Control

There are many heading controller autopilots on the market today for vessels of all sizes, from oil tankers to sailing yachts. Their ubiquity, simplicity, and effectiveness make them a valuable asset for any sort of path-following control. If the heading controller is robust enough to handle any reasonable desired heading input then it is straightforward to design a path-following controller that measures the vessel position, orientation, and velocities and generates a desired heading output. This is inner loop / outer loop control in which the inner loop is the heading controller and the outer loop is the path-following controller (Figure 2.2.1, top). The path input to the path controller may come from a higher-level mission planner or simply a set of waypoints. There is also a rudder controller that takes the rudder command from the heading controller and generates a gearmotor torque or piston pressure to actuate the rudder.

An advantage to this system is that it is modular and there is the flexibility to replace the heading controller or path-following controller independently. Of course, stability should always be examined whenever control systems are combined in this fashion. When the path controller is switched off the heading controller can be used alone.

An alternative is to have a single controller that measures the vessel position, heading,

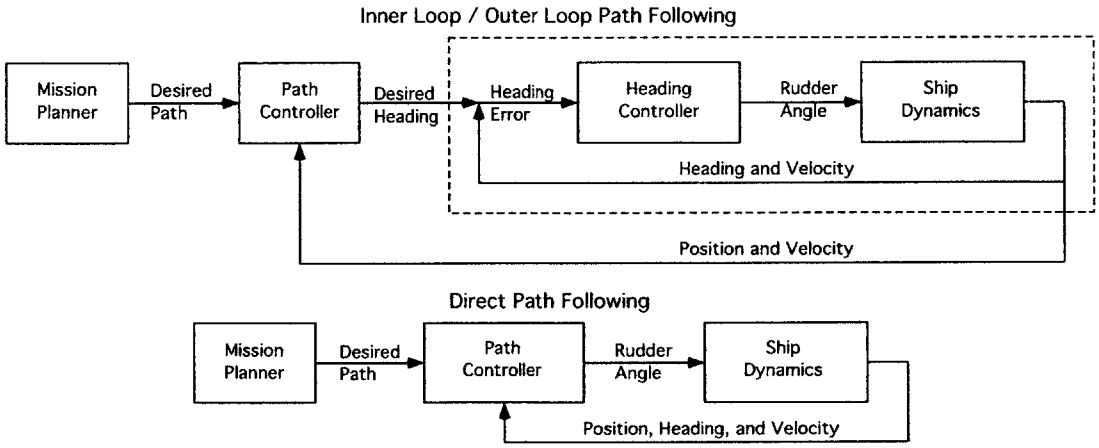

Figure 2.2.1: Top: A heading controller is embedded inside a path controller. The path controller generates a desired heading and the heading controller generates a desired rudder angle. Bottom: A path controller generates a desired rudder angle directly. Adapted from [5]

and velocity to generate the desired rudder angle directly (Figure 2.2.1, bottom). This system may perform better than an inner loop / outer loop system because there is no unnecessary desired heading signal. The drawback is that the design may be more complex, less universal, and more difficult to tune heuristically. Specifically the controller must explicitly handle the non-minimum phase behavior of the system.

2.3 Path Following Algorithms

Path following is a universal problem in robotics and automatic control. Satellites, rovers, aircraft, ships, underwater vehicles, and even machine tools must be able to follow prescribed paths in a stable and efficient manner. Many techniques have been applied to the path following control problem: linear control, nonlinear control, sliding control, adaptive control, and genetic algorithms, among others.

There are two main classes of path following control. Simple path following requires only that the vehicle stay on a specified geometric path with no timing restrictions implied. Trajectory control imposes timing constraints; the vehicle must track a reference point that moves along the path in time [1]. The research presented here focuses on simple path following with no timing constraints.

A typical robot arm has as many control inputs (joint torques) as degrees of freedom

(joints). A thoroughly-designed underwater vehicle has at least two thrusters pointing in each direction so that by applying the right amount of power to the appropriate thrusters any motion in 6 degrees of freedom can be obtained. Some surface vessels, in particular those vessels designed for dynamic positioning, have a full complement of thrusters. A vessel with only a rudder or an azimuthing pod has two control inputs (thrust and rudder / azimuth direction) and at least three degrees of freedom (surge, sway, and yaw). The system is therefore underactuated and it is impossible to, for example, maintain a constant forward speed and heading yet vary the sway velocity. This makes path following slightly more challenging: in general sway and yaw are coupled, sometimes very strongly (as in Figure 2.1.4). This means that path error can only be corrected by changing the heading angle and a heading error can only be corrected by initiating some sideslip. In fact ships with rudders aft are non-minimum phase systems, so path error can only by corrected by temporarily making the error worse. This imposes a bandwidth limit on the controller and affects overall response time. These are challenges not always present in other forms of path following control.

2.3.1 Line-Of-Sight Path Following for Ships

Despite these difficulties it is quite easy to get a ship to follow a simple path such as a set of line segments connecting waypoints. A logical first choice is to always aim for the waypoint, then switch to the next waypoint when the vessel approaches within a certain distance d_{switch} of the original waypoint (Figure 2.3.1). This algorithm, known as line-of-sight tracking, is very widely used for its simplicity.

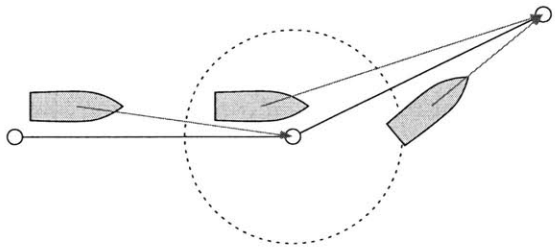

Figure 2.3.1: Line-of-sight waypoint tracking: always aim for the next waypoint.

A very simple addition to the algorithm makes it much more effective: instead of aiming for the waypoint, aim for a point along the path that is two or three boatlengths ahead.

This algorithm, referred to here as interpolated tracking, results in much more precise path following than simple line-of-sight, and it is the most common waypoint tracking algorithm in the literature [12, 4].

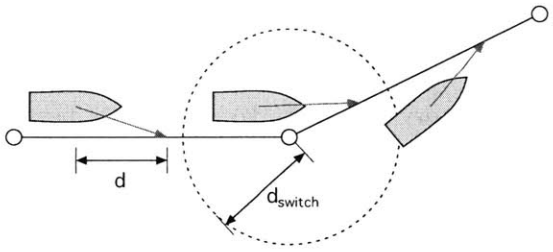

Figure 2.3.2: Interpolated tracking, the most common variant of line-of-sight waypoint tracking: always aim for a point d meters ahead on the path. Typically d is 2 or 3 boatlengths.

Path following for straight paths is rarely more sophisticated than the interpolated tracking algorithm. Variations include adding terms to cancel steady environmental forces and currents [10], smoothing the path around the waypoints [2, 14], and following a constant-radius trajectory at course changes [8].

The switching distance d_{switch} is generally picked from experience, from simulation, or adaptively. Adaptive algorithms are either online or in the form of offline genetic algorithms [11].

2.3.2 Path Following for Smooth and Non-Smooth Paths

In most cases the path following algorithms presented above can be directly applied to smooth paths such as circles or splines that are fitted to the waypoints or other constraints [10]. However new control systems have been developed for general continuous paths in which the local path curvature is taken into consideration when computing the desired heading [3]. These control systems share applications with wheeled robots and aircraft.

Ideal paths for general ship or marine vehicle navigation are generally smooth and they vary more slowly than the speed of the vehicle dynamics. Even at waypoints a smooth circular path is generated and the vessel follows the circular path instead. The center of the circle can be placed at the waypoint (the ship trajectory is outside of the waypoint), the edge of the circle can be placed at the waypoint (the ship trajectory passes through the waypoint) or the circle can be tangent to the straight paths entering and leaving the

waypoint (the ship trajectory is on the inside of the waypoint) [9]. The third method is most common in the literature.

However there are cases where a slow, smooth path is not ideal. Consider the surveying or mapping search grid in Figure 2.3.3. A smooth linear / circular path is easy for a normal ship to follow using any of the path following controllers mentioned above, and with careful controller design the desired trajectory can be tracked exactly (or nearly exactly). However to fill out this search grid the vessel would need to skip every other row on the way down and fill them in on a second pass. A vessel capable of high maneuverability can survey all rows in one pass. Note that the vessel trajectory at the corners does not exactly track the rectangular path. This can be tolerated as long as the vessel returns to the straight pass when it re-enters the search grid.

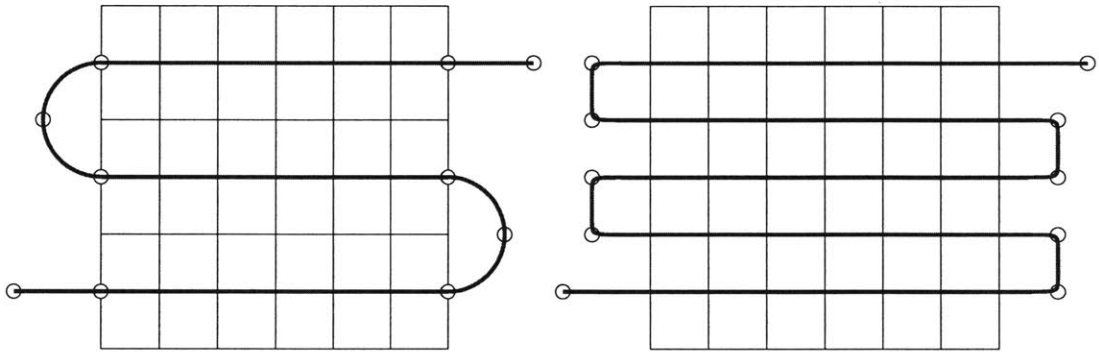

Figure 2.3.3: Left: A smooth waypoint scheme cannot reach every row of the survey grid. Right: a non-smooth path can be used with a vessel with high yaw rate dynamics, such as a vessel with azimuthing podded propulsion, to reach every row.

Another case where non-smooth paths may be tolerated and even required is navigation in a narrow riverway or channel. If the channel is too narrow to accept a path smooth enough that exact path following can be achieved, then sliding motions induced by high sideforce can be used to guide the vessel around banks or obstacles.

Figure 2.3.4 shows the path-following performance of a simulated vessel with a rudder and with an azimuthing propulsor. In each case the feedforward algorithm is used; this algorithm is described in Chapter 5. The ruddered vessel must cut the 60° corner but the vessel with podded propulsion can get much closer to the waypoint and the total path error is less. Note that both vessels exhibit reverse sideslip, but this effect is much greater for the vessel with podded propulsion.

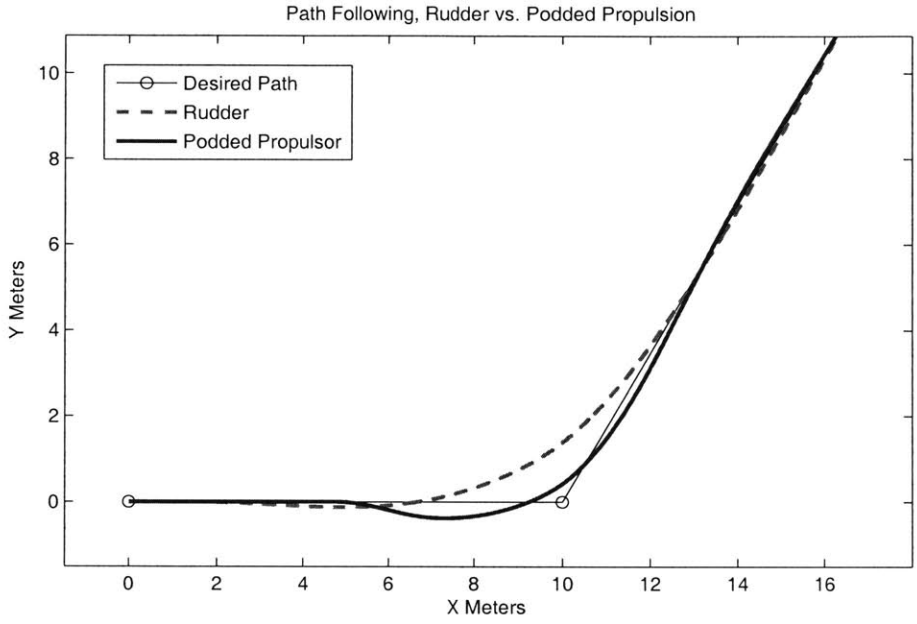

Figure 2.3.4: Path-following performance at a 60° course change. Both vessels use the feedforward algorithm from Chapter 5, and each is optimized to generate the smallest error. The vessel with podded propulsion can get much closer to the waypoint with less total error than the vessel with a rudder.

Chapter 3

Autonomous Kayak Test Platform

Experimental tests were anticipated from the outset of the project. A previous student, Jeff Stettler, built an autonomous surface vessel to perform research on azimuthing podded propulsion, autonomous control, and various maneuvering problems [15]. His vessel was available for use with this project. The vessel is a 12-foot kayak with an onboard computer for autonomous operation and communication with the shore. It has a full sensor suite and a propulsion system instrumented to be capable of full closed-loop control. Stettler obtained propeller curves for the propulsor at the full 360° range of azimuth angles and nonlinear hydrodynamic coefficients from Planar Motion Mechanism experiments.

The physical, electrical, and control systems were simulated in Mathworks' MATLAB and Simulink software so that the dynamics and control systems could be analyzed off-line. MATLAB's xPC Target software was used to implement the control systems on the kayak and establish communications between the onboard computer and the onshore computer.

This chapter describes the physical system, the computer and sensor equipment, the communication schemes, and the simulation.

3.1 Overview

The overall system is depicted in Figure 3.1.1. The kayak is 12 feet long and it is divided into three sections by two plywood bulkheads. The propulsor, an off-the-shelf electric trolling motor, extends below the aft section of the kayak. The propulsor angle is controlled with an azimuth gearmotor. The main compartment of the kayak contains the power electronics, the PC-104 computer system, the inertial measurement unit (DMU), and the GPS. The

compass is far away from any other electrical equipment in the bow of the vessel. The vessel can be controlled from the shore with a radio transmitter and onboard receiver. The details of the system are described in the following sections.

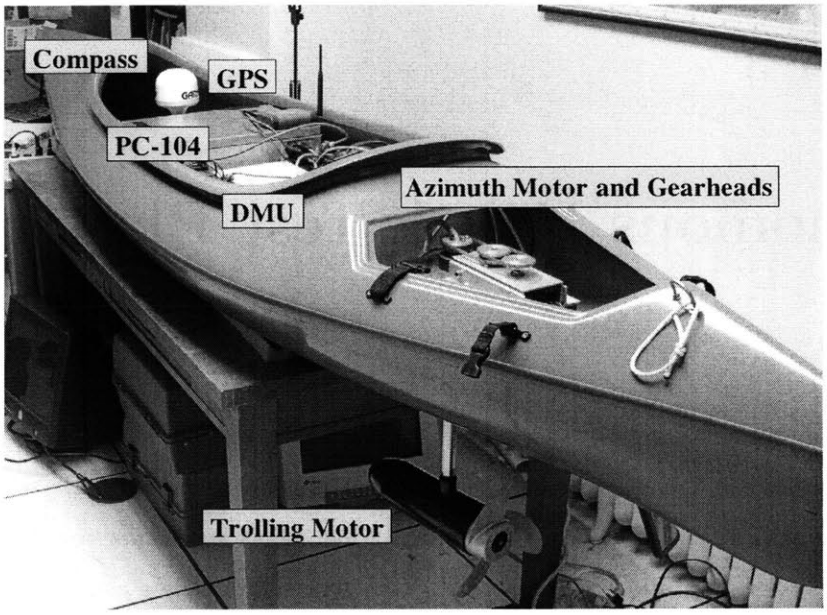

Figure 3.1.1: General layout of the components in the autonomous kayak.

3.2 Hull

The vessel hull is a Wilderness Systems Chesapeake Pro sporting kayak. It is 3.72 meters long with a 0.70 meter beam. Stettler added two watertight plywood bulkheads at the fore and aft of the cockpit for strength and to keep water leakage compartmentalized. A full-length internal fiberglass keel was added to improve longitudinal stiffness. A wooden tower was constructed in the aft section to mount the azimuth gearmotor and the vertical shaft of the propulsor. The stuffing box for the vertical shaft extends well above the waterline so that a watertight bearing is unnecessary.

The center section of the kayak contains the onboard computer, lead-acid batteries, power amplifiers, and sensors. These components are secured to a removable plywood deck plate. Barbell weights fit between the hull and the bottom of the deck plate for ballast. The total ballasted mass of the vessel is 122.5 kg.

3.3 Propulsion System

The kayak is powered by a single thruster located 72.6 cm aft of the center of gravity and far enough below the hull that interaction effects are negligible. The propulsor rotates about its vertical axis by means of an azimuth gearmotor. The propulsor can be set to any angle but in general operation it is limited to $\pm 90°$. The propeller is driven in open loop without feedback on propeller speed or vessel speed, but it is instrumented to allow for closed loop RPM control in the future.

3.3.1 Thruster

The thruster is a Motorguide ET54 trolling motor which can deliver a maximum of 54 pounds of thrust. It uses a 3-bladed Machete II aluminum propeller. Eight magnets with alternating polarity are glued to the inside of the propeller hub. An Allegro Microsystems UGN-3175U Hall effect sensor on the back face of the trolling motor body detects the changing magnetic field as the propeller rotates. These changes are measured by the onboard computer and used to compute the propeller rotation rate. The thruster body is 13 inches long and 3.625 inches in diameter. The propeller is 9.875 inches in diameter.

The trolling motor's vertical shaft was replaced with a 1-inch diameter stainless steel tube. The tube enters the hull through a Delrin bushing. A Pittman GM9234S031 servo gearmotor connects to the top of the tube through a toothed belt-and-pulley system. The gearmotor is also connected to a Vishay-Spectrol 601-1045 angle sensor. All the pulleys are the same diameter so that the gearmotor, the vertical propulsor shaft, and the angle sensor move together at the same rate.

3.3.2 Power System

Electric power for all of the kayak's systems come from two 12-volt, 35 amp-hour lead acid batteries connected in series. The 24 volts (nominal) goes through a fuse before splitting to the onboard computer, the gearmotor amplifier, and the propulsion amplifier. The gearmotor amplifier is a Copley Controls 413CE DC brush servo amplifier. A reference voltage from the onboard computer sets the output current. The propulsion amplifier is an Advanced Motion Controls 50A8 servo amplifier. This amplifier has a peak continuous current of ± 25 amps. All wiring from between the batteries and the propulsion motor is 12

gauge or larger.

3.4 Sensors

There are many sensors on the kayak but only three are used in closed-loop control: the GPS for position, the magnetometer for heading, and the DMU for yaw rate and compass corrections. A Kalman filter is used to filter all of the data at once using a linearized model of the vessel dynamics. The coordinate system used by the sensors and the control systems is shown in Figure 3.4.1.

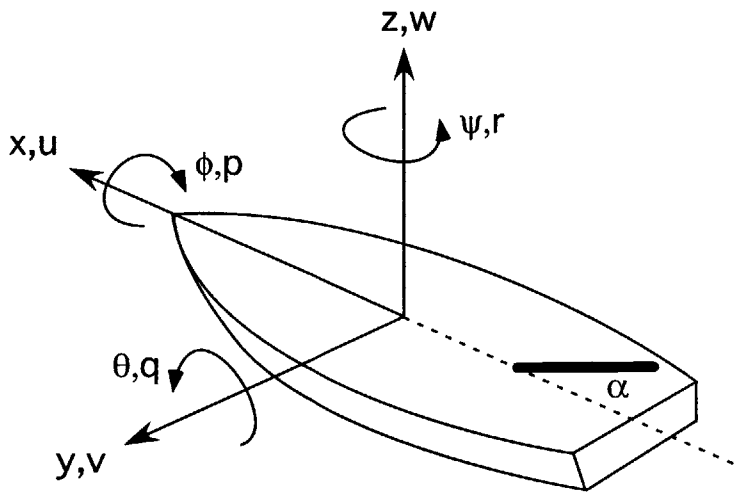

Figure 3.4.1: Coordinate system: first letter is position, second is velocity. y is to port, and the propulsor/rudder generates a sideforce in the positive y direction when the propulsor/rudder angle α is positive. Turning to port is positive in r.

3.4.1 DMU

The Crossbow VGX-300 inertial measurement unit (DMU) has three linear accelerometers ($\ddot{x}$, $\ddot{y}$, and $\ddot{z}$) and three rate-sensing gyroscopes (p, q, and r). An internal algorithm determines pitch and roll angles based on the integrated gyroscope readings and the accelerometer components. The gyroscopes are subject to bias and must be zeroed before any useful pitch or roll data can be acquired. It is assumed that on average the unit will be vertical and not subject to constantly-variable accelerations. Hence in the long term the accelerometers can be used to correct the integrated rotation rate data. The erection rate

is a settable parameter that determines how much the accelerometer data is used to adjust the pitch and roll values.

As soon as the unit is turned on it is zeroed and the erection rate is set through an RS-232 serial connection. All the data, however, is collected through analog channels. The data is computed internally at 400 Hz and it is sampled by the onboard computer at 50 Hz. The onboard computer reads x and y acceleration, roll rate p and yaw rate r, and roll and pitch angles ϕ and θ respectively. The unit's coordinate system is y-starboard, z-down, and θ bow up. The output values must be switched to the boat's y-port coordinate system once the data is read by the onboard computer.

Although the rate sensors are zeroed by the serial command signal, the analog signals sent to the computer are the raw unprocessed, biased data directly from the sensors. Therefore it must be separately zeroed in the onboard computer's program. When the computer program is started, yaw rate data is collected for the first ten seconds. The average measured yaw rate during this period is used to de-bias the rest of the data (Figure 3.4.2). For this to work the kayak must be stationary for the first ten seconds after the onboard computer program is started.

The acceleration data is not used in real time, nor is the roll rate data. The yaw rate data is passed through a Kalman filter and used by the heading controller. The pitch and roll angles are used to correct the magnetometer data and compute a valid heading angle. The noise variance of the yaw rate is 1.54×10^{-5} $(\text{rad/sec})^2$, which is equivalent to a standard deviation of $0.225°/\text{sec}$.

3.4.2 Magnetometer

Heading is computed using a combination of magnetometer data and tilt data from the DMU. The 3-dimensional vector of the earth's gravitational field relative to the kayak is measured by a 3-axis magnetometer. This vector is rotated to an earth-fixed orientation using the DMU's pitch and roll angles. Finally the horizontal components of the earth-fixed vector are used to determine the heading of the kayak.

Two rotation matrices are applied to an earth-frame vector to express it in terms of a coordinate system that is at a roll angle ϕ and a pitch angle θ. First, for roll rotation about

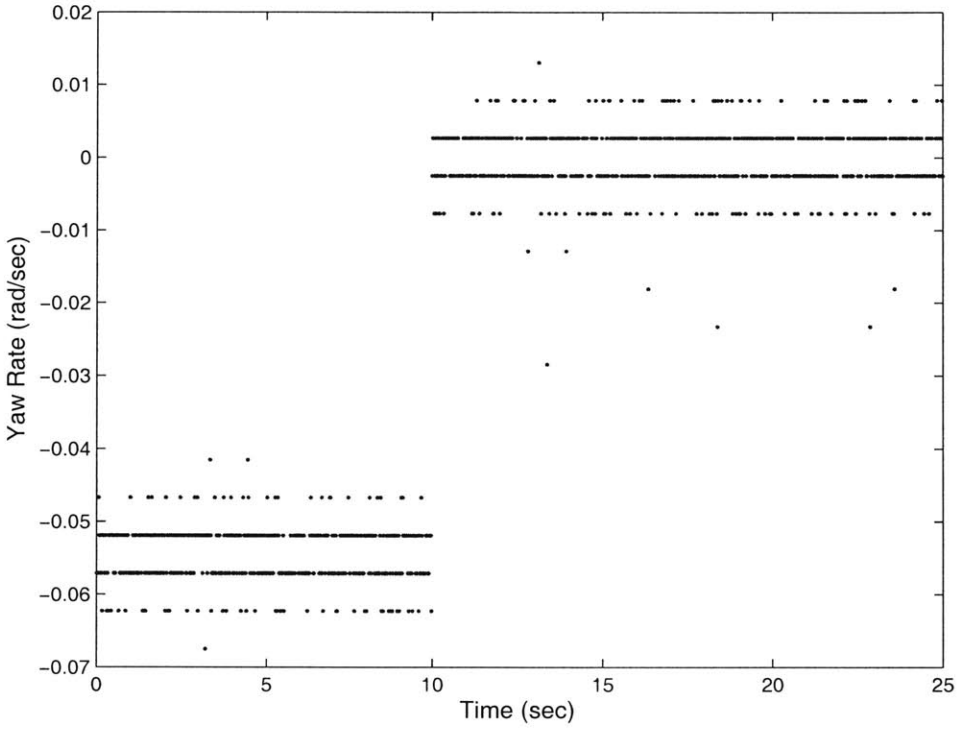

Figure 3.4.2: Raw yaw rate data collected in the onboard computer. The average is computed for the first ten seconds and is subtracted from the subsequent data in real time. The banding is due to the sensor's 0.3°/sec resolution. Data collected June 1, 2005.

the x-axis:

$$\mathbf{R_x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & \cos\phi & \cos\phi \end{bmatrix} \tag{3.4.1}$$

Next, for pitch rotation about the y-axis:

$$\mathbf{R_y} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3.4.2}$$

If the horizontal-frame components of the magnetic field form the column vector $\mathbf{F}$, then the components measured by the magnetometer in the kayak's frame are

$$\begin{aligned} \mathbf{f} &= \mathbf{R_x}\mathbf{R_y}\mathbf{F} \\ &= \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ \sin\phi\sin\theta & \cos\phi & \sin\phi\cos\theta \\ \cos\phi\sin\theta & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \mathbf{F} \end{aligned} \tag{3.4.3}$$

In practice we want to go the other way, so if we recall that the inverse of a rotation matrix is the same as its transpose, we get

$$\begin{aligned} \mathbf{F} &= (\mathbf{R_x}\mathbf{R_y})^T \mathbf{f} \\ &= \begin{bmatrix} \cos\theta & \sin\phi\sin\theta & \cos\phi\sin\theta \\ 0 & \cos\phi & -\sin\phi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \mathbf{f} \end{aligned} \tag{3.4.4}$$

The column vector $\mathbf{f}$ contains the x-, y-, and z-components of the measured magnetic field. The 3-by-3 matrix $(\mathbf{R_x}\mathbf{R_y})^T$ is computed at each time step from the DMU tilt data. The first two elements of the horizontal frame magnetic field vector $\mathbf{F}$ are relevant for heading calculations:

$$\psi = \tan^{-1} -\frac{F(2)}{F(1)} \tag{3.4.5}$$

The kayak uses the MicroMag3 3-axis magnetometer from PNI. This sensor sends magnetic field data from each of the three axes over an RS-232 serial interface at a rate of

11 samples (of all three axes) per second. The noise variance of the tilt-corrected data is 1.37×10^{-5} rad^2, which is equivalent to a standard deviation of 0.212°.

3.4.3 GPS

Position measurements are obtained using a Garmin GPS 18 5-Hz WAAS-enabled receiver. It sends position fixes to the onboard computer over a serial connection with a 5 Hz sample rate. The position resolution is 0.00001 minutes, which translates to 1.38 cm meters East and 1.86 cm meters North in Cambridge, MA. The signal is essentially noise free of high frequency noise, with only a slow walk on the order of a few centimeters per minute. Occasionally (around 5 minutes apart) the position measurement jumps by several meters. Sometimes this only lasts for one data point, and sometimes it is a steady correction for built-up random walk. These jumps occur with a low enough frequency that they can be ignored as long as the data sets are long enough and the data surrounding the jump is thrown away. The walk is slow enough that it is well below any frequency of the dynamics of the kayak. The walk over one minute is used to compute a noise covariance for the Kalman filter. Because the noise is essentially negligible, the Kalman filter reduces to a model-based extrapolation between 5 Hz data points.

There is negligible lag in the position signal. This was confirmed with a "shove test" in which the kayak was pushed forward about 1 meter from an initial rest state to a final rest state. The acceleration data from this test was integrated to give an inertial estimate of position; this was compared with the GPS position data. The results are shown in Figure 3.4.3. Given limitations of the accelerometer data due to drift (possibly due to a slightly different tilt orientation between the beginning and end of the test) there does not appear to be any lag in the GPS data.

3.5 Onboard Computer and MATLAB xPC Target Real-Time Embedded System

The onboard computer is a modular PC-104 computer system. The CPU is a Jumptec MOPSlcd6 166 MHz Pentium II MMX processor. A Tri-M Systems HE104 power module converts the unregulated 24 volt input to +12, +5, and -5 volt supplies. Sensors connect to a Computerboards DAS16JR/12 card which has 12 channels of analog input. A Diamond

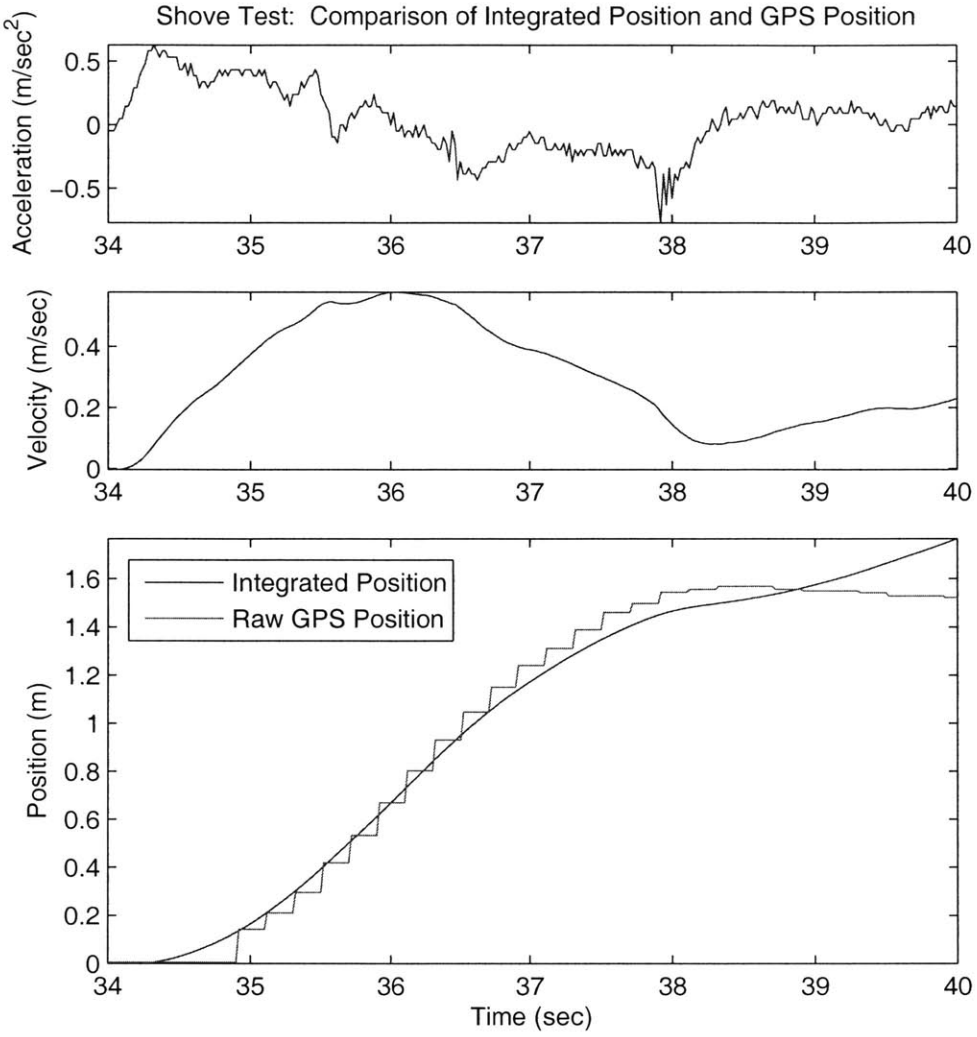

Figure 3.4.3: Determination of GPS lag from a "shove test" with motionless initial and final conditions. There is effectively no lag within the limits of drift of the accelerometer data.

Systems Ruby-MM card has 4 channels of analog output; two are used to control the Copley gearmotor amplifier and AMC propulsion motor amplifier. A Real Time Devices DM6804 counter module interfaces with the radio receiver, and a Real Time Devices CM202 NE2000 Ethernet card interfaces with the onshore computer.

The PC-104 stack is mounted in a sealed aluminum enclosure. This enclosure also contains a 3.5" floppy disk drive (used to load the xPC Target operating system), a 90 GB hard disk drive (used to store the sensor and parameter data), and a stack of PC boards that connect to the sensors and amplifiers.

3.5.1 xPC Target Description

xPC Target is a MATLAB environment used for real-time control of embedded systems. The development begins on a host PC, in this case the on-shore laptop, an IBM Thinkpad G40 with a 3 GHz Pentium 4 processor and 512 MB of RAM. A model of the inputs and outputs of the embedded system is created in MATLAB's Simulink block diagram interface. xPC Target includes Simulink blocks that correspond to a variety of data acquisition cards and controllers, including those in the PC-104 stack. The rest of the program is written in Simulink using a combination of stock or user-defined blocks, scripts, and functions. The Simulink model is then compiled on the host PC using MATLAB's Real Time Workshop. The result of the compilation is a downloadable model that runs on the target PC.

The target PC is any PC-based computer that can be booted from a floppy disk. In this case the target PC is the CPU of the onboard PC-104 stack. The target PC is booted from an xPC Target floppy disk that loads the xPC Target operating system. The target PC is connected to the host PC through an ethernet cable. The compiled model from the host PC is downloaded to the target PC. An interface on the host PC is used to load, start, and stop the model on the target PC. The host PC can be disconnected from the target PC while the model is running. Data collected during the run is either stored in RAM or on a hard disk drive. However, there is a limit to the number of data channels that can be written to the hard drive. In this application the most crucial data (raw sensor data and filtered states) is written to the hard drive and the rest of the data is stored in memory. If the target PC crashes or must be restarted then the most important data can still be retrieved.

3.5.2 Interfaces and Control

Diagnostic information and real-time data can be sent to the host PC while the target PC is connected and running its model. This data can be viewed using a custom graphical user interface (GUI) built in Simulink. Additionally, parameters in the model can be updated in real-time using the GUI. This theoretically permits live remote operation of the target system (the kayak) from the host system (the onshore laptop). However, this requires a strong ethernet connection or serial link. Originally the system was designed using a wireless router on the kayak attached to the onboard computer; the onshore laptop could communicate with the kayak through its built-in wireless capabilities. However, the range of this system was very limited (less than 100 ft) and unreliable. The baud rate of serial radio modems was too low to be feasible. The following alternative system was used (Figure 3.5.1).

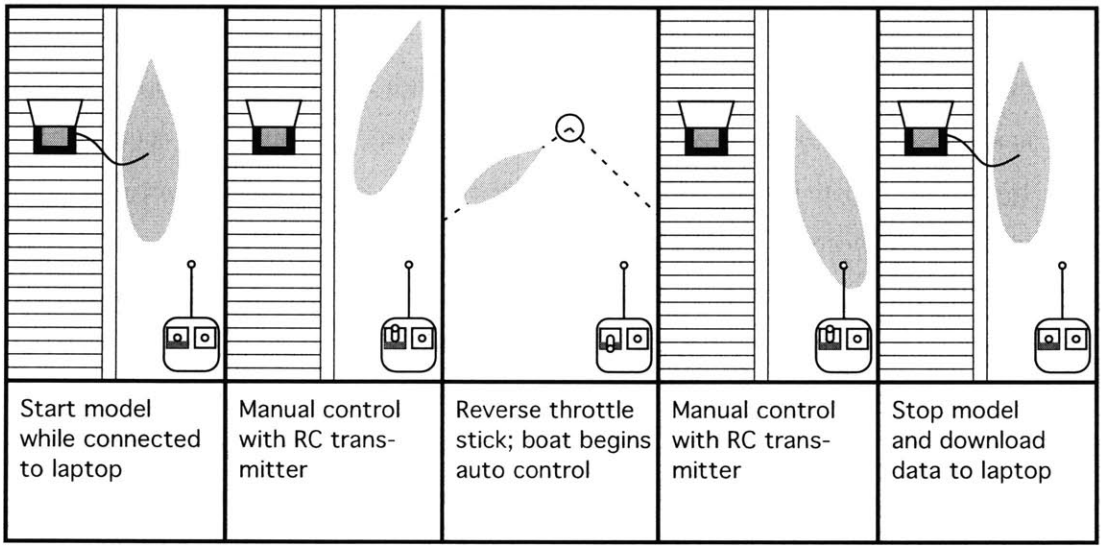

Figure 3.5.1: Procedures for launching and recovering the kayak.

When the kayak is at the dock it is connected to the onshore laptop with an ethernet cable. Through this link the model is downloaded to the onshore computer. The DMU is connected to the DMU/GPS serial port and it is power-cycled to reset pitch and roll drift. The model is started through the xPC Target interface on the laptop. Within the first ten seconds the onboard computer sends serial commands to the DMU to zero the rate gyros and set the erection rate. The kayak must be held steady during this period so that the

gyros are zeroed correctly. After the first ten seconds the DMU cable is disconnected from the serial port and the GPS cable is connected in its place. To ensure that the model is operating correctly, certain real-time data (tilt-compensated heading and azimuth angle) is viewed using a custom GUI. If everything is working correctly then the ethernet cable linking the laptop to the onboard computer is disconnected.

The vessel is now controlled using a 75 MHz radio control (RC) transmitter and receiver. The output of the receiver is one 50 Hz pulse-width modulated (PWM) signal per channel. Only two channels can be used due to a limitation on the number of PWM signals that the DM6804 counter board can process. Propulsion motor current is controlled using the throttle stick on the transmitter and azimuth angle is controlled using the rudder stick. To begin automatic control of the vessel, the throttle stick is put in reverse. As long as the onboard computer reads that the throttle is in reverse it runs whatever automatic control program is loaded into its model. If the throttle is set back to zero or the receiver loses the signal then the automatic control program stops. The kayak can then be operated under manual control until the throttle is set in reverse again. Data is collected the entire time that the model is running, both when it is under manual control and automatic control.

Once testing is complete the kayak is driven back to the dock under manual control. The ethernet cable is reconnected to the onboard computer and the model is stopped through the laptop's xPC Target interface. A script is then used to download the data from the RAM and hard drive of the onboard computer to the laptop.

The limitation of this procedure is that only one automatic control program can be loaded into the model at once, and to change the program one must recompile the model on the host PC. The advantage is that the procedure is very robust and it permits the operator to instantly take over manual control if the kayak is heading for a collision or the automatic control system drives the kayak to instability.

3.6 Determining Hydrodynamic Parameters

The remainder of this report is based on a detailed model of the hydrodynamic properties of the test kayak. The simulation is derived from a nonlinear model of the hydrodynamics of the hull and the force characteristics of the propulsion system, and the control systems design uses a largely linearized form of these equations. The full nonlinear hydrodynamic

model was obtained mostly from Planar Motion Mechanism tests of the bare hull, with adjustments to fit experimental data and empirical adjustments for the existence of the propulsor.

3.6.1 Planar Motion Mechanism Data

The basis for the nonlinear hydrodynamic model of the vessel is a set of Planar Motion Mechanism (PMM) tests performed by Jeff Stettler at the US Naval Academy [15]. The tests were performed on the bare hull. The data from those tests included the physical inertia of the vessel in the added mass terms $(X_{\dot{u}}' - m')$, $(Y_{\dot{v}}' - m')$, and $(N_{\dot{r}}' - I_z z')$, as well as the centrifugal terms $(X_{vr}' + m')$ and $(Y_r' - m'u')$. However, the mass and moment of inertia of the vessel were different for the PMM tests, so their effects had to be removed from the data so that the actual mass and moment of inertia could be used in the equations of motion. At the time of the tests the mass and moment of inertia were measured with the full weight of the PMM mount. The moment of inertia was measured using a bifilar pendulum rig, as described in Figure 3.6.1.

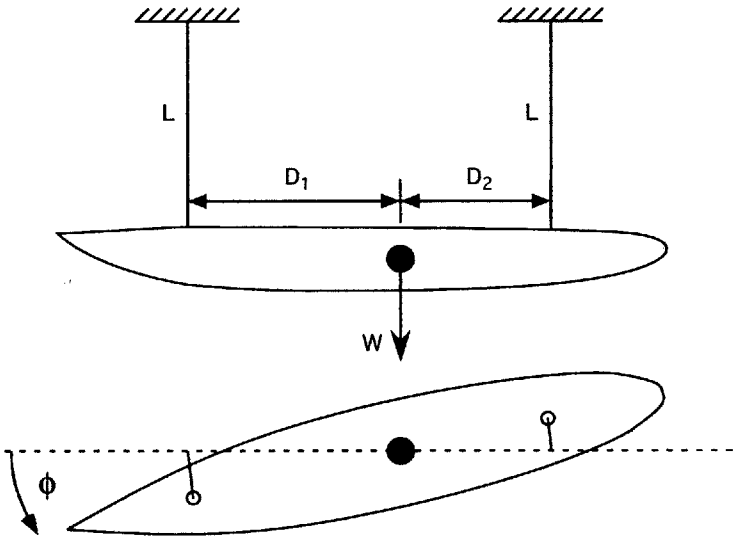

Figure 3.6.1: Bifilar pendulum test setup to determine the I_{zz} moment of inertia of the bare hull. The hull is suspended from the ceiling as shown and given an initial angular displacement ϕ. The moment of inertia can be computed from the period T of the resulting oscillations using the formula $I_{zz} = \frac{T^2 W}{4\pi^2 L}\left(\frac{{D_1}^2}{1+D_1/D_2} + \frac{{D_2}^2}{1+D_2/D_1}\right)$ where W is the weight of the bare hull.

The empty hull mass and moment of inertia were measured in separate experiments;

yet again a bifilar pendulum rig was used to measure the empty hull moment of inertia. The total mass and total moment of inertia were calculated by adding the effects of all the interior components such as the batteries, motor, computer, sensors, and ballast. 110 pounds of ballast was added to make the hull sit at the same waterline that was used in the PMM tests.

The PMM test were performed at two nominal speeds, 0.799 m/sec and 1.597 m/sec. For most of the low order terms the non-dimensional hydrodynamic coefficients at these two speeds differed by less than 20%, but the higher order terms had more variability, sometimes even changing sign. The nominal speed for this research is $u_o = 0.89$ m/sec which corresponds to 50% throttle (10 amps propulsion motor current). The non-dimensional hydrodynamic coefficients for this speed were found using linear interpolation between the two PMM test speeds. The coefficients were then dimensionalized using u_o.

3.6.2 River Tests

In December 2004 a sequence of open-loop tests were performed on the Charles River in Cambridge/Boston, MA to verify the hydrodynamic coefficients derived from the PMM tests. Tests included step changes in azimuth angle of 30°, 60°, and 90°. The yaw rate and GPS position were measured during these tests. These tests were recreated using the simulation (described in the next section) and the original PMM data. The simulated kayak was found to turn at a much faster rate than the real kayak.

A parameter search was performed in MATLAB to try to correct several parameters at once. The objective function for the minimization was the sum of the squared error from the yaw rate data and GPS position data from each of the three step azimuth tests. The search could not converge on a set of reasonable adjustments to the parameters. In the end a small set of parameters were adjusted manually to try to match the 30° and 60° data as well as possible.

Plots of the contributions from each of the nonlinear hydrodynamic factors revealed that none of the parameters are negligible in the 60° and 90° tests, although most of the higher order terms are of secondary importance in the 30° test. The N_{vrr} term accelerates the yaw rate unreasonably, so it was eliminated. The total X and Y mass terms (inertial plus added) were increased by 25%. These and other minor adjustments are listed in Table 3.6.1. These final hydrodynamic parameters result in good trajectory tracking; the yaw rates do not

match as well, but they represent a compromise between transient error and steady-state error. Figure 3.6.2 shows the actual and simulated yaw rates for the three tests, Figure 3.6.3 shows the trajectory in the 60° test, and Figure 3.6.4 shows the moment due to each of the linear and nonlinear terms for a 90° step azimuth change; all of the nonlinear terms have some non-negligible contribution.

Table 3.6.1: Parameter adjustments to match the simulation to river test data.

Parameter	$(X_{\dot{u}} - m)$	$(Y_{\dot{v}} - m)$	$(Y_r - mu_o)$	Y_{vvr}	Y_{vrr}	N_r	N_{ur}	N_{vrr}	N_{rrr}
Factor	1.25	1.25	-0.5	0.1	1.2	0.5	0.5	0	3

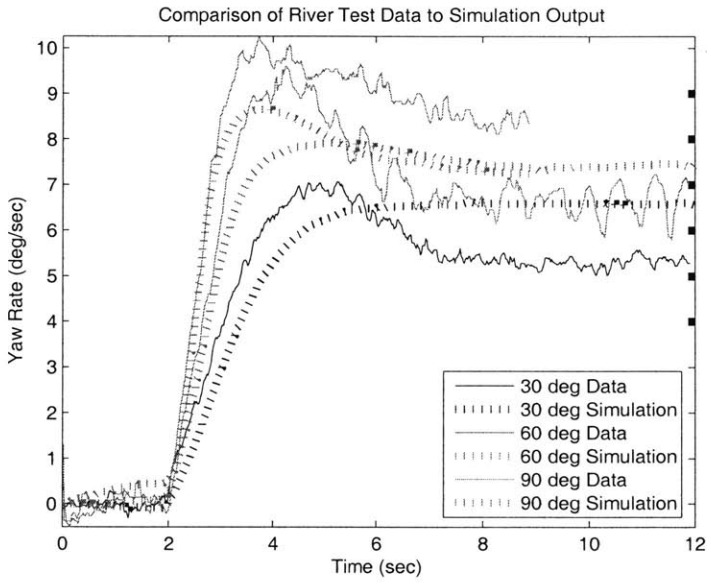

Figure 3.6.2: Measured and simulated yaw rate. The simulation parameters were optimized for the 30° and 60° tests. The yaw rates do not match as well as desired, but this could not be fixed using a manual search.

The final non-dimensional hydrodynamic parameters with corrections from the December 2004 river tests are listed in Table 3.6.2.

3.7 Thruster Forces and Dynamics

In addition to obtaining PMM data for the kayak, Jeff Stettler also developed a set of propeller curves for the propulsor operating at a complete set of azimuth angles from -180°

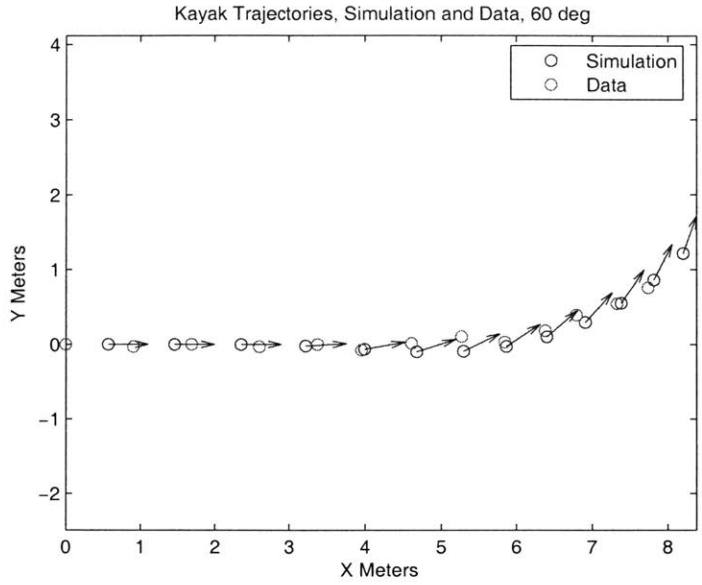

Figure 3.6.3: Measured and simulated GPS trajectory. The GPS unit used for these tests had an output rate of only 1 Hz. The arrows on the simulation points show the heading of the kayak.

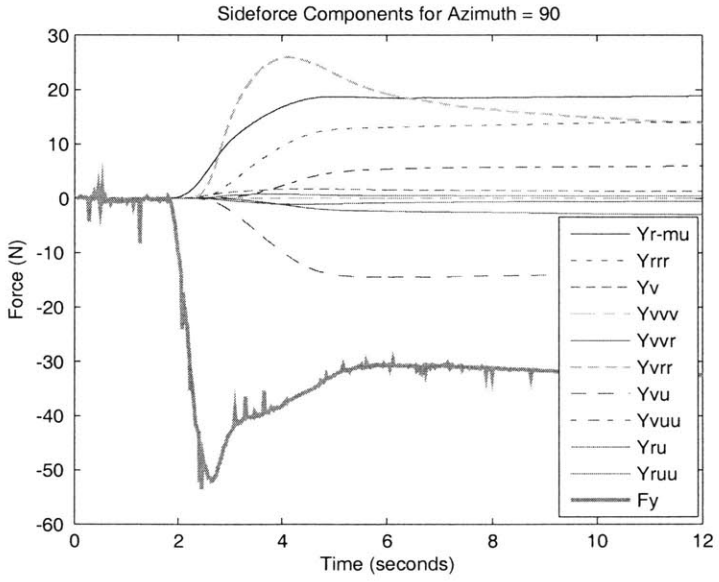

Figure 3.6.4: Contributions to the yaw moment from each of the nonlinear terms during a 90° step azimuth change. Most of these terms are non-negligible. The F_y input is computed from the propeller curves using the measured azimuth angle during the step. The noise comes from measurement noise; the unsteady behavior comes from the propeller dynamics.

Table 3.6.2: Non-dimensional hydrodynamic coefficients for the bare kayak hull, derived from PMM data and calibrated with river tests.

X coeff.	Non-dim. Value (x10^{-5})	Y coeff.	Non-dim. Value (x10^{-5})	N coeff.	Non-dim. Value (x10^{-5})
$X_{\dot{u}}'$	-496.2	$Y_{\dot{v}}'$	-778.7	$N_{\dot{r}}'$	-52.91
X_o'	-64.47	$Y_{\dot{r}}'$	-12.63	$N_{\dot{v}}'$	-9.148
X_u'	-120.8	Y_v'	-100.4	N_v'	-286.4
X_{uu}'	-57.56	Y_r'	526.5	N_r'	-90.47
X_{vv}'	-82.07	Y_{uv}'	-1249	N_{uv}'	-567.1
X_{vr}'	-210.1	Y_{ur}'	-673.6	N_{ur}'	-175.2
X_{rr}'	-48.51	Y_{uuv}'	-647.3	N_{uuv}'	-240.1
X_{uuu}'	-2.828	Y_{uur}'	-167.0	N_{uur}'	-195.0
X_{uvv}'	-496.4	Y_{vvv}'	-32.53	N_{vvv}'	-2142
X_{uvr}'	117.2	Y_{vvr}'	-298.3	N_{vvr}'	-874.6
X_{urr}'	-79.74	Y_{vrr}'	-4222	N_{vrr}'	0
		Y_{rrr}'	71.42	N_{rrr}'	-3436

to +180° [15]. This data is used directly in the simulation. The propeller speed, local fluid velocity, and azimuth angle are used to determine the advance coefficient J, the torque Q, the thrust T, and the normal force N in the propulsor-fixed coordinate system. The forces T and N are rotated to the body-fixed coordinate system forces F_X (axial thrust) and F_Y (the sideforce positive to port). The propeller speed used in the simulation is, for simplicity and speed of calculation, simply the nominal propeller speed at half throttle (10 amps propulsion motor current) and the straight-ahead cruising condition, 550 rpm. It was observed during the December 2004 river tests that the propeller speed does not change substantially even during hard turns.

Propulsor drag was modeled empirically. The vertical shaft was assumed to have a drag coefficient of 1.2 because at cruising speed it has a Reynolds number of 2.3×10^5. The main propulsor body was assumed to have an axial drag coefficient of 0.25 because of its higher Reynolds number of 2.4×10^6 and more streamlined form. The propulsor is less streamlined in the transverse direction, so a drag coefficient of 0.5 was used.

In the simulation a rudder was compared with the azimuthing podded propulsor. The data comes from a NACA 0015 foil tested at a full range of angles of attack [13]. The rudder size is approximately equal to the propeller disk area. The rudder has an initial lift slope of 0.8056 N/deg while the azimuthing propulsor has an initial lift slope of 0.5151 N/deg, which is 36% less. However, the rudder stalls by 15° while the azimuthing propulsor

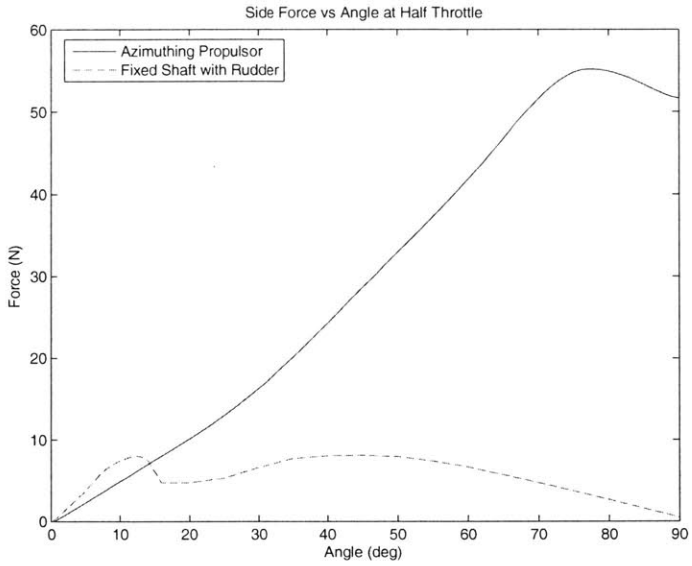

Figure 3.7.1: Side force generated by the azimuthing propulsor or a simulated rudder. The rudder in this case has a higher initial lift slope but a much lower range.

continues to generate a sideforce at much higher angles (Figure 3.7.1). But in the small-angle range the drag is much larger for the azimuthing propulsor. This difference would be even more pronounced if the rudder system used a standard propeller shaft instead of a podded propulsor, which has significantly more drag. Yet if the goal is to provide as much sideforce as possible, the azimuthing propulsor is clearly superior. The magnitude and direction of the total force vector over the range of angles for both systems is plotted in Figure 3.7.2. At zero angle both systems produce a net 4.3 N of thrust, which is what is required to overcome the hull drag. At all other angles, with constant motor current, the hull decelerates; this is much more pronounced with the azimuthing propulsor.

A PID controller is used to set the azimuth angle from a reference signal. This reference signal first passes through a second-order filter with two real poles at $s = -8$ to avoid allow the derivative component of the controller to operate correctly. The PID controller makes the actual angle follow the filtered signal very well, so in the simulation the azimuth dynamics are modeled simply as the filter dynamics, as below.

$$F_y = \frac{\lambda_1 \lambda_2}{(s+\lambda_1)(s+\lambda_2)} F_{ycmd} \tag{3.7.1}$$

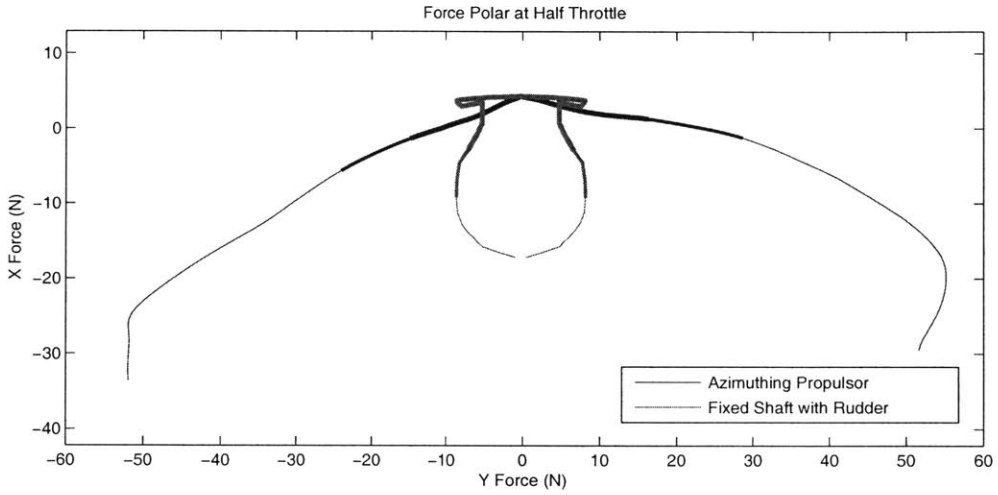

Figure 3.7.2: The direction and magnitude of the total force vector due to the azimuthing propulsor or propulsor with rudder (including both thrust and drag) as the propulsor and rudder sweep from -90° to +90°. The -45° to +45° range is indicated with the medium line and the -30° to +30° range is indicated with the thick line. The asymmetry of the plot is due to asymmetries in the propeller forces. Interaction effects with the vertical shaft cause an unbalanced side force when the propulsor is at a nonzero angle of attack relative to the local flow velocity.

In Equation 3.7.1 $\lambda_1 = \lambda_2 = 8$.

3.8 Summary

The control system design and simulation are based around the autonomous surface vessel used in the experiments. The vessel is a 12-foot kayak with an azimuthing trolling motor for propulsion. Although the vessel can cruise at around 1.6 m/sec, for the control design and analysis the vessel is operated at half-throttle with a nominal speed of 0.89 m/sec. The vessel motions are measured with a magnetometer for heading, a DMU for yaw rate and compass correction, and a GPS for position. An onboard computer based on the PC-104 architecture controls the vessel, measures the sensor data, and communicates with the shore-based laptop.

Chapter 4

Heading Control

The path following algorithm described in the next chapter has a heading controller at its core. The algorithm specifies a desired heading for the vessel based on the position and velocity of the vessel relative to the path. The heading controller drives the vessel heading to the desired heading as quickly as possible. The topic of this research is very precise path following, so the heading controller is required to be very fast even if it requires a substantial amount of control action (propulsor movement). Furthermore the path following algorithm does not generate a smooth, continuous desired heading near the waypoints. Instead there is a large jump in the desired heading and the vessel is expected to respond in a timely and stable manner.

As a result of the stringent requirements of the path following algorithm, the motions of the vessel are highly nonlinear. Consequently both linear and nonlinear heading controllers must be investigated to find the best possible controller. For this research only one nonlinear controller was studied: feedback linearization, in which the nonlinearities of the system are directly cancelled. This form of nonlinear control requires a well-identified system. Part of the last chapter was devoted to the process behind the system identification. The system is described by a set of nonlinear equations of motion. These equations are used almost directly to develop the nonlinear heading controller and they are linearized to develop the linear controller. The next section describes these equations in more detail.

4.1 Planar Equations of Motion

4.1.1 3rd Order Nonlinear Model

The full 3rd-order nonlinear equations of motion for a symmetric surface ship are listed below.

$$\begin{aligned} m(\dot{u} - vr - x_{cg}r^2) &= X_o + X_{\dot{u}}\dot{u} + X_u u + X_{uu}u^2 + X_{vv}v^2 + X_{vr}vr + X_{rr}r^2 \\ &\quad + X_{uuu}u^3 + X_{uvv}uv^2 + X_{uvr}uvr + X_{urr}ur^2 + F_x \qquad (4.1.1) \\ m(\dot{v} + ur + x_{cg}\dot{r}) &= Y_{\dot{v}}\dot{v} + Y_{\dot{r}}\dot{r} + Y_v v + Y_r r + Y_{uv}uv + Y_{ur}ur + Y_{uuv}u^2 v + Y_{uur}u^2 r \\ &\quad + Y_{vvv}v^3 + Y_{vvr}v^2 r + Y_{vrr}vr^2 + Y_{rrr}r^3 + F_y \qquad (4.1.2) \\ I_{zz}\dot{r} + mx_{cg}(\dot{v} + ur) &= N_{\dot{v}}\dot{v} + N_{\dot{r}}\dot{r} + N_v v + N_r r + N_{uv}uv + N_{ur}ur + N_{uuv}u^2 v + N_{uur}u^2 r \\ &\quad + N_{vvv}v^3 + N_{vvr}v^2 r + N_{vrr}vr^2 + N_{rrr}r^3 + X_p F_y + M \qquad (4.1.3) \end{aligned}$$

In these equations F_y is the sideforce generated by the propulsor (or rudder) and M is the moment about the propulsor; henceforth it will be assumed that $M = 0$ because it is a much less significant effect than the moment about the center of gravity $X_p F_y$. These full nonlinear equations are used in the simulation but they are of limited use for practical control, partly because it is so difficult to accurately determine the higher order nonlinear coefficients.

4.1.2 3rd Order Nonlinear Sway-Yaw Model

One simplification to the full 3rd-order nonlinear equations 4.1.1, 4.1.2, and 4.1.3 is to remove the surge effects and keep only the terms related to sway and yaw. This is only valid for small azimuth angles and yaw rates because of the significant drag associated with larger angles. However we retain the nonlinear sway-yaw model for feedback linearization control.

$$\begin{aligned} (m - Y_{\dot{v}})\dot{v} + (mx_{cg} - Y_{\dot{r}})\dot{r} &= Y_v v + (Y_r - mu_o)r + Y_{vvv}v^3 \\ &\quad + Y_{vvr}v^2 r + Y_{vrr}vr^2 + Y_{rrr}r^3 + F_y \qquad (4.1.4) \\ (mx_{cg} - N_{\dot{v}})\dot{v} + (I_z z - N_{\dot{r}})\dot{r} &= N_v v + N_r r + N_{vvv}v^3 \\ &\quad + N_{vvr}v^2 r + N_{vrr}vr^2 + N_{rrr}r^3 + X_p F_y \qquad (4.1.5) \end{aligned}$$

If the off-diagonal added mass terms are assumed to be small and the center of gravity is near the origin then the equations simplify to the more convenient expressions 4.1.6 and 4.1.7.

$$\begin{aligned}\dot{v} &= \frac{1}{m-Y_{\dot{v}}}\left[(Y_r - mu_o)r + Y_v v + Y_{vvv}v^3 + Y_{vvr}v^2 r\right. \\ &\quad \left. + Y_{vrr}vr^2 + Y_{rrr}r^3\right] + \frac{1}{m-Y_{\dot{v}}}F_y \qquad (4.1.6)\\ \dot{r} &= \frac{1}{I_{zz}-N_{\dot{r}}}\left[(N_r - mx_{cg}u_o)r + N_v v + N_{vvv}v^3 + N_{vvr}v^2 r\right. \\ &\quad \left. + N_{vrr}vr^2 + N_{rrr}r^3\right] + \frac{1}{I_{zz}-N_{\dot{r}}}X_p F_y \qquad (4.1.7)\end{aligned}$$

4.1.3 Linear Model

The equations 4.1.1, 4.1.2, and 4.1.3 can be linearized about the nominal speed u_o to greatly simplify controller design and analysis. The linear planar equations of motion are listed in 4.1.8, 4.1.9, 4.1.10.

$$(m - X_{\dot{u}})\dot{u} = X_u u + F_x \qquad (4.1.8)$$

$$(m - Y_{\dot{v}})\dot{v} + (mx_{cg} - Y_{\dot{r}})\dot{r} = Y_v v + Y_r r + F_y \qquad (4.1.9)$$

$$(mx_{cg} - N_{\dot{v}})\dot{v} + (I_z z - N_{\dot{r}})\dot{r} = N_v v + N_r r + X_p F_y \qquad (4.1.10)$$

Now that the system is linear we can represent it in matrix form

$$\mathbf{M}\dot{\mathbf{x}} = \mathbf{N}\mathbf{x} + \mathbf{B_o}F_y \qquad (4.1.11)$$

where $\mathbf{x} = \begin{bmatrix} u & v & r \end{bmatrix}^T$ and

$$\mathbf{M} = \begin{bmatrix} (m - X_{\dot{u}}) & 0 & 0 \\ 0 & (m - Y_{\dot{v}}) & (mx_{cg} - Y_{\dot{r}}) \\ 0 & (mx_{cg} - N_{\dot{v}}) & (I_{zz} - N_{\dot{r}}) \end{bmatrix}$$

$$\mathbf{N} = \begin{bmatrix} X_u & 0 & 0 \\ 0 & Y_v & Y_r \\ 0 & N_v & N_r \end{bmatrix} \qquad \mathbf{B_o} = \begin{bmatrix} 0 \\ 1 \\ X_p \end{bmatrix}$$

From here it is a simple transformation to the standard state space form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}F_y \tag{4.1.12}$$

using $\mathbf{A} = \mathbf{M^{-1}N}$ and $\mathbf{B} = \mathbf{M^{-1}B_o}$.

4.1.4 Linear Sway-Yaw Model

When the equations of motion are linearized, the u equation becomes decoupled from the v and r equations. It is a simple matter to rewrite the system in terms of v and r alone. At the same time we can add heading ψ as a useful output of the system; it is the integral of the yaw rate r. If a_{ij} and b_i are the components of the $\mathbf{A}$ and $\mathbf{B}$ matrices above, then the new state space system is

$$\begin{bmatrix} \dot{v} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} a_{22} & a_{23} & 0 \\ a_{32} & a_{33} & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} b_2 \\ b_3 \\ 0 \end{bmatrix} F_y \tag{4.1.13}$$

4.2 PD Control

There are many different methods for tuning PID gains, but the most common method in practical control applications is to tune them heuristically. Here we use a slightly more model-based approach. A PD controller is developed first and integral control is added at the end. Ignoring sway, the linear yaw equation of motion is:

$$(I_{zz} - N_{\dot{r}})\ddot{\psi} = N_r r + X_p F_y \tag{4.2.1}$$

We cannot realistically expect the ship to follow any arbitrary desired heading ψ_d, so we filter the input with a second-order filter to remove discontinuities. This filtered signal is the heading reference model ψ_m.

$$\psi_m = \frac{k_0}{s^2 + k_1 s + k_0}\,\psi_d \tag{4.2.2}$$

which can be rewritten as

$$\ddot{\psi}_m + k_1\dot{\psi}_m + k_o\psi_m = k_o\psi_d \tag{4.2.3}$$

We can now define a signal $z(t)$ as follows:

$$z(t) = \ddot{\psi}_m - k_1\dot{e} - k_0 e \tag{4.2.4}$$

where the error is measured with respect to the reference model, $e = \psi - \psi_m$. If there is no error then the actual heading matches the heading reference model and $z(t) = \ddot{\psi}$. Plugging this into Equation 4.2.1 and solving for the control input F_y we get:

$$F_y = \frac{I_{zz} - N_{\dot{r}}}{X_p} z(t) - \frac{N_r}{X_p} r \tag{4.2.5}$$

Combining $z(t) = \ddot{\psi}$ with Equation 4.2.4 results in

$$\ddot{\psi} = \ddot{\psi}_m - k_1(\dot{\psi} - \dot{\psi}_m) - k_o(\psi - \psi_m) \tag{4.2.6}$$

which simplifies to the error dynamics

$$\ddot{e} + k_1\dot{e} + k_o e = 0 \tag{4.2.7}$$

Here it becomes clear that the stability of the controller boils down to the choice of k_1 and k_0 in the reference model of Equation 4.2.2. A natural choice is a pair of real poles at $s = -\lambda$ so that the reference model becomes

$$\psi_m = \frac{\lambda^2}{s^2 + 2\lambda s + \lambda^2}\, \psi_d \tag{4.2.8}$$

Plugging Equation 4.2.4 into the control law of Equation 4.2.5 using the reference model of Equation 4.2.8 results in the final control law

$$\begin{aligned} F_y &= \frac{(I_{zz} - N_{\dot{r}})}{X_p} z(t) - \frac{N_r}{X_p} r \\ &= \frac{(I_{zz} - N_{\dot{r}})}{X_p} \left(\ddot{\psi}_m - 2\lambda(r - \dot{\psi}_m) - \lambda^2(\psi - \psi_m) \right) - \frac{N_r}{X_p} r \\ &= \frac{(I_{zz} - N_{\dot{r}})}{X_p} \left(\ddot{\psi}_m + 2\lambda\dot{\psi}_m + \lambda^2\psi_m - 2\lambda r - \lambda^2\psi \right) - \frac{N_r}{X_p} r \\ &= \frac{(I_{zz} - N_{\dot{r}})}{X_p} \left(\lambda^2\psi_d - 2\lambda r - \lambda^2\psi \right) - \frac{N_r}{X_p} r \\ &= -\left(\frac{2\lambda(I_{zz} - N_{\dot{r}})}{X_p} + \frac{N_r}{X_p} \right) r - \lambda^2 \frac{(I_{zz} - N_{\dot{r}})}{X_p} (\psi - \psi_d) \qquad (4.2.9) \\ F_y &= -K_{dH}\, r - K_{pH}(\psi - \psi_d) \qquad (4.2.10) \end{aligned}$$

This PD control law is tuned with a single parameter λ. Using $\lambda = 1$ provides quick dynamic response; the reference model reaches 90% of its final value in 4 seconds. Using this λ, $K_{pH} = -234.95$ N/rad and $K_{dH} = -379.91$ N/(rad/sec). There is quite a bit of control action using these gains but the controller does a very good job of rejecting disturbances. A quick heading controller is essential for tight path following as we will see in the next chapter.

4.3 Feedback Linearization Nonlinear Control

The main thrust of feedback linearization control is to directly cancel the nonlinearities of the system using feedforward terms in the control law, then stabilize the dynamics with a stable feedback model. The following derivation of the feedback linearization controller can be applied to more complicated system models than the one used in the final implementation of this controller. The nonlinear system can be written in the general SISO form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u \qquad (4.3.1)$$

$$y = \mathbf{h}(\mathbf{x}) \qquad (4.3.2)$$

where $\mathbf{x} = \begin{bmatrix} v & r & \psi \end{bmatrix}^T$, $y = \psi$, $u = F_y$, the size of $\mathbf{x}$ is $n = 3$, and the relative degree of the system is 2 (meaning that the output ψ must be differentiated twice for the input F_y to appear). The first two rows of Equation 4.3.1 come from Equations 4.1.6 and 4.1.7. We

will use the notation of a Lie derivative whose definition is

$$L_{\mathbf{a}}\, b(\mathbf{x}) \equiv \nabla b\, \mathbf{a} \tag{4.3.3}$$

The first step is to transform the state vector into normal coordinates μ; the size of μ is the same as the relative degree, which for this system is 2. A third variable $\mathbf{p}$ will be necessary to form a complete state transformation; its size is n minus the relative degree, in this case 1. Working from the output, we define

$$\mu_1 = h(\mathbf{x}) = \psi \tag{4.3.4}$$

$$\mu_2 = L_{\mathbf{f}}\, h(\mathbf{x}) = \nabla h(\mathbf{x})\, \mathbf{f} = r \tag{4.3.5}$$

For completeness we can also derive the third variable p in the transformed coordinates. To ensure that the third variable is independent from the other two we choose p such that

$$\begin{aligned} L_{\mathbf{g}}\, p &= 0 \\ &= \frac{\partial p}{\partial x_1} g_1 + \frac{\partial p}{\partial x_2} g_2 + \frac{\partial p}{\partial x_3} g_3 \\ &= \frac{\partial p}{\partial v} \frac{1}{(m - Y_{\dot{v}})} + \frac{\partial p}{\partial r} \frac{X_p}{(I_{zz} - N_{\dot{r}})} \end{aligned} \tag{4.3.6}$$

By inspection the solution to this partial differential equation is

$$p = (m - Y_{\dot{v}})v - \frac{(I_{zz} - N_{\dot{r}})}{X_p} r \tag{4.3.7}$$

Tracking control reduces to specifying a desired μ_d and driving the error $\tilde{\mu} = \mu - \mu_d = \begin{bmatrix} \psi - \psi_d \\ r - \dot{\psi}_d \end{bmatrix}$ to zero. For a system of relative degree 2 the control law has the form

$$u = \frac{1}{L_{\mathbf{g}}(L_{\mathbf{f}}\, \mu_1)} \left[-L_{\mathbf{f}}{}^2 \mu_1 + \ddot{\mu}_{d1} - k_1 \tilde{\mu}_2 - k_0 \tilde{\mu}_1 \right] \tag{4.3.8}$$

where, for our system,

$$\begin{aligned} L_{\mathbf{f}}{}^2\mu_1 &= L_{\mathbf{f}}\,(L_{\mathbf{f}}\,\psi) = L_{\mathbf{f}}\,r = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}\mathbf{f} \\ &= \frac{1}{I_{zz}-N_{\dot{r}}}\left[(N_r - mx_{cg}u_o)r + N_v v + N_{vvv}v^3\right. \\ &\qquad \left. + N_{vvr}v^2 r + N_{vrr}vr^2 + N_{rrr}r^3\right] \end{aligned} \tag{4.3.9}$$

Equation 4.3.9 is simply the part of Equation 4.1.7 not associated with the input F_y. Next we find

$$L_{\mathbf{g}}\,(L_{\mathbf{f}}\,\mu_1) = \nabla(\nabla\mu_1\,\mathbf{f})\,\mathbf{g} = \nabla r\,\mathbf{g} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}\mathbf{g} = \frac{X_p}{(I_{zz}-N_{\dot{r}})} \tag{4.3.10}$$

We combine Equations 4.3.9 and 4.3.10 to get the final control law

$$\begin{aligned} F_y &= \frac{(I_{zz}-N_{\dot{r}})}{X_p}\left[-\frac{1}{(I_{zz}-N_{\dot{r}})}\left((N_r - mx_{cg}u_o)r + N_v v + N_{vvv}v^3\right.\right. \\ &\qquad \left.\left. + N_{vvr}v^2 r + N_{vrr}vr^2 + N_{rrr}r^3\right) + \ddot{\psi}_d - k_1\dot{e} - k_o e\right] \\ &= -\frac{1}{X_p}\left[(N_r - mx_{cg}u_o)r + N_v v + N_{vvv}v^3 + N_{vvr}v^2 r + N_{vrr}vr^2 + N_{rrr}r^3\right] \\ &\qquad + \frac{(I_{zz}-N_{\dot{r}})}{X_p}\left[\ddot{\psi}_d - k_1\dot{e} - k_o e\right] \end{aligned} \tag{4.3.11}$$

Note that in this case the error is defined with respect to the desired heading, $e = \psi - \psi_d$, not the reference heading as in the PID control derivation. If the final control law of Equation 4.3.11 is plugged into the yaw dynamics of Equation 4.1.7 then the nonlinear terms are exactly canceled:

$$(I_{zz}-N_{\dot{r}})\dot{r} = (\text{NL terms}) - (\text{NL terms}) + (I_{zz}-N_{\dot{r}})(\ddot{\psi}_d - k_1\dot{e} - k_o e) \tag{4.3.12}$$

which reduces to the following error dynamics:

$$\ddot{e} + k_1\dot{e} + k_o e = 0 \tag{4.3.13}$$

Noting the similarity between Equations 4.2.7 and 4.3.13 we can choose $k_o = \lambda^2$ and $k_1 = 2\lambda$ as before.

4.4 Comparison of Linear PD/PID Control with Nonlinear Feedback Linearization Control

It is expected that, given the nonlinear nature of high-rate-of-turn dynamics and large propulsive sideforces, a nonlinear heading controller performs much better than a linear one. To a certain extent this is true. A controller that knows about nonlinearities can compensate for them; this is precisely the basis for feedback linearization control. The most noticeable effect is that the nonlinear controller prevents overshoot when the vessel heading is quickly approaching the desired heading in step response, for example. However, because both the linear controller and nonlinear controller are limited by the azimuth saturation angle, this effect is not especially pronounced. Figure 4.4.1 shows the step response simulation of the two controllers to a 60° heading change. The two controllers are very similar, but the PD controller has a small amount of overshoot while the FL controller backs off just before reaching the desired heading.

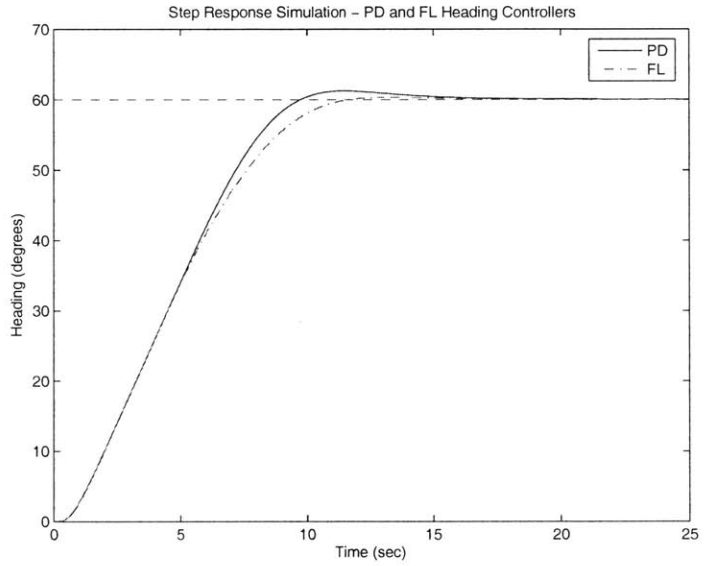

Figure 4.4.1: Step response simulation of the linear PD and nonlinear Feedback Linearization heading controllers to a 60° heading change. The nonlinear controller prevents overshoot, but overall the controllers perform similarly. Azimuth saturation is 80°.

These two heading controllers were tested in the kayak. They were found to be stable but there was a great deal of steady state error in both cases (Figures 4.4.2 and 4.4.3).

This is due either to environmental forces such as wind or waves or it is due to thruster misalignment. No matter what the cause, the steady-state error was unacceptable, so integral control was added to each controller.

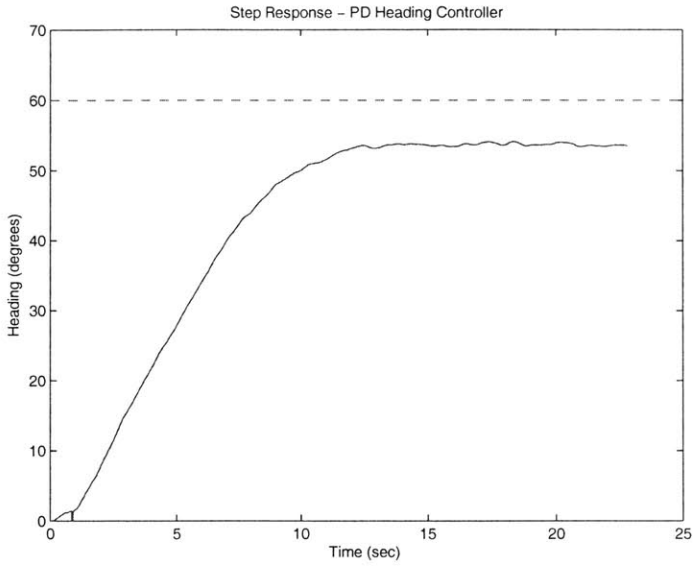

Figure 4.4.2: Step response of the linear PD controller to a 60° heading change. There is significant steady-state error. Azimuth saturation is 60°. Data taken November 9, 2005.

In a system with saturation limits on the system input it is often necessary to use an anti-windup scheme with the integrator. It is unreasonable to integrate the error of the controller while the propulsor is at its saturation limit. The simple fix is to stop adding to the integration variable when the commanded sideforce is greater than the sideforce that can be obtained at the saturation limit. For consistency the integrator cutoff was set at the 60° saturation limit whether the actual saturation limit was 60° or 80°. From Figures 4.4.2 and 4.4.3 it is clear that a large integral gain is required in the controller. A value of $K_{iH} = -40$ N/(rad-sec) was used for both controllers. A simulation of the controllers is presented in Figure 4.4.4.

The anti-windup scheme prevents most of the error during the approach to the desired heading from adding to the integration, but once the azimuth angle pulls away from its limit the integral term grows. The integral is reduced back to zero through overshoot, which both the linear and nonlinear controllers display. However the linear controller settles back to

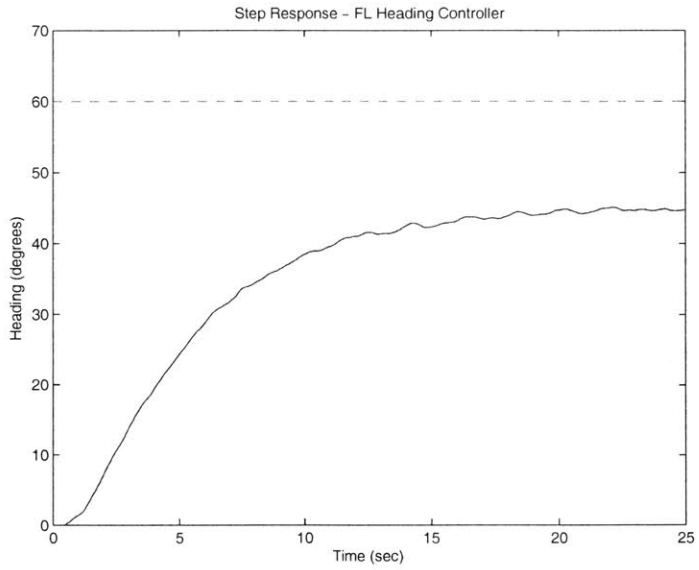

Figure 4.4.3: Step response of the nonlinear Feedback Linearization controller to a 60° heading change. There is significant steady-state error. Azimuth saturation is 60°. Data taken November 9, 2005.

the desired heading more quickly than the nonlinear controller. This is because the azimuth angle pulls away from its saturation value sooner with the FLI controller, so more of the heading error on the approach is integrated than with the PID controller. The PID controller was tested on the kayak, but the FLI controller was not. The data from the PID test is presented in Figure 4.4.5. The PID controller is used for path following, discussed in the next chapter. Its form is shown in Equation 4.4.1.

$$F_{ycmd} = -K_{pH}(\psi - \psi_d) - K_{dH}r - K_{iH}\int_0^t (\psi - \psi_d)\, dt \tag{4.4.1}$$

4.5 Summary

Linear controllers handle linear dynamics well, while nonlinear controllers are better equipped to handle nonlinear dynamics. However in this system the dynamics are most nonlinear in hard turns when the azimuth angle is likely to be saturated anyway; once the azimuth angle is saturated it does not matter what controller is used. However, once the vessel approaches

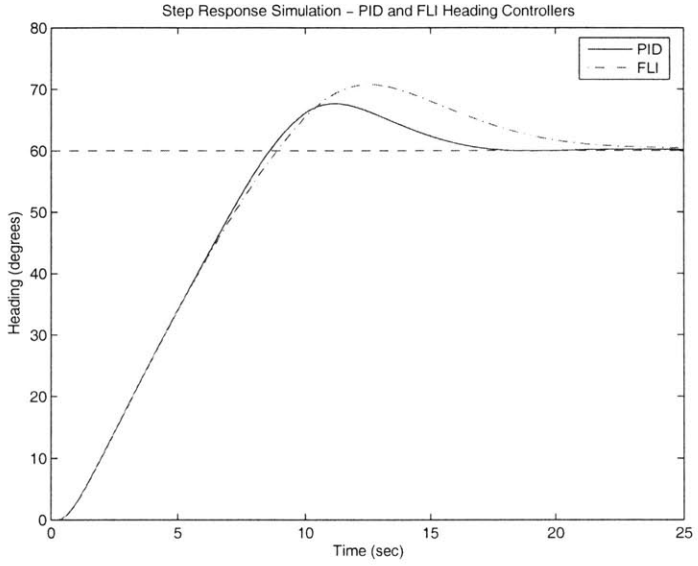

Figure 4.4.4: Step response simulation of the linear PID and nonlinear Feedback Linearization plus Integral heading controllers to a 60° heading change. Some overshoot is to be expected to balance the approach to the desired heading, even with the anti-windup scheme, but the linear controller recovers from the overshoot faster than the nonlinear controller. Azimuth saturation is 80°.

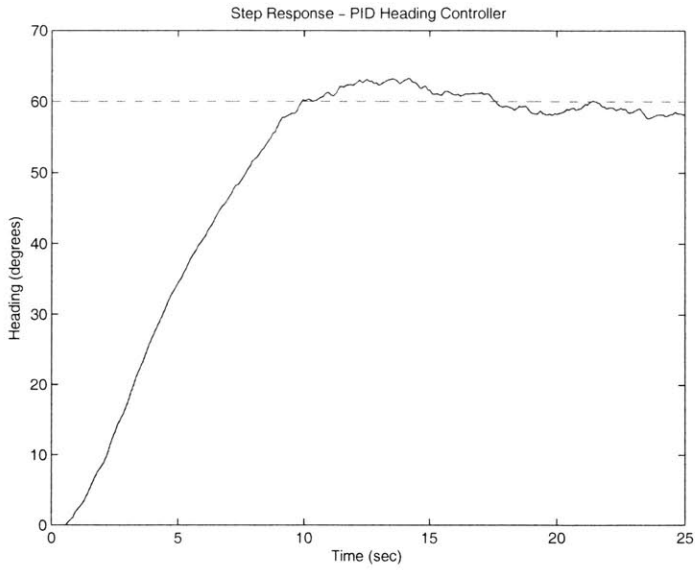

Figure 4.4.5: Step response of the linear PID controller to a 60° heading change. The integrator eliminates steady-state error, although it does add overshoot. Azimuth saturation is 80°. Data taken November 21, 2005.

the desired heading and the propulsor pulls away from the azimuth saturation limit, the nonlinear controller has an opportunity to improve the dynamics. In this system the nonlinear FL controller did prevent overshoot in the simulation, but the overall performance was very similar to the much simpler PD control. River tests showed that integral control was necessary. When an integral term was added to the two controllers they both gained overshoot and the linear PID controller recovered faster than the nonlinear FLI controller. Ultimately the PID controller was used in the path following algorithm for its simplicity.

Chapter 5

Waypoint Tracking

The greater sideforce achieved with an azimuthing propulsor allows for much more accurate path following than with a rudder. Figure 5.0.1 shows a vessel with an azimuthing propulsor entering a 90° course change. The high sideforce causes the vessel to yaw quickly and enter a skid. The path-following algorithm pulls the vessel out of the skid and accelerates on the new path leg. The trajectory of the center of gravity follows the path very well.

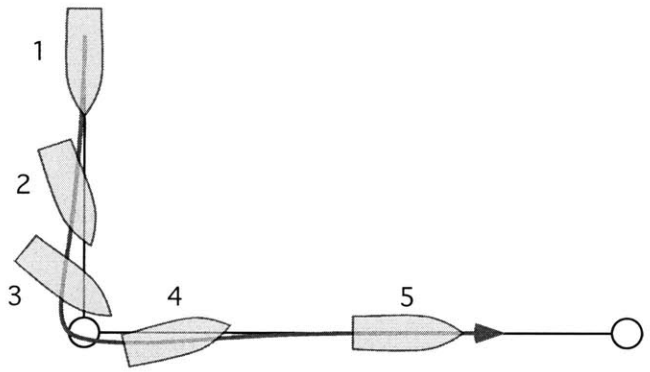

Figure 5.0.1: A vessel with an azimuthing propulsor approaches a 90° course change (1). The propulsor generates a large sideforce that causes the vessel to yaw (2) and skid towards the waypoint (3). The path-following controller pulls the vessel out of the turn (4) and the vessel accelerates onto the new path leg (5).

5.1 Path Following Definitions for Straight Paths

When deriving control laws for path following or studying their stability we must rotate the global coordinates to the local path coordinates. Definitions of the coordinates and distances

are indicated in Figure 5.1.1. For an arbitrary path segment between the last waypoint at coordinates $(LastX, LastY)$ and the current waypoint at coordinates $(CurrentX, CurrentY)$, the cross-track error y_b is computed as follows:

$$y_b = -\frac{(CurrentX - LastX)(LastY - Y_{boat}) - (LastX - X_{boat})(CurrentY - LastY)}{\sqrt{(CurrentX - LastX)^2 + (CurrentY - LastY)^2}} \tag{5.1.1}$$

The denominator of Equation 5.1.1 is defined as D_{wp}, the distance between the waypoints of the current path segment. The distances R_{boat} and x_b are computed through simple geometry:

$$R_{boat} = \sqrt{(CurrentX - X_{boat})^2 + (CurrentY - Y_{boat})^2} \tag{5.1.2}$$

$$x_b = D_{wp} - \sqrt{{R_{boat}}^2 - {y_b}^2} \tag{5.1.3}$$

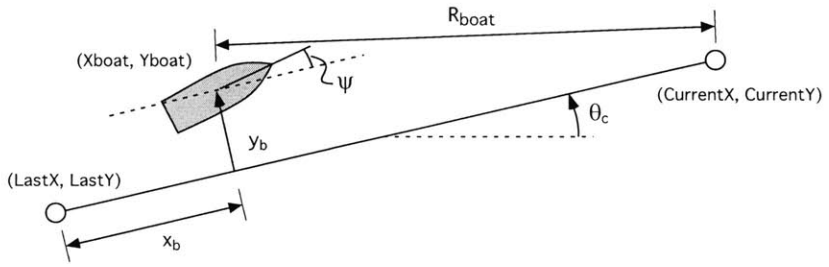

Figure 5.1.1: Definitions of the nomenclature used in straight-line path following.

The heading angle ψ is defined relative to the local path angle θ_c so that we can use the small-angle approximation when the vessel is near the path. In general the along-track position x_b and the cross-track error y_b evolve according to Equations 5.1.4 and 5.1.6. When the vessel dynamics are linear about a nominal forward speed u_o and the heading angle ψ is small, these equations linearize to 5.1.5 and 5.1.7.

$$\dot{x}_b = u\cos\psi - v\sin\psi \tag{5.1.4}$$

$$\cong u_o \tag{5.1.5}$$

$$\dot{y}_b = u\sin\psi + v\cos\psi \tag{5.1.6}$$

$$\cong u_o\psi + v \tag{5.1.7}$$

The dynamics of the path-following controller can be summarized in a state space feedback system. The input to the system is the commanded sideforce F_{ycmd}, which is passed through a second-order filter to model the azimuth dynamics. The output depends on the application, but it is either the local position (x_b, y_b) and the local heading and yaw rate of the vessel (for use with the Kalman filter) or it is just the cross-track error y_b. In either case the state space system is

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}F_{ycmd} \tag{5.1.8}$$

where

$$\mathbf{x} = \begin{bmatrix} u & v & r & \psi & \psi_{int} & x_b & y_b & y_{bint} & \dot{F}_y & F_y \end{bmatrix}^T$$

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & 0 & 0 & 0 & 0 & 0 & b_1 \\ a_{21} & a_{22} & a_{23} & 0 & 0 & 0 & 0 & 0 & 0 & b_2 \\ a_{31} & a_{32} & a_{33} & 0 & 0 & 0 & 0 & 0 & 0 & b_3 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & u_o & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -(\lambda_1+\lambda_2) & -\lambda_1\lambda_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \\ r \\ \psi \\ \psi_{int} \\ x_b \\ y_b \\ y_{bint} \\ \dot{F}_y \\ F_y \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1\lambda_2 & 0 \end{bmatrix}^T$$

Note that this state space system includes integrals of heading angle and cross-track error for use with the integral control in the heading controller and the path-following controller. λ_1 and λ_2 are the poles in the azimuth controller reference model of Equation 3.7.1. The values a_{ij} and b_i are the elements of the **A** and **B** matrices from Equation 4.1.12.

5.2 Kalman Filter

There are three measurements that are used by the heading controller and path following algorithm: heading ψ from the compass, yaw rate r from the DMU, and position (x_b, y_b) from the GPS sensor. The heading and position are rotated to the local path direction so

that nonzero ψ and y_b represent error. The noise properties of the sensors were discussed in Chapter 3. Low-pass filters would introduce lags into the system, and tight path following can only be achieved if lags are kept to a minimum. A Kalman filter is an optimal model-based alternative that not only filters the system outputs but also generates estimates of the hidden states. The Kalman filter runs a linear simulation of the system using estimated states and predicts the measurements that the sensors receive. The predicted measurements are compared with the actual measurements and any differences are propagated back to the estimated states through a Kalman filter gain matrix $\mathbf{H}$ (Equation 5.2.1).

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\,F_{ycmd} + \mathbf{H}\,(\mathbf{y} - \mathbf{C}\hat{\mathbf{x}}) \tag{5.2.1}$$

The outputs $\mathbf{y}$ and the output matrix $\mathbf{C}$ are defined below.

$$\mathbf{y} = \begin{bmatrix} r \\ \psi \\ x_b \\ y_b \end{bmatrix} \qquad \mathbf{C} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

The state space system with noise is modeled as in Equations 5.2.2 and 5.2.3.

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}F_{ycmd} + \mathbf{w} \tag{5.2.2}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{v} \tag{5.2.3}$$

It is assumed that the plant disturbance noise $\mathbf{w}$ and the sensor noise $\mathbf{v}$ can be modeled as white noise with diagonal covariance matrices $\mathbf{W}$ and $\mathbf{V}$ respectively. The elements of $\mathbf{V}$, a diagonal 4 x 4 matrix, are the noise variances of the sensors:

$$\mathbf{V} = \begin{bmatrix} V_{dmu} & 0 & 0 & 0 \\ 0 & V_{compass} & 0 & 0 \\ 0 & 0 & V_{gps} & 0 \\ 0 & 0 & 0 & V_{gps} \end{bmatrix}$$

Because most of the states in $\mathbf{x}$ are integrators, disturbance noise and wave forces can only enter the system through $\dot{u}$, $\dot{v}$, and $\dot{r}$. The 10 x 10 diagonal matrix $\mathbf{W}$ only has nonzero

values on the first three elements of the diagonal. This is the only physically viable form for **W**.

The Kalman filter is designed for the continuous steady-state problem, but it is implemented on the kayak as a discrete unsteady filter according to the formulation in [7]. The sample rate of the computer system is high enough that this substitution is valid.

5.2.1 Noise Models

The sensor noise variances $V_{compass}$, V_{dmu}, and V_{gps} were computed from stationary test data over a 60 second timespan. The compass noise and DMU noise were high frequency with no drift. However, the GPS noise had essentially no high frequency noise and, over a 60 second timespan, no drift. Periodic GPS jumps were not modeled. Furthermore the compass data has a rate of 11 Hz and the GPS data has a rate of 5 Hz, while the sample rate of the onboard computer code is 50 Hz. A zero order hold is used on the sensor values and this was not incorporated into the noise model either. The noise variances are listed in Table 5.2.1.

Table 5.2.1: Sensor noise variance

Sensor	Model	Variance	Units
Compass	PNI MicroMag3	1.37×10^{-5}	rad^2
Yaw Rate	Crossbow VGX-300	1.53×10^{-5}	$(\text{rad/sec})^2$
GPS Position	Garmin GPS18-5Hz	2.22×10^{-4}	m^2

The Kalman filter is based on a linear model of a nonlinear system. In order to be effective it must trust the sensors when the system is in the nonlinear regime but still filter out as much noise as possible. Examples of nonlinearity are yaw rate saturation (there is a limit to how quickly the vessel can turn) and the position rotation when the vessel is approaching the path with a large relative heading ψ (causing the linearizations of Equations 5.1.5 and 5.1.7 to become unrealistic). The plant disturbance covariance was adjusted manually until the Kalman filter achieved a good balance between noise filtering and response in the presence of nonlinear vessel dynamics. The first three nonzero elements of **W** were chosen to be 0.005. The units are m^2/sec^4 for $\dot{u}$ and $\dot{v}$ and $1/\text{sec}^4$ for $\dot{r}$.

5.2.2 Performance

Despite the zero-order hold on the GPS signal, the filtered position signal tracks and predicts the GPS signal very well, as shown in Figure 5.2.1.

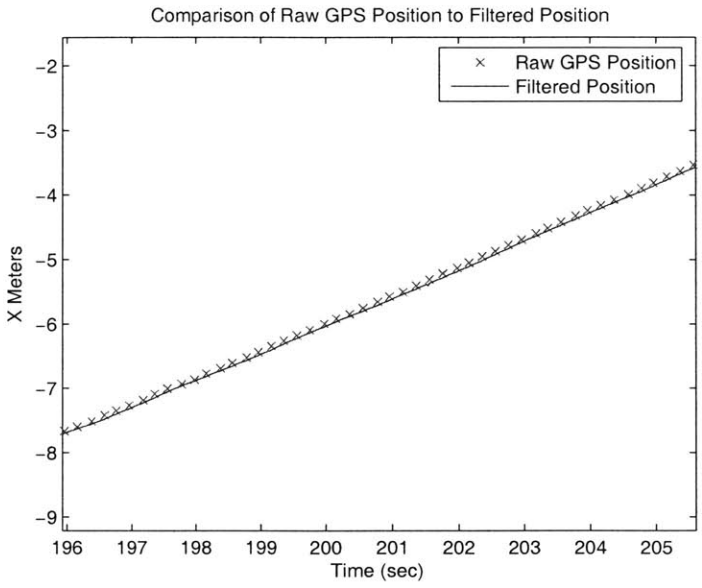

Figure 5.2.1: Raw and Kalman-filtered X-position of the vessel. Although the GPS data is discrete the filter smooths the signal and extrapolates future values very well.

The yaw rate is only lightly filtered so that in hard turns, when the system is far from the linear operating point, the filter trusts the measurement values more than the linear prediction. As a result the noise power is only reduced by a factor of two or so. However the heading signal is filtered more and reduces the noise power by a factor of around four.

5.3 Line-Of-Sight Tracking on Straight Paths

5.3.1 Definition

The most basic method of path following on straight paths for marine vessels is the Line-Of-Sight method. Shown in Figure 5.3.1, this method sets the desired heading of the vessel to point towards the current waypoint. To avoid stability problems near the waypoint, a distance d_{switch} is defined for each waypoint and if the vessel approaches within this radius of the waypoint then the controller switches to the next waypoint.

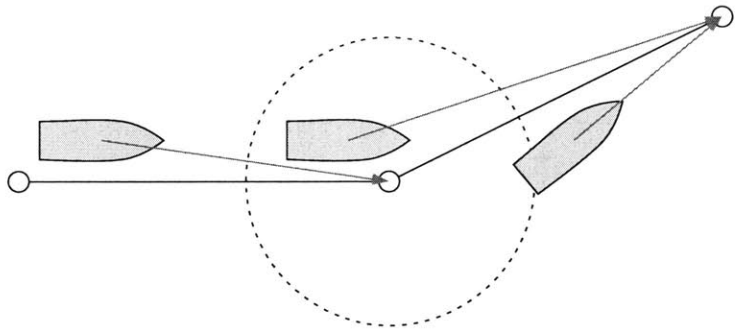

Figure 5.3.1: Line of Sight waypoint tracking; the desired heading always points towards the current waypoint, and the controller switches to the next waypoint when it is within a distance d_{switch} of the current one.

As we will see below stability is closely linked to the distance between the vessel and the point on the path towards which its heading controller is trying to point the vessel. In short, if the vessel is 1 km away from the waypoint and it is 100 meters off the path then the heading correction is $\psi_d = \tan^{-1} \frac{100}{1000} = 5.7°$. If the vessel is 100 meters off the path but only 200 meters away then then heading correction is 26.6°. The heading controller will have to work much harder (turn the rudder or propulsor to a larger angle) to get the vessel back on track in the second case. Such behavior may be desired if precise path following is not necessary except near the waypoints, but the system would much less resilient to disturbances such as current or wind at far distances from the waypoint. In general, for consistency of dynamics, stability, and disturbance rejection, it is more desirable to apply the same heading correction for the same cross-track error no matter where the vessel is on the path. The way to do this is to point not towards the waypoint but rather towards a point on the path between the current waypoint and the last waypoint; this point is an linear interpolation of the coordinates of the two waypoints, so here it is referred to as "interpolated waypoint tracking."

There are several ways to choose the point on the path used by the interpolated tracking method to set the desired heading for the heading controller. Three are presented here. The first is to simply project the vessel's position to the closest point on the path, then move a distance d towards the current waypoint. The distance d is known as the "look-ahead distance." This scheme is shown in Figure 5.3.2. It is the most relaxed of the three methods, meaning that for very large deviations from the path the heading correction is the smallest.

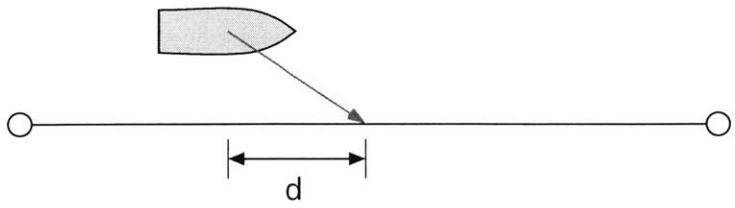

Figure 5.3.2: First scheme for interpolated waypoint tracking; choose the point on the path that is a distance d ahead of the path-projected location of the vessel.

A second method draws a circle of radius d around the vessel; the interpolated point used by the heading controller is the intersection of this circle with the path (Figure 5.3.3. One obvious wrinkle appears when the cross-track error of the vessel is larger than d. In this case the interpolated point could be the path-projected position of the vessel (that is, the point on the path that comes closest to the circle around the vessel) or another scheme could be used. This is the most stringent of the three methods, meaning that it causes the largest heading corrections of the three methods for large deviations from the path.

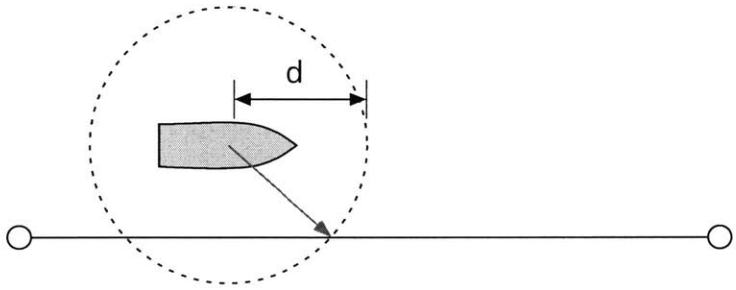

Figure 5.3.3: Second scheme for interpolated waypoint tracking; choose the point on the path that intersects a circle of radius d drawn around the vessel.

The third method determines the distance of the vessel from the waypoint R_{boat}, subtracts d from that distance, and sets the point on the path a distance $R_{boat} - d$ from the waypoint (Figure 5.3.4). This method is the easiest to compute of the three presented, and for large cross-track errors it falls between the first and second methods in terms of heading correction magnitude. However there is a small variation in heading correction at large cross-track errors as the vessel approaches the waypoint.

Table 5.3.1 shows the heading corrections due to the three methods at various locations relative to the path. Note that all three interpolation methods cause a larger heading

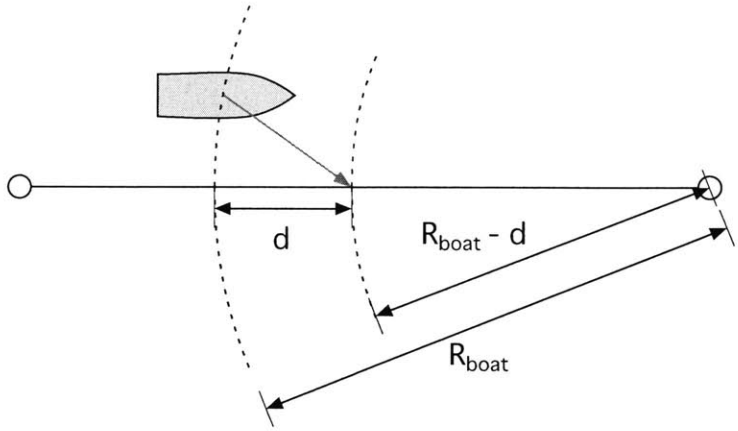

Figure 5.3.4: Third scheme for interpolated waypoint tracking; choose the point on the path a distance $R_{boat} - d$ from the waypoint, where R_{boat} is the distance from the vessel to the waypoint.

correction than pure LOS waypoint tracking when the vessel is far from the waypoint. Also note that the correction is mostly independent of distance to the waypoint and at small cross-track errors all three methods are essentially the same.

Table 5.3.1: Heading corrections for the three interpolated waypoint tracking methods at two distances from the waypoint and two cross-track errors; $d = 200$ m. Also pure Line of Sight (aim for waypoint).

Interpolated Waypoint Tracking Method	10 m error	100 m error	200 m error
Method 1 - 1000 m from waypoint	2.86°	26.6°	45°
Method 1 - 500 m from waypoint	2.86°	26.6°	45°
Method 2 - 1000 m from waypoint	2.87°	30°	90°
Method 2 - 500 m from waypoint	2.87°	30°	90°
Method 3 - 1000 m from waypoint	2.86°	27.1°	48.0°
Method 3 - 500 m from waypoint	2.86°	27.7°	51.1°
Pure LOS - 1000 m from waypoint	0.57°	5.71°	11.3°
Pure LOS - 500 m from waypoint	1.15°	11.3°	21.8°

5.3.2 Path-Following PID Controller

For small errors all three interpolated tracking schemes put the interpolated point a distance d ahead on the path. The heading correction ψ_d for a cross-track error y_b then becomes

$$\psi_{dp} = -\tan^{-1}\frac{y_b}{d} \tag{5.3.1}$$

Since we are already assuming small errors, Equation 5.3.1 linearizes to

$$\psi_{dp} = -\frac{1}{d}\, y_b \tag{5.3.2}$$

The value $1/d$ is the proportional gain of the path following controller. (This is quasi-P control because the gain actually change slightly with increasing cross-track error.) Small values of d result in a large gain (large heading corrections) and large values of d result in a small gain (small heading corrections). This is intuitive because d, as the "look-ahead distance," represents the sensitivity of the path follower to error. If d is too small then the path follower will be too sensitive and generate unreasonably large control action. The desired heading due to proportional control is shown below.

$$\psi_{dp} = -K_p\, y_b \tag{5.3.3}$$

Integral control is necessary to compensate for environmental forces such as wind, waves, or current that act normal to the path. These forces are generally steady in magnitude and earth-fixed direction over the time scales of path following. This means that as long as the vessel is traveling on a straight path for a reasonable amount of time then the integrator of the path-following PID controller can be expected to compensate for the effects of the environmental forces. The desired heading due to integral control is shown below.

$$\psi_{di} = -K_i\, y_{bint} \tag{5.3.4}$$

Once integral control is introduced, derivative control must be added as well to put some damping in the system. Although there vessel itself has drag and linear damping characteristics in sway and yaw to damp the vessel *motions*, there is very little to damp the vessel *trajectory* (Figure 5.3.5). The derivative term shown below adds an artificial damping to the trajectory.

$$\psi_{dd} = -K_d\,(v + u_o\psi) \tag{5.3.5}$$

A simulation of a path-following control system is presented in Figure 5.3.6. Under a constant environmental force in the negative y direction the interpolated path following algorithm with no integral or derivative terms is well behaved but it cannot reach the path

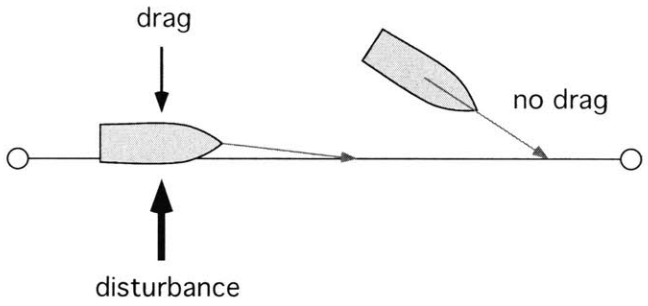

Figure 5.3.5: Drag and damping resist disturbances to the vessel's motion (left), but there is no physical damping of the vessel's trajectory (right). This is the motivation for derivative path following control.

in the steady state due to the constant force. Adding an integral term removes steady state error but makes the system more oscillatory. Adding a derivative term stabilizes the system once more.

5.3.3 Heading PID / Path-Following PID Stability

In most applications a heading controller is designed independently to provide the appropriate heading response. Once the heading controller has been designed the path-following PID gains can be chosen to stabilize the full state space system in Equation 5.1.8. The PID heading controller for the kayak was designed in the last chapter. The heading control law is shown in Equation 4.4.1 and reprinted below.

$$F_{ycmd} = -K_{pH}(\psi - \psi_d) - K_{dH}r - K_{iH}\int_0^t (\psi - \psi_d)\, dt$$

Combining Equations 5.3.3, 5.3.4, and 5.3.5 we get the total path following PID control law of Equation 5.3.6.

$$\psi_d = -K_p y_b - K_d(v + u_o\psi) - K_i y_{bint} \tag{5.3.6}$$

Plugging Equation 4.4.1 into Equation 5.3.6 and rearranging terms results in Equation 5.3.7.

$$F_{ycmd} \;=\; -\mathbf{K}\mathbf{x} \tag{5.3.7}$$

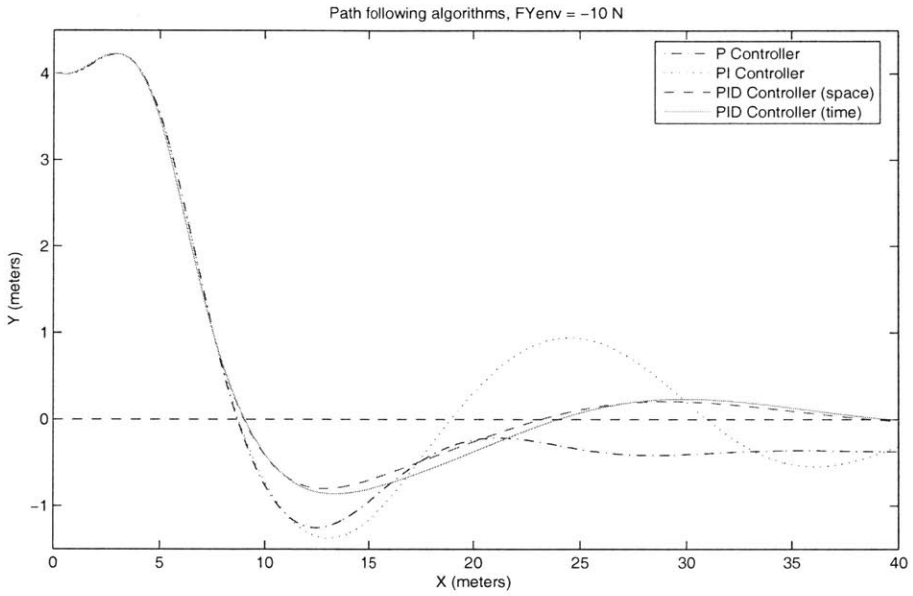

Figure 5.3.6: Interpolated waypoint tracking (blue) has a good response but is susceptible to steady-state error when a constant environmental force is present. A PI controller (green) eliminates steady state error but is more oscillatory. Full PID control (red and light blue) provides the best response.

where

$$\mathbf{K} = \begin{bmatrix} 0 & K_{pH}K_d & K_{dH} & K_{pH}(1+K_d u_o) & K_{iH} & 0 & K_{pH}K_p & K_{pH}K_i & 0 & 0 \end{bmatrix}$$

and

$$\mathbf{x} = \begin{bmatrix} u & v & r & \psi & \psi_{int} & x_b & y_b & y_{bint} & \dot{F}_y & F_y \end{bmatrix}^T$$

Recall that the system input F_{ycmd} is filtered within the full state space system's $\mathbf{A}$ matrix of Equation 5.1.8.

Using the feedback in Equation 5.3.7 with the full state space system of Equation 5.1.8, the closed-loop system becomes

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\,F_{ycmd} = \mathbf{A}\mathbf{x} - \mathbf{B}\mathbf{K}\mathbf{x} = (\mathbf{A} - \mathbf{B}\mathbf{K})\,\mathbf{x} \tag{5.3.8}$$

In the actual system the Kalman filter estimates of the states $\hat{\mathbf{x}}$ are used instead of the raw r, ψ, x_b, and y_b data. The combined control and filter state space system is derived in Equation 5.3.9.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}F_{ycmd} \\ F_{ycmd} &= -\mathbf{K}\hat{\mathbf{x}} \\ \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} - \mathbf{B}\mathbf{K}\hat{\mathbf{x}} \\ \dot{\hat{\mathbf{x}}} &= \mathbf{A}\hat{\mathbf{x}} - \mathbf{B}\mathbf{K}\hat{\mathbf{x}} + \mathbf{H}(\mathbf{C}\mathbf{x} - \mathbf{C}\hat{\mathbf{x}}) \\ \begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\hat{\mathbf{x}}} \end{bmatrix} &= \begin{bmatrix} \mathbf{A} & -\mathbf{B}\mathbf{K} \\ \mathbf{H}\mathbf{C} & \mathbf{A} - \mathbf{B}\mathbf{K} - \mathbf{H}\mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \hat{\mathbf{x}} \end{bmatrix} \end{aligned} \tag{5.3.9}$$

System stability is guaranteed around the nominal operating conditions if all of the eigenvalues of Equation 5.3.9 have negative real parts. However, a highly oscillatory response of any state is undesirable as well. A more stringent restriction than basic stability is to ensure that none of the poles of the system are more than a certain angle, such as 60°, away from the negative real axis in the complex plane. For a fixed heading controller we can adjust the path-following controller's PID gains to find where the system meets the 60° stability requirement. Intuitively we expect a high K_i to make the system less stable (although for environmental force rejection we want it to be as large as possible), a high K_p

to make the system less stable (although we want it to be as large as possible as well), and a high K_d gain make the system less oscillatory up to the point where azimuth lag become a problem. The system stability was analyzed for K_i ranging from 0 to 0.1, K_d ranging from 0 to 1, and d ranging from 3 meters to 8 meters. The manifold of 60° stability is plotted in Figure 5.3.7. Within this manifold all of the system poles are less than 60° off the negative real axis. Note that there is a region of points outside this manifold that represent stable systems in which some poles are between 60° and 90°.

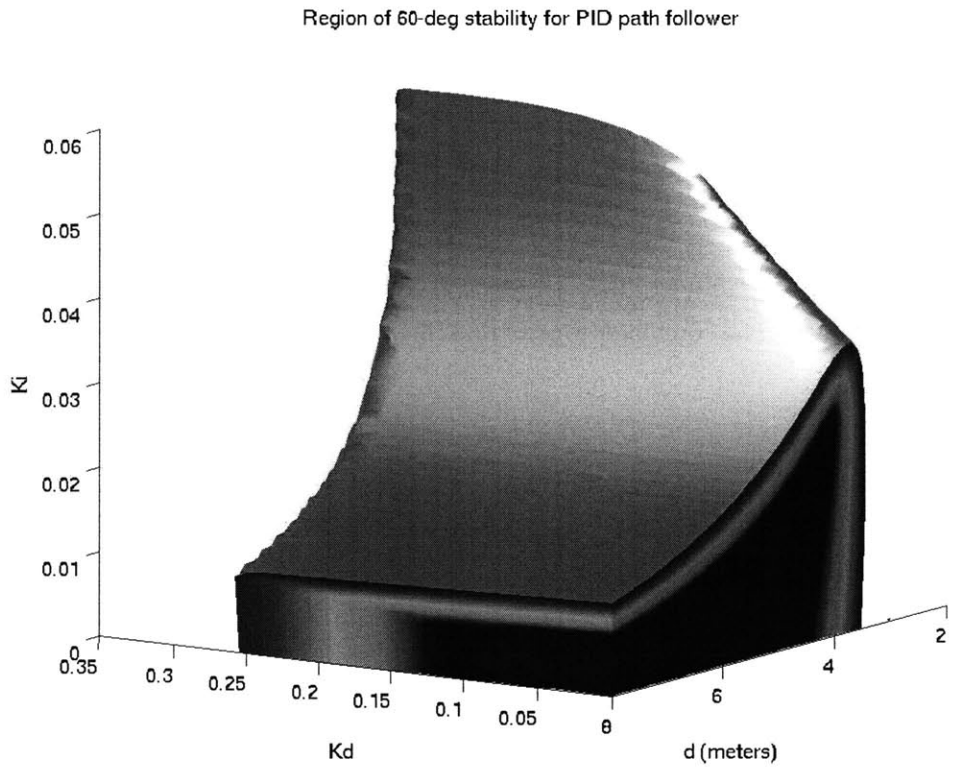

Figure 5.3.7: Region of $[d, K_d, K_i]$ space representing closed-loop systems whose poles are less than 60° off the negative real axis.

Figure 5.3.8 shows the slice of the PID parameter space at which $d = 5$ meters. Based on this plot gains of $K_d = 0.25$ and $K_i = 0.03$ were chosen. However, in simulations and kayak testing the K_i gain was kept at 0 unless it was needed to counteract steady forces or errors. Note that d is only 135% of the ship length, whereas the look-ahead distance most often cited in the literature is 2 to 3 ship lengths [8, 9, 10, 16]. The higher available

sideforce associated with azimuthing podded propulsion systems enables ships to have a much stronger response to path errors. This is not always desirable; for example high error rejection in an ocean crossing is not necessary and contributes to high control action. But for surveying and mapping applications it is very important to stay perfectly on the path despite unsteady disturbance forces so high control action is tolerated.

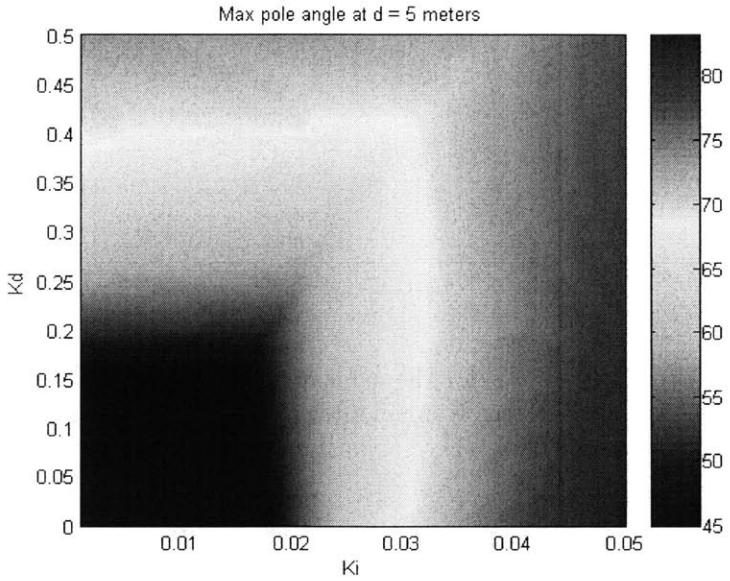

Figure 5.3.8: Slice of $[d, K_d, K_i]$ space at $d = 5$ meters. The color represents the maximum angle of any closed-loop pole.

Figure 5.3.9 shows the pole locations for the gains $d = 5$ meters, $K_d = 0.25$, and $K_i = 0.03$.

5.3.4 Non-Minimum Phase System

The path-following system is non-minimum phase, meaning that there is an extra delay between the input (sideforce) and output (cross-track error). This delay comes from the fact that the propulsor is aft of the center of gravity of the vessel. If the vessel is to turn right then the propulsor (or rudder, in the more general case) must generate a thrust to the left; this creates a moment that rotates the ship to the right but at the same time it pushes the vessel to the left. In effect the vessel center of gravity moves left before it moves right (Figure 5.3.10).

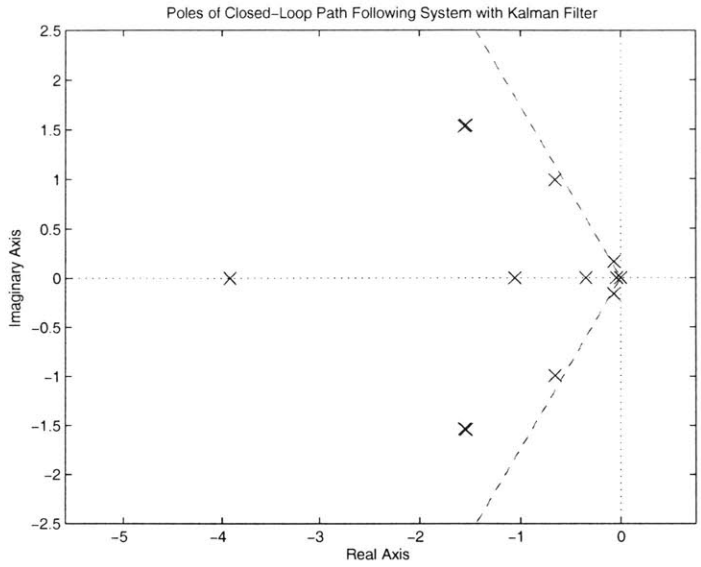

Figure 5.3.9: Closed-loop pole locations for the gains $d = 5$ meters, $K_d = 0.25$, and $K_i = 0.03$ for the full feedback system with the Kalman filter in the loop. There is one additional pole at -10.66 and another at -18.00.

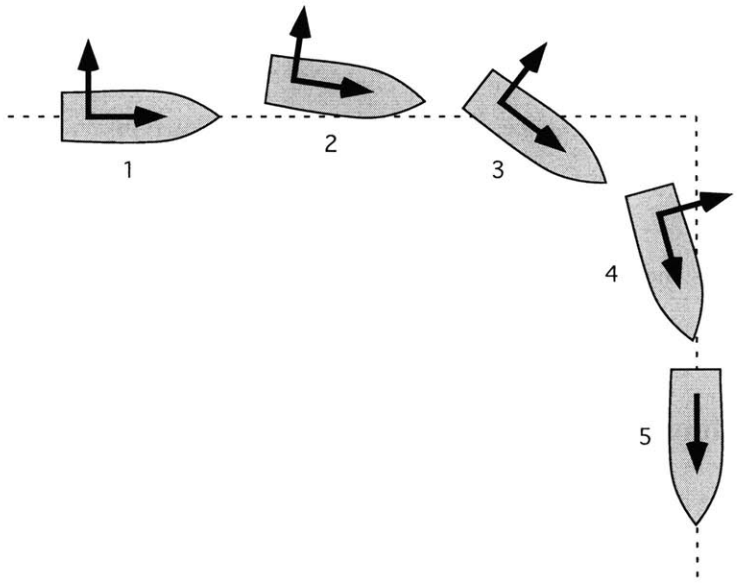

Figure 5.3.10: The vessel dynamics are non-minimum phase because the side force is generated aft of the center of gravity. When the vessel starts a turn to the right (1) it first moves to the left (2). The vessel returns to the original path after a certain distance (3) and continues its turn (4) before straightening out (5). The distance between (1) and (3) represents the delay due to the non-minimum phase behavior.

The amount of "reverse sideslip" (motion opposite the ultimate turn direction) can be computed from linear dynamics but for high rates of turn the linear model is no longer valid.

5.3.5 Space-Based Integrator

The point of integral control is to eliminate steady-state errors. Integral control is most often used to overcome friction (as it is used in the kayak's azimuth controller) or steady disturbances (as it is used in the kayak's heading controller). There is no friction in the path following problem but there are disturbances in the form of steady environmental forces such as wind, waves, and currents. With proportional path-following control alone, these forces will be balanced by the heading controller; the vessel will drive alongside the path with a steady angle relative to the path direction. This steady angle is known as the crabbing angle. The angle is necessary to generate a force to balance the steady environmental disturbance. However in proportional path following control this angle can only be created if there is nonzero cross-track error y_b as indicated in Figure 5.3.11. In PI control the angle is created by the integral term while the proportional term keeps the vessel on track.

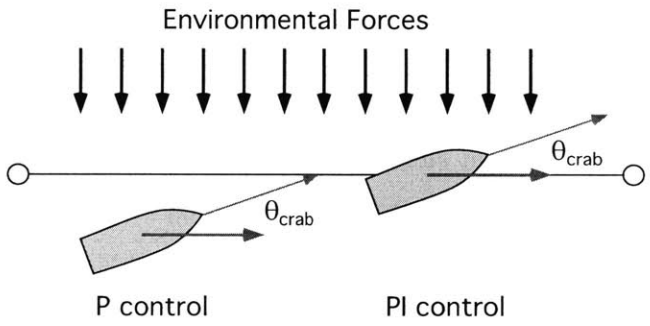

Figure 5.3.11: Proportional control alone cannot compensate for environmental forces without sliding off the path (left). If integral control is added (right) then the integral term compensates for environmental forces while the proportional term keeps the vessel on the path. θ_{crab} is the known as the crabbing angle.

Integral control responds to accumulated error. However, for path following there are two ways to accumulate error: in time and in space. The standard time integral adds the instantaneous error at every time step as in Equation 5.3.10.

$$y_{bint} = \int_0^t y_b \, dt \tag{5.3.10}$$

However, for situations like the one shown in Figure 5.0.1 the vessel is approaching the new path leg very quickly without moving along the new path at all. The cross-track error is reducing as quickly as it can and the vessel is actually following the correct trajectory but time-based integration would punish the cross-track error anyway. The accumulated error on one side of the path has to be balanced by error on the other path so that the integral can diminish back to zero. Time-based integration in Figure 5.0.1 would overshoot past the new path leg and possibly become unstable.

If error is accumulated in space, as in Equation 5.3.11, then the integral term only grows if the vessel is actually moving along the new path leg. Here if dx_b is negative then the integration is temporarily switched off.

$$y_{bint} = \int_0^{x_b} y_b \, dx_b \tag{5.3.11}$$

Equation 5.3.11 is actually integrating the area between the vessel trajectory and the path. This is a more intuitive way to represent the accumulated path error. Unfortunately this form of integral control is nonlinear because it depends on the product of the two system states y_b and x_b. However we can get a sense of what will happen with some simple calculations. First of all the two methods are essentially the same when the vessel is in its nominal operating condition, straight along the path with a velocity u_o. In fact, with the appropriate choice of integral gains the two methods are identical in this condition. If the time-based integral gain is K_{it} and the space-based integral gain is K_{is} then between two points the integrated error is

$$\begin{aligned} \text{(time)} \quad & \Delta I &= K_{it}\, y_b\, \Delta t \\ \text{(space)} \quad & \Delta I &= K_{is}\, y_b\, \Delta x_b \end{aligned}$$

which collapses to

$$K_{it} = \frac{\Delta x_b}{\Delta t} K_{is} = u_o \, K_{is} \tag{5.3.12}$$

Figure 5.3.12 shows the maximum pole angle of the closed-loop system as the vessel speed drops for equivalent integrator gains $K_{it} = 0.03$ and $K_{is} = 0.03/u_0$. The system becomes less stable when the time-based integrator is used but it stays about the same

when the space-based integrator is used. This is because the effective integration gain increases as speed drops in the time-based case. Another way to study the problem is to look at the region of stability for the system as speed drops. Figure 5.3.13 shows the region of stability of the closed-loop system over a range of u and K_{it} (and equivalent K_{is}). As the vessel slows down the system can tolerate larger space-based integral gains but the region of stability for time-based integration shrinks.

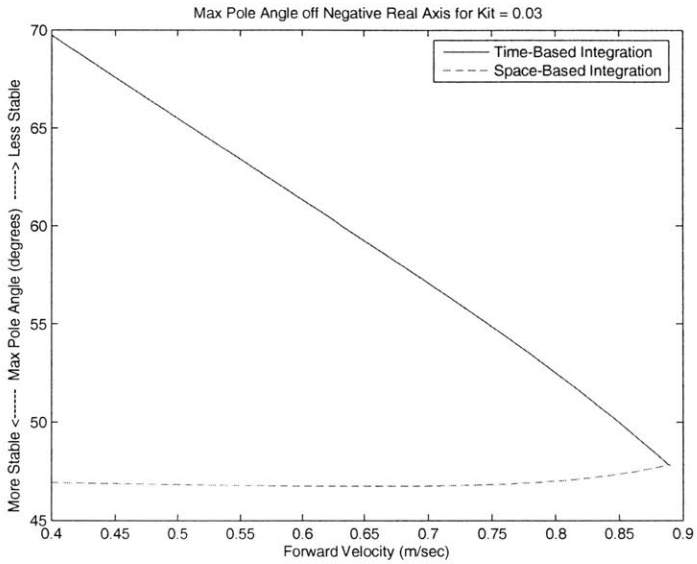

Figure 5.3.12: Maximum pole angle of the closed-loop system for $K_{it} = 0.03$ and the equivalent $K_{is} = 0.03/u_o$ as the vessel slows down.

The differences are clearly illustrated in Figure 5.3.14. If the vessel is initially far off the path with a large relative heading angle then the time-based integration scheme goes unstable but the space-based integration scheme can recover. Both controllers initially steer in the $-x$ direction because of the derivative control term. The simulation in Figure 5.3.14 was run with no exclusion rule on the integration. An exclusion rule turns the integration off under certain circumstances. In Figure 5.3.6 the integration is turned off if the vessel is farther than 2 meters off the path or the vessel is within a distance $d/2$ from the last waypoint (to allow for small overshoot at the waypoint). With these restrictions the space-based integration scheme performs only slightly better than the time-based integration scheme.

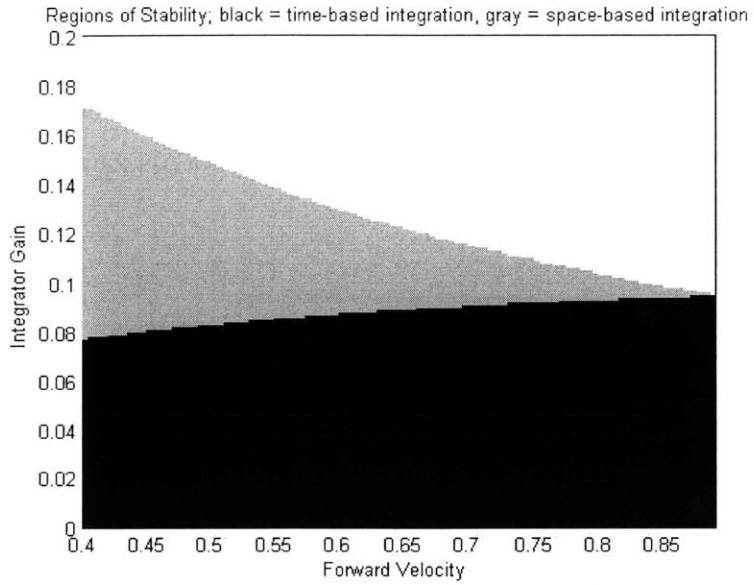

Figure 5.3.13: Region of stability for time-based integration (black) and space-based integration (gray) as vessel speed drops.

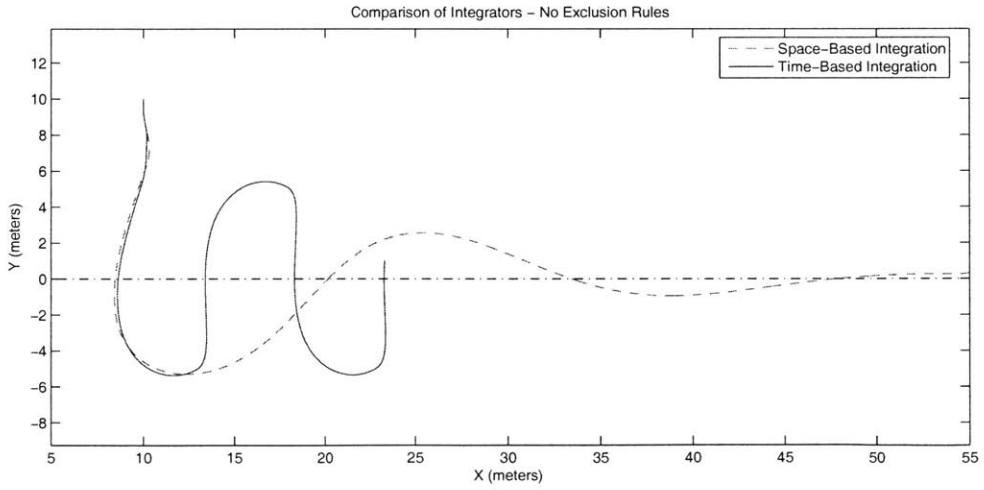

Figure 5.3.14: With no exclusion rule the time-based integrator goes unstable from a large initial offset and heading angle, but the space-based integrator can recover because the integral term does not build up unless there is forward progress along the path.

Figure 5.3.14 represents a rather pathological case, but without integration exclusions the space-based integrator outperforms the time-based integrator in real applications. Figure 5.3.15 shows the path following performance of the two schemes. The path-following controller using space-based integration follows the path very well while the controller using time-based integration goes unstable.

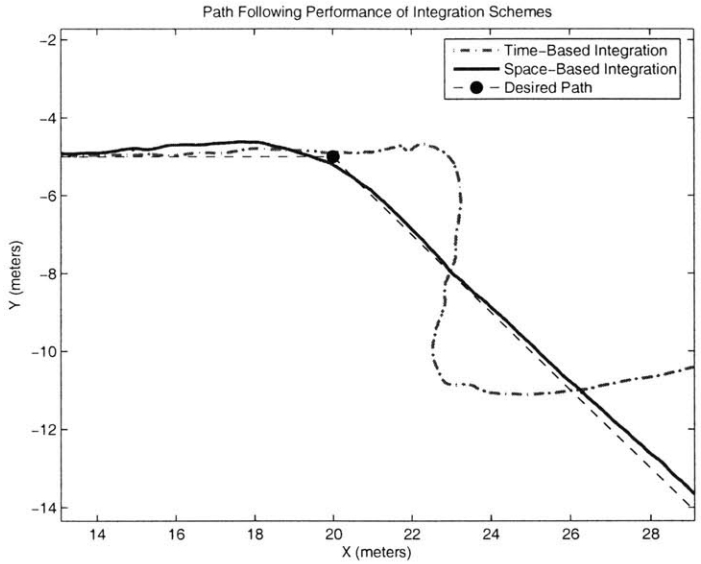

Figure 5.3.15: Path-following performance using time-based integration and space-based integration. The space-based scheme performs very well while the time-based scheme goes unstable. Data collected November 18, 2005.

5.4 Course Change Feedforward

The simplest line-of-sight waypoint tracking methods do not have any mechanisms for dealing with course change. The vessel simply switches waypoints and follows the new desired heading to converge to the new path. More sophisticated algorithms start the turn a certain distance early, typically one ship length, so that the yaw rate reaches its steady state value by the time the vessel reaches the d_{switch} distance [8]. However the intermediate trajectory is still smooth and the d_{switch} distance is chosen so that the vessel can follow a circular trajectory from one path leg to the next.

To follow non-smooth paths as well as possible we do not want to be bound by the

circular-path restriction. Ideally we can take advantage of the fast yaw-rate dynamics to enter into a skid that the vessel pulls out of at just the right moment (Figure 5.0.1). To do this the propulsor must react rather violently at the beginning of the turn. This is accomplished using a feedforward algorithm. We assume for this problem that there is low control cost and we can move the propulsor quickly if need be.

5.4.1 Line-Of-Sight

The simplest way to follow a path from one leg to the next is to simply switch to the new path leg when the vessel gets within a distance d_{switch} of the waypoint. The desired heading jumps and the heading controller responds to drive the heading of the vessel to the new value. If the desired heading jump is large then a large amount of propulsor or rudder action may be required. Figure 5.4.1 shows the desired heading jump if the switching distance $d_{switch} = d$. The plot also shows the jump if an equivalent linear path following controller with proportional gain $K_p = 1/d$.

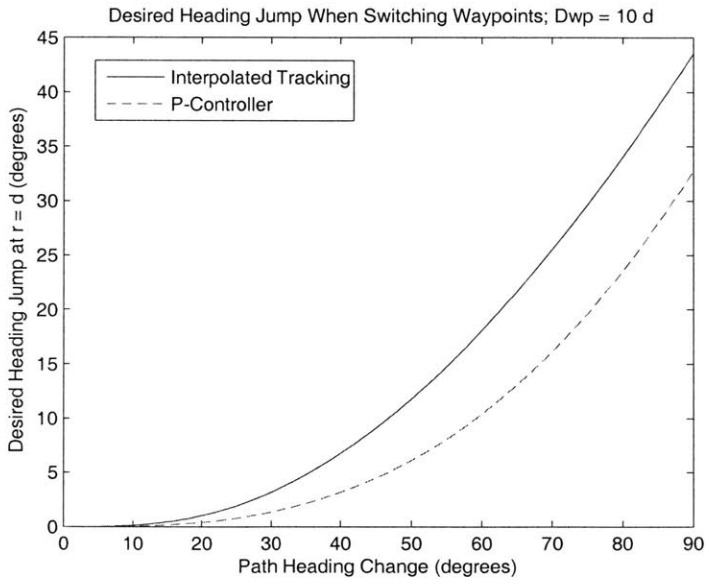

Figure 5.4.1: Desired heading discontinuity magnitude as a function of path angle change. The distance between waypoints is $D_{wp} = 10d$.

If d_{switch} is large compared to the minimum turning radius of the vessel then the vessel trajectory will be smooth and "cut the corner" of the course change. d_{switch} must be large

for vessels with slow turning dynamics or limits to the sideforce that can be generated by the propulsor or rudder. However if d_{switch} is small then the vessel must generate a large sideforce to be able to get to the new path leg without overshoot. Large sideforces are more easily obtained with azimuthing propulsors than with rudders because the entire propeller thrust can be directed sideways. This causes a high rate of turn but also significant sideslip.

5.4.2 Feedforward

The non-minimum phase behavior of the system causes delays that make high-performance path following difficult, especially near waypoints. Vessels with azimuthing propulsors can achieve higher yaw rates than vessels with rudders but reverse sideslip is increased as well. The reverse sideslip can be anticipated and counteracted using feedforward terms in the path following controller.

A human operator of a small vessel with an azimuthing propulsor such as an outboard engine can easily anticipate the sideslip associated with hard turns and perform 90° docking maneuvers with only basic experience with the vessel. The task here is to anticipate the sideslip and correct for it in an automatic control system.

Assertive control action is required to produce a trajectory that closely follows a sharp corner at a waypoint. This control action can be forced by permitting a discontinuity in the desired heading. Yet the natural discontinuity presented in Figure 5.4.1 may not be large enough in magnitude or early enough in time to produce a trajectory that accurately follows the path. The course change feedforward algorithm presented here produces a desired heading change that is larger and occurs earlier in time than the line-of-sight method alone.

5.4.3 Development of the Feedforward Algorithm

To take advantage of the high yaw rate of the vessel an ideal control input near a waypoint saturates at one extreme to initiate the turn and stays there until the turn is almost complete; then it recovers in the opposite direction to cancel the yaw rate (Figure 5.4.2).

There are two difficulties in implementing this ideal input. First, it is not trivial to get the timing correct. A human operator can figure this out with visual cues and experience but a simple path-following controller has the luxury of neither. Second, this input is time-based and is not robust to disturbances in position or heading angle. This controller must be implemented using spatial information so that precise path following is guaranteed.

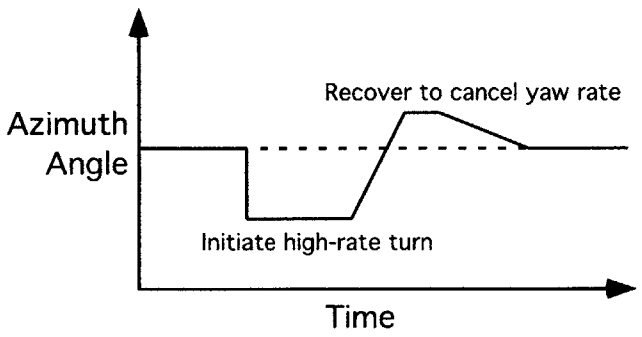

Figure 5.4.2: Ideal azimuth angle for a sharp course change at a waypoint. The angle saturates at one extreme to initiate a high rate turn and sideslip. At the appropriate moment the propulsor moves to the other side to stop the turn and recover onto the new path.

For straight-line path control we have been using a path-following controller outside of a heading controller; the path controller generates a desired heading, the heading controller generates a desired sideforce, and the azimuth controller moves the propulsor to the appropriate orientation to achieve both. This is a useful breakdown of the control problem because the controllers can be designed independently as long as stability of the entire system is verified at the end of the design process.

The design of the feedforward algorithm takes its cues from the non-minimum phase behavior of the system. One way to deal with the induced time delay is to steer as if the vessel were a certain distance farther along the path than it actually is. If the distance between points 1 and 3 of Figure 5.3.10 is 2 meters, for example, then the delay will be approximately cancelled if the turn is initiated 2 meters early.

This feedforward distance is labeled d_{turn} and it depends on the vessel dynamics, the heading controller, and the angle of the course change. This distance will be small or zero for small course change angles and larger for larger course change angles. Without this feedforward distance the turn would initiate when the vessel approaches within a distance d_{switch} of the waypoint; using the feedforward algorithm the turn initiates at a distance $d_{switch} + d_{turn}$.

Once the turn is initiated an imaginary vessel is placed a distance d_{turn} ahead of the actual ship; this models the non-minimum phase behavior. The desired heading of the actual ship is set to be the desired heading of the imaginary ship. The difference between

this algorithm and simply increasing d_{switch} by d_{turn} is that the initial desired heading jump is larger using the feedforward algorithm. Larger desired heading jumps ensure that the vessel will enter a high-rate turn and exploit the advantages of the podded propulsion system.

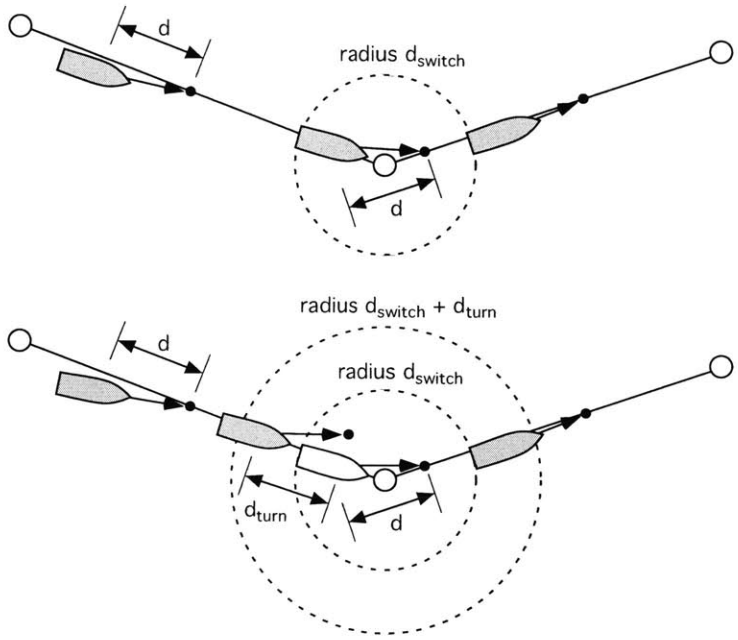

Figure 5.4.3: The upper diagram shows a standard line-of-sight tracking scheme: the vessel switches to the new course leg when it approaches within d_{switch} meters of the waypoint. The lower diagram shows the feedforward algorithm: the vessel uses the desired heading of an imaginary vessel d_{turn} meters ahead (white) when it approaches within $d_{switch} + d_{turn}$ meters of the waypoint and switches to the new course leg when the actual vessel approaches within d_{switch} meters of the waypoint.

One decision that must be made is what "ahead" means. For the purposes of high-precision path following the ideal vessel track follows the path exactly, right into the waypoint. One choice for the imaginary vessel location is to put it on the path leg no matter where the actual ship is cross-wise relative to the path. However this is impractical because it does nothing to correct for path errors of the actual ship. Another choice is to put the imaginary vessel on the line connecting the actual vessel and the waypoint. However this is also impractical because in the middle of a high-rate turn at large course change angles the vessel may not actually be moving towards the waypoint. A third choice, the one used in this algorithm, is to keep the imaginary vessel at the same cross-track error as the actual

vessel; the line connecting the actual vessel and the imaginary one is parallel to the course leg (Figure 5.4.4).

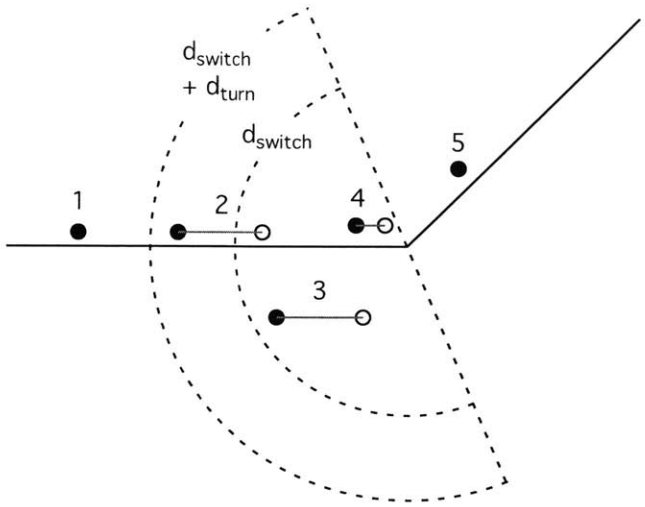

Figure 5.4.4: The bisection feedforward algorithm. The normal path-following controller is used when the vessel is outside of the $d_{switch} + d_{turn}$ radius (1). When the vessel enters this radius (black) it uses the desired heading of an imaginary vessel (white) d_{turn} meters ahead on a parallel path (2 and 3). If the actual vessel gets within d_{turn} meters of the bisection of the course legs then the d_{turn} distance is reduced to avoid discontinuities (4). Normal path following resumes when the vessel is past the bisection line (5).

The initial heading discontinuity when the vessel enters the $d_{switch} + d_{turn}$ radius is important to force the thruster hard to one side, but subsequent discontinuities are undesireable. A key issue is when to stop using the imaginary vessel's heading and switch back to the actual vessel. This transition can go smoothly if the d_{turn} distance reduces linearly to zero when the imaginary vessel gets within d_{turn} meters of a line that bisects the course legs (Figure 5.4.4). The imaginary vessel essentially stops at the bisection line and waits for the actual vessel to catch up. This algorithm results in a smooth desired heading if the path K_d is zero. If K_d is not zero there is a discontinuity when the actual vessel crosses the d_{switch} radius because the vessel switches to the new path leg and sees a large velocity perpendicular to the new path.

Once the heading and path-following PID gains are set the only additional user input in this algorithm is a map of d_{turn} to the course change angle θ_c.

5.4.4 Determining the Feedforward Distance

There are four ways that the feedforward distance d_{turn} (as a function of course change angle θ_c) could be obtained: analytically, from simulation, from experiment, or adaptively. The analytical solution is very difficult because of the highly nonlinear dynamics at high yaw rates and sideslip velocities. Experiments would give the most accurate determination of d_{turn} if a sufficiently thorough test matrix were used. An initial guess could be obtained by setting d_{switch} and driving through several course change angles; the d_{turn} estimate would be the trajectory overshoot. However if an accurate nonlinear model of the ship is available then a very dense test matrix can be run in simulation. This is the approach used here.

For a set of course change angles $\theta_c = 10°$ to 90° in steps of 10° the feedforward distance d_{turn} was varied from 0 to 4 meters. The PID heading controller was used with $\lambda = 1$ and $K_{iH} = -40$. The look-ahead distance d and switching distance d_{switch} were both 5 meters. The path-following K_d was 0.25 and the path integration was turned off. The RMS cross-track error and maximum cross-track error were computed in each case from 10 meters before the turn to 15 meters after the turn (by which time the simulated vessel had converged to the path in all cases). The performance metric was the sum of the RMS error and the maximum error. This combined error is plotted in Figure 5.4.5 for a 60° azimuth saturation.

A bicubic surface was used to interpolate the data and find a polynomial fit to the functional relation between the best d_{turn} and θ_c. In general a 4th order polynomial is required to define the function but for 60° and 90° saturation angles with this particular heading controller a linear fit is sufficient. The results are plotted in Figures 5.4.6 and 5.4.7.

5.4.5 Simulation Results

The simulation was used to demonstrate the merits of the course change feedforward algorithm compared to the standard line-of-sight control system. The simulation was also used to determine the control input sequence that results in the optimal path following for course angle changes.

It is useful to compare the feedforward algorithm to the line-of-sight algorithm but it is also important to determine how well the feedforward algorithm matches the optimal control input. There exists an azimuth angle trajectory that results in the "best" path

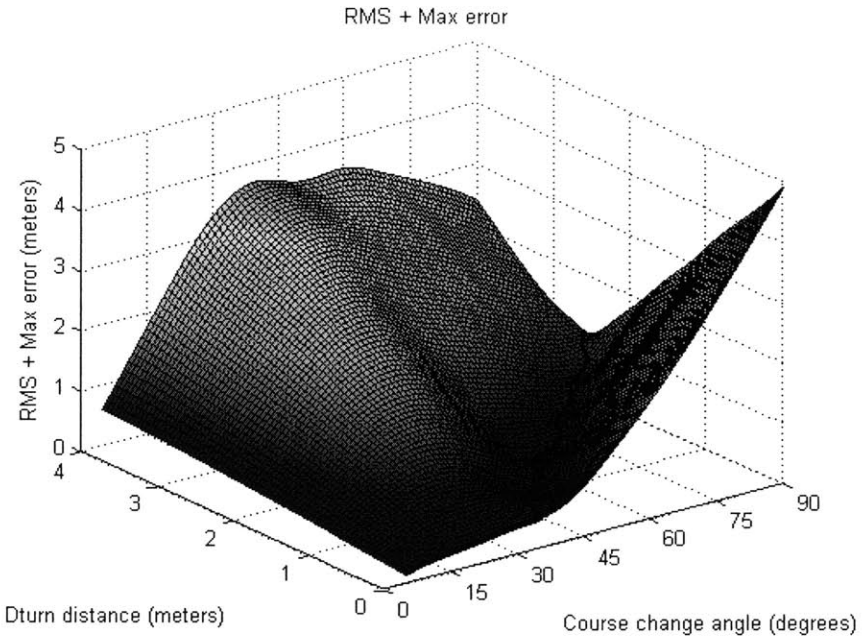

Figure 5.4.5: Combined error (RMS + Max) throughout the d_{turn}, θ_c space. The optimal d_{turn} is found in the valley (indicated with a black line). Azimuth saturation 60°.

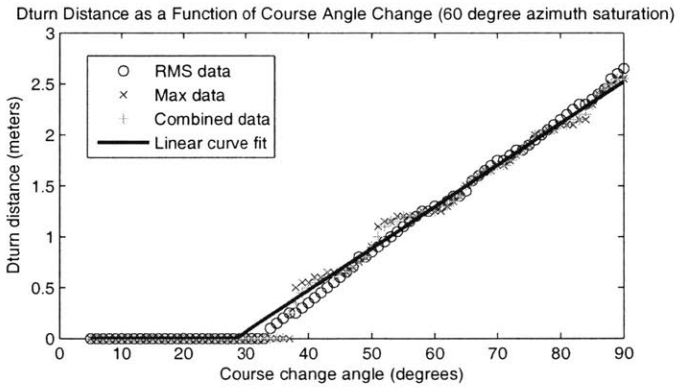

Figure 5.4.6: Optimal d_{turn} based on RMS error, maximum error, and combined error (RMS + Max). The linear curve fit corresponds to the combined data. Azimuth saturation 60°.

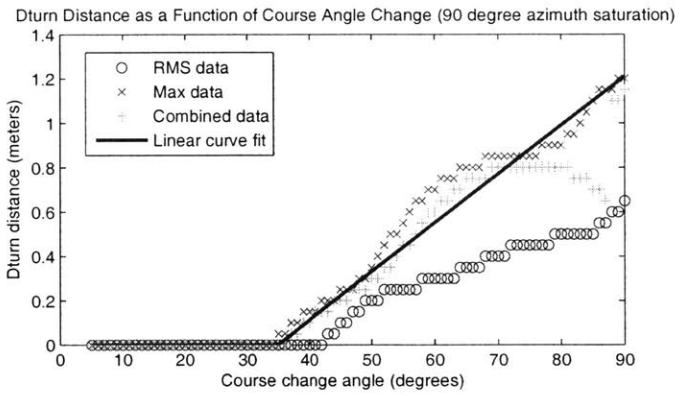

Figure 5.4.7: Optimal d_{turn} based on RMS error, maximum error, and combined error (RMS + Max). The linear curve fit corresponds to the combined data. Azimuth saturation 90°.

following by some metric. A perfect path following algorithm will generate this azimuth trajectory. We are limited by the performance of the heading controller, which actually prescribes the sideforce and thus the azimuth angle, but we can try to match the ideal control input as well as possible. (An alternative would be to determine the ideal desired heading input, therefore eliminating the heading controller limitations.)

The optimal control input takes the form of Figure 5.4.2. The azimuth angle saturates as soon as the turn begins and after a certain amount of time it recovers to the other side and settles out. The form of this curve can be represented as a piecewise linear function. The endpoints of the segments have coordinates in time and azimuth angle. The curve in Figure 5.4.2 is represented using seven control points: the start of the saturation, the end of the saturation, four intermediate points, and the final return to center. These seven control points can be represented by twelve parameters because the saturation is assumed to be at a constant azimuth angle and the curve is known to return to zero azimuth angle. These twelve parameters can be adjusted to cause the vessel to make any course change with high accuracy. This problem is solved using the Simplex algorithm in MATLAB's Optimization Toolbox. The combined error (RMS + Max) is used as an objective function; it is computed for the trajectory from 10 meters before the waypoint to 15 meters after it.

Figure 5.4.8 shows the result of the optimal control input for a 90° course change, along with the best results for the line-of-sight algorithm and the feedforward algorithm. The errors associated with each are summarized in Table 5.4.1. For a 90° course change the feedforward algorithm results in a combined error only 13% greater than the best possible

Table 5.4.1: Errors (RMS, maximum, and combined RMS + Max) for a 45° and 90° course change: Optimal (best possible) control input, Line of Sight algorithm, and Feedforward algorithm. d_{switch} for the feedforward algorithm is 5 meters.

Method	Settings	RMS (m)	Max (m)	Comb. (m)
45° Optimal		0.075	0.198	0.273
45° LOS	$d_{switch} = 5.5$ m	0.171	0.352	0.522
45° FF	$d_{turn} = 0.68$ m	0.163	0.400	0.564
90° Optimal		0.159	0.455	0.615
90° LOS	$d_{switch} = 7$ m	0.254	0.665	0.919
90° FF	$d_{turn} = 1.5$ m	0.211	0.484	0.695

control input in the same metric; the line-of-sight algorithm has a combined error 49% greater. For a 45° course change the line-of-sight algorithm is actually slightly better than the feedforward algorithm, 91% above the optimal control input compared with 107%. Here it is clear that the feedforward algorithm is best suited for large course change angles.

It should be noted from Figure 5.4.8 that the optimal control input could actually be improved slightly if it first caused the vessel to drift to the inside of the turn; then the reverse sideslip would not deviate as far from the path and the maximum error would be reduced. However that would result in errors earlier in the path, which may be undesirable for survey-type applications.

A standard course was used to evaluate the performance of the path-following controllers in simulations and in river tests (Figure 5.4.9). This course has a variety of course change angles. The first two waypoints are just used to get the kayak away from the dock; the kayak is not expected to be able to accurately follow the path change from the last leg to the first leg if the kayak is allowed to go around the course more than once. All error processing does not start until the middle of the second leg to allow for some settling onto the path. That said, the feedforward path following algorithm actually handles the acute course change angle at the first waypoint very well and reconverges to the path quickly.

The simulation was run using the interpolated line-of-sight and feedforward algorithms with azimuth saturations of 60° and 90°. The results are presented in Figure 5.4.10. The statistics of the cross-track error were computed for the portions of the path displayed in the figure. These statistics are presented in Table 5.4.2.

The feedforward algorithm performs better in simulation than the line-of-sight algorithm, but it should be noted that the feedforward algorithm is optimized for each course

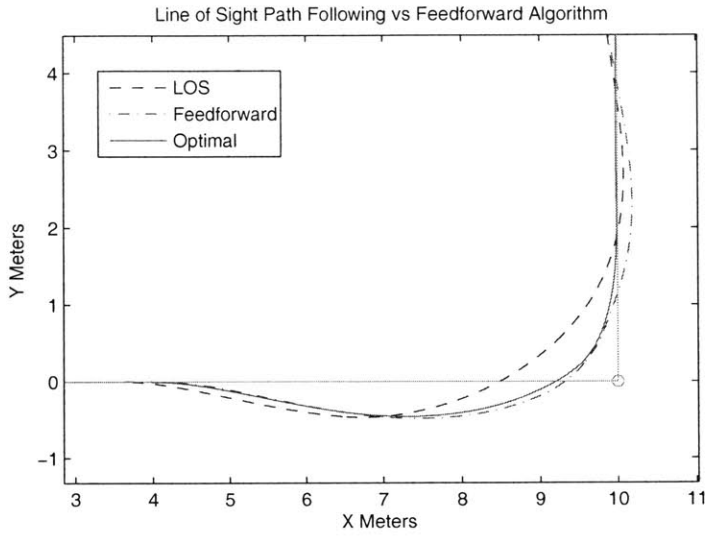

Figure 5.4.8: No amount of control can make the vessel track a 90° course change perfectly, but this plot shows the best trajectory that is possible using the metric of combined error (RMS + Max). The line-of-sight and feedforward algorithms can be compared to this standard and to each other. The feedforward algorithm initially matches the optimal trajectory but it then succumbs to oscillations due to the heading controller integrator.

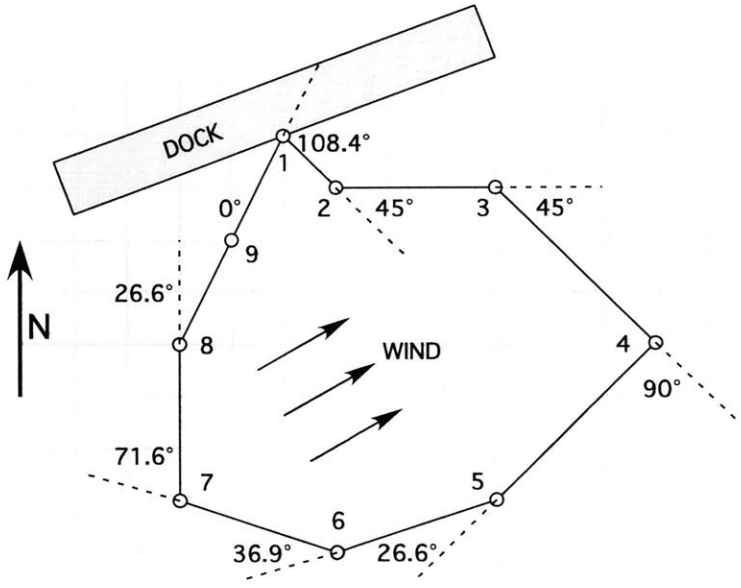

Figure 5.4.9: Course used in simulations and river tests. Grid spacing is 5 meters.

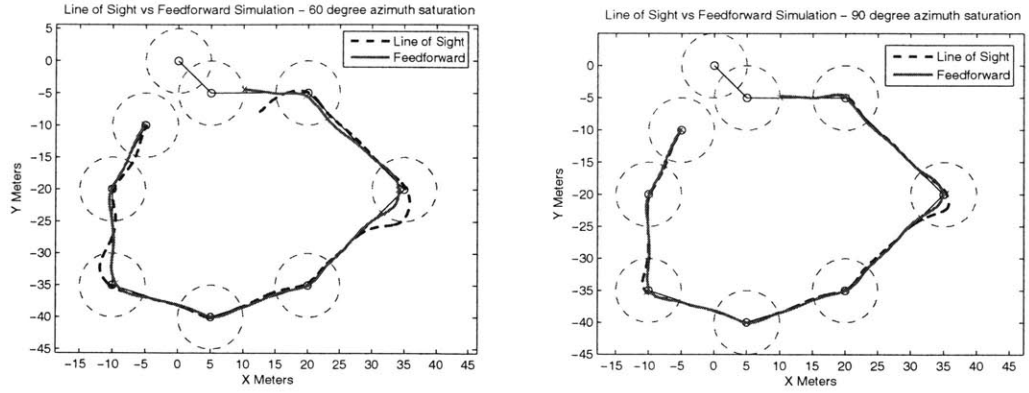

Figure 5.4.10: Simulations of the line-of-sight and feedforward path following algorithms for azimuth saturations of 60° and 90°.

Table 5.4.2: Errors (RMS, maximum, and combined RMS + Max) for the Line of Sight and Feedforward algorithms (simulation).

Method	Saturation	RMS (m)	Max (m)	Comb. (m)
LOS	60°	0.810	3.292	4.103
FF	60°	0.297	0.771	1.068
LOS	90°	0.386	1.858	2.244
FF	90°	0.217	0.613	0.830

change angle and the line-of-sight algorithm is not. The most important result from the simulations is that the errors at the waypoints can be very small: the maximum error for the feedforward algorithm at 90° azimuth saturation is less than 20% of the vessel length.

5.4.6 River Test Results

River tests on the Charles River in Cambridge/Boston, MA were used throughout this research to validate the many components of the path-following controller design. There were many challenges: the GPS signal sometimes scattered for seconds at a time, waves yawed and rolled the vessel as it tried to follow the path, and wind pushed the vessel off course. The GPS scatter was infrequent enough that if the kayak went around the course three times then at least one circuit would be clean. The roll due to waves became much less of a problem when tilt compensation was added to the compass reading. There was no solution for wind except to try to perform tests in calm weather. However numerous constraints made it necessary to test in winds as high as 24 knots.

Surprisingly tail winds were the most disruptive disturbances. Tail winds caused the vessel to move faster than its nominal speed because there is no closed-loop speed controller. This speed increase was often enough to cause the kayak to turn too late. The wind on the Charles River is predominantly from the west and southwest. Therefore there was often overshoot at the northeast waypoint. Head winds were less of a problem because the vessel can recover more easily if it has turned too soon than if it has turned too late, but they tended to make the kayak turn too soon at the southwest waypoint.

Crosswind was also a problem, most often causing the trajectory along the rightmost path to drift to the east. Integral control did a reasonable job of keeping this error in check when the path integral gain was larger than 0.01.

The results for 60° azimuth saturation are presented in Figure 5.4.11. These tests were performed in winds that averaged 12 knots with gusts to 24 knots. The wind gusts are responsible for the steady state error on the rightmost leg and the undercut of the lower left waypoint. The statistics for this data, over the same region as for the simulation, are presented in Table 5.4.3

The algorithms are compared again in Figure 5.4.12. The feedforward 90° algorithm clearly performs the best. At high azimuth angles the kayak comes to a complete stop and rotates about at point at its bow. This effect does not appear in the simulation because

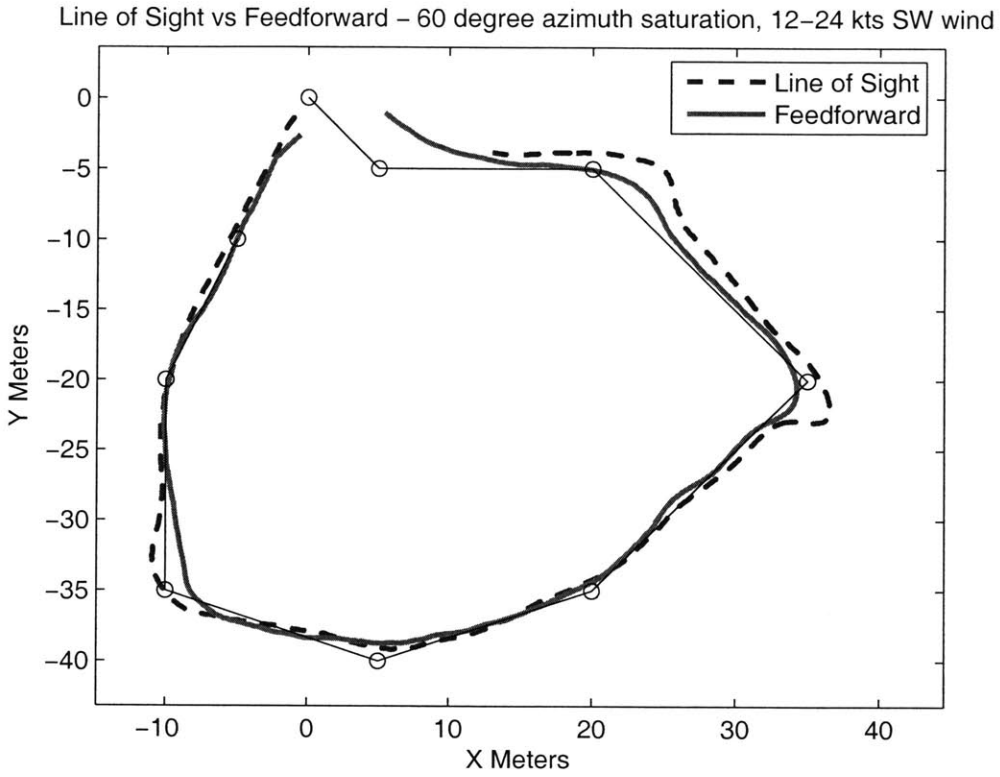

Figure 5.4.11: Path following using the line-of-sight and feedforward algorithms with a 60° azimuth saturation. Wind was from the southwest (lower left), 12-24 knots. $K_i = 0$ in the right plot. Data taken November 13, 2005.

Table 5.4.3: Errors (RMS, maximum, and combined RMS + Max) for the Line of Sight and Feedforward algorithms (data from Figure 5.4.11).

Method	Saturation	RMS (m)	Max (m)	Comb. (m)
LOS	60°	1.057	3.498	4.555
FF	60°	0.554	1.555	2.099

the nonlinear hydrodynamic coefficients on which the simulation is based are designed for the nominal speed of the hull; the 3rd order model is not sufficient to capture the dynamics when $u = 0$. Nonetheless the feedforward algorithm, based on simulation data, performs well at the 90° course change. The path integrator gain K_i was zero for the test presented in Figure 5.4.12 (right plot) so the steady error in the lower leg is understandable given the WSW wind. The line-of-sight algorithm began its turn too late at the 90° course change and was blown off course as it came to a stop to make the corrections to get back on track. As mentioned in the discussion of the simulation results the line-of-sight algorithm is not tuned to the course changes, although it easily could be. In fact at the 45° turn, where (for 90° azimuth saturation) the feedforward distance is small anyway, the line-of-sight algorithm performs well.

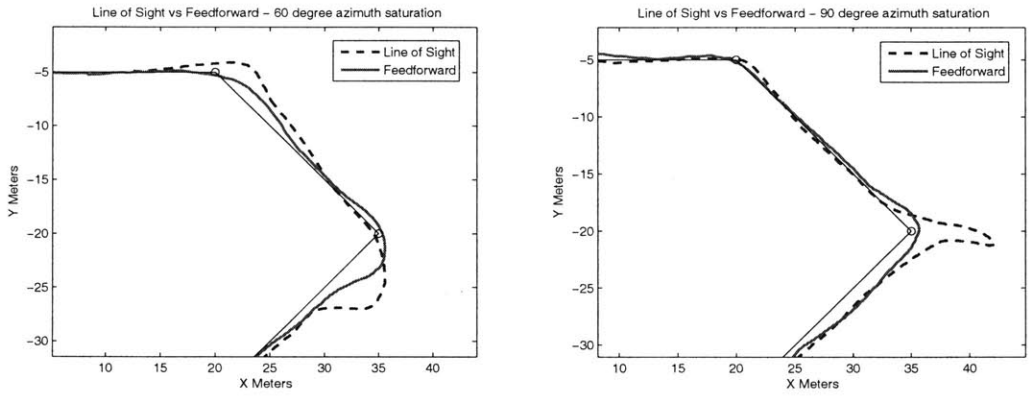

Figure 5.4.12: Comparison of line-of-sight and feedforward algorithms for 60° and 90° azimuth saturations. Of the four controllers the feedforward algorithm saturated at 90° performs the best. Data taken November 18, 2005 and November 21, 2005.

5.5 Summary

The standard line-of-sight algorithm is guaranteed to be stable, so it is the logical choice for path following. The line-of-sight algorithm is, for small cross-track errors, a quasi-P controller; its performance can be enhanced by adding trajectory-based derivative and integral terms to make a quasi-PID path-following controller. The extra terms reject steady disturbances and reduce overshoot when recovering from large path errors. Linear stability analysis can be used on the resulting inner-loop / outer-loop heading and path-following

control systems to determine the appropriate path-following PID gains.

The path-following controller can be enhanced further by creating a feedforward action near waypoints so that the vessel trajectory matches the desired path as well as possible. The feedforward algorithm needs only one input: a map of the feedforward distance d_{turn} to the course angle change θ_c. This map is found using the simulation of the entire system. This algorithm matches the best-possible trajectory very well. The algorithm's performance was confirmed in river tests.

Chapter 6

Conclusions

In the literature there has been very little study of control techniques for vessels with azimuthing podded propulsion. The unique features of this form of propulsion are that a much greater sideforce and yaw rate can be generated and this force can be created even if the vessel is not moving forward. For these reasons podded propulsion can enhance the maneuverability of surface vessels.

6.1 Contributions and Results

Although the vessel dynamics are highly nonlinear at large yaw rates and sideslip velocities, a linear heading controller performs nearly as well as a nonlinear controller. Feedback linearization is the only nonlinear control technique presented here, but even in simulation with perfect parameter knowledge the controller did not perform significantly better than the linear PD controller and its performance actually decreased relative to the linear controller when an integral term was added.

The interpolated line-of-sight waypoint tracking method has been proven to be stable given a stable heading controller. Adding integral and derivative terms to the path-following controller improves the performance further. Using a space-based integral instead of the standard time-based integral can reduce integral wind-up and improve recovery from large path errors. Linear analysis was used to prove stability of the heading and path-following controllers and help choose the path-following PID gains.

Although podded propulsion systems offer greater maneuverability and therefore more precise path following on their own, an additional improvement can be made to anticipate

the non-minimum phase behavior of the system and the sideslip associated with large side-forces. The feedforward algorithm exploits these features to create vessel trajectories that match the desired path nearly as well as physically possible given the limitations of the heading controller and the constant thrust assumption.

All of these results were confirmed with river tests using an autonomous kayak with an onboard computer and various sensors.

6.2 Future Work

The development of the linear and nonlinear heading controllers and the path-following algorithm is highly model-dependent; the system parameters must be identified to a fair degree of accuracy for the procedures outlined here to work for an arbitrary vessel. It was fortunate that hydrodynamic data was available for the autonomous kayak and that it could be easily confirmed and adjusted in river tests. However, it would be more useful if the controllers could be designed using a minimal amount of data specific to the vessel. Ideally the relevant model parameters could be determined from a handful of simple river tests so that these tools could be easily applied to other vessels for which hydrodynamic data is not readily available.

Throughout this research the input torque to the propeller was assumed to be constant (that is, the vessel was operated at a constant throttle setting). Clearly path following could be improved if the vessel speed were reduced at the corners. In fact the advantages of podded propulsion over ruddered vessels would become even more apparent because rudders lose effectiveness at low speeds. A closed-loop speed controller and high-level cost function could be added to determine the optimal speed to enter each waypoint given path error allowances and timing constraints.

Environmental forces and currents are currently countered with trajectory-based integral control. If the forces and currents are steady in a global frame then their cross-track components are different for different path leg angles. A more practical approach is to estimate the forces and currents in a global frame and use environmental feedforward terms to correct for them in the local path frame as in [2]. This addition, combined with closed-loop speed control and a speed-dependent feedforward algorithm, would make the path-following controller extremely precise and robust.

6.3 Final Thoughts

Although the application for precise path following most often cited in this thesis is surveying or mapping, the techniques presented here can be applied to a larger set of problems. In particular, high-performance path following can and should be used with dynamically-generated paths in situations such as docking maneuvers, rescue maneuvers, and collision avoidance. Furthermore there is no need to restrict the application to surface vessels, although many underwater vehicles are more fully actuated and can perform precise maneuvers by distributing control action over several thrusters on the hull.

The feedforward algorithm was motivated in part by human intuition: a human operator of a small craft deals with non-minimum phase behavior by anticipating the lag and starting the turn earlier than if the system were minimum-phase. The resulting algorithm is very simple yet near-optimal. If simple algorithms can be created to match human intuition then many nonlinear behaviors or complicated maneuvers could be handled easily with automatic control.

Bibliography

[1] A. Pedro Aguiar, João P. Hespanha, and Petar V. Kokotović. Path-following for non-minimum phase systems removes performance limitations. *IEEE Transactions on Automatic Control*, 50, 2005.

[2] V. Bakaric, Z. Vukic, and R. Antonic. Improved basic planar algorithm of vehicle guidance through waypoints by the line of sight. In *1st International Symposium on Control, Communications, and Signal Processing.* IEEE, 2004.

[3] Morten Breivik and Thor I. Fossen. Path following for marine surface vessels. In *Oceans*, 2004.

[4] Morten Breivik and Thor I. Fossen. Path following of straight lines and circles for marine surface vehicles. In *IFAC Conference on Control Applications in Marine Systems*, 2004.

[5] Thor I. Fossen. *Guidance and Control of Ocean Vehicles.* John Wiley & Sons Ltd., 1994.

[6] Isao Funeno. Hydrodynamic development of azimuthing podded propulsion system. In *9th International Symposium on Practical Design of Ships and other Floating Structures.* IEEE, 2004.

[7] M. Gopal. *Modern Control System Theory.* Wiley, 2nd edition, 1993.

[8] Thomas Holzhüter and Roland Schultze. Operating experience with a high-precision track controller for commercial ships. *Control Engineering Practice*, 4(3), 1996.

[9] Knut Eilif Husa and Thor I. Fossen. Backstepping designs for nonlinear way-point tracking of ships. In *Manoeuvring and Control of Marine Craft*, 1997. Proceedings of the 4th IFAC Conference.

[10] A. Lozowicki and A. Tiano. On the design of a high precision ship track-keeping system. In *Manoeuvring and Control of Marine Craft*, 2000. Proceedings of the 5th IFAC Conference.

[11] Euan W. McGookin, David J. Murray-Smith, Yun Li, and Thor I. Fossen. Parameter optimisation of a non-linear tanker control system using genetic algorithms. In *Genetic Algorithms in Engineering Systems: Innovations and Applications*, 1997.

[12] K. Y. Pettersen and E. Lefeber. Way-point tracking control of ships. In *Proceedings of the 40th IEEE Conference on Decision and Control*, 2001.

[13] Sandia National Laboratories, NACA 0015 data. Reported at website http://www.aerospaceweb.org/question/airfoils/q0150b.shtml.

[14] Roger Skjetne. *The Maneuvering Problem.* PhD thesis, Norwegian University of Science and Technology, 2005.

[15] Jeffrey Stettler. *Steady and Unsteady Dynamics of an Azimuthing Podded Propulsor Related to Vehicle Manufacture.* PhD thesis, Massachusetts Institute of Technology, 2005.

[16] Jasmin Velagic, Zoran Vukic, and Edin Omerdic. Adaptive fuzzy ship autopilot for track-keeping. In *Manoeuvring and Control of Marine Craft*, 2000. Proceedings of the 5th IFAC Conference.