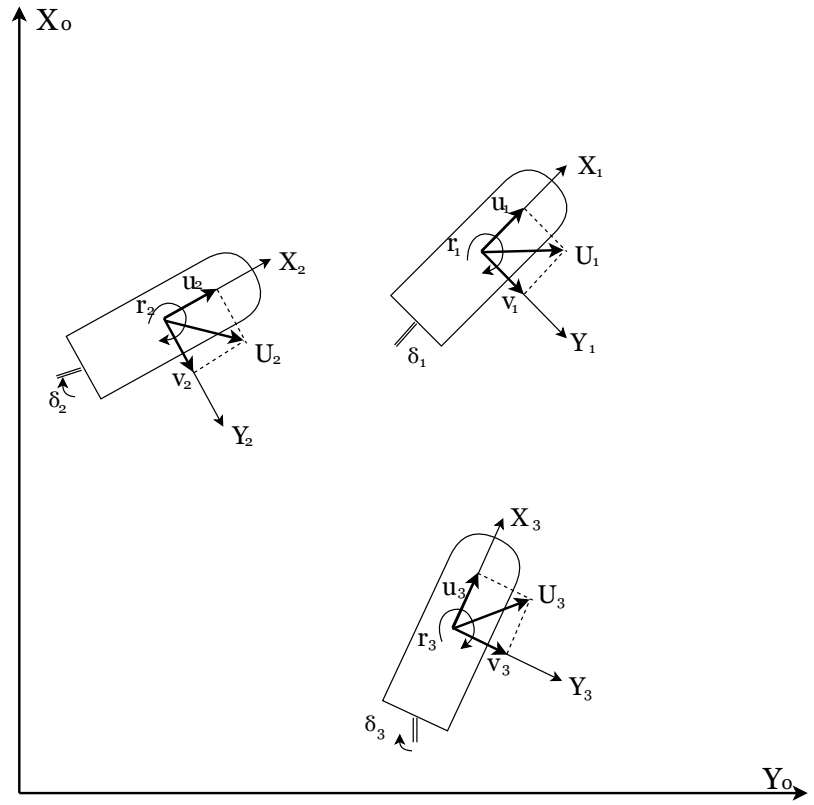
CHALMERS
UNIVERSITY OF TECHNOLOGY



Coordinated Marine Vessel Formations

A control approach for developing an experimental platform

Master's thesis in Systems, Control and Mechatronics

Carl Hjerpe
Wendy Mo

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

Master's thesis 2019

Coordinated Marine Vessel Formations

A control approach for developing an experimental platform

Carl Hjerpe
Wendy Mo

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
Gothenburg, Sweden 2019

Coordinated Marine Vessel Formations
A control approach for developing an experimental platform
CARL HJERPE
WENDY MO

Supervisor: Jonas Williamsson, Consat Engineering AB
Examiner: Balázs Kulcsár, Chalmers University of Technology

Master's Thesis 2019
Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Reference frames for three boats, each with a local coordinate system.

Typeset in LaTeX
Gothenburg, Sweden 2019

Coordinated Marine Vessel Formations
A control approach for developing an experimental platform
Carl Hjerpe, Wendy Mo
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Formation control of marine vessels may be relevant for saving fuel consumption and manual labor in several application areas such as in icebreaker escorting or collecting marine debris. With one manually steered boat, it is investigated if it is possible to develop a system where other vessels can follow it autonomously to an extent. For this purpose, an experimental platform has been implemented for testing different control strategies in a small scale environment. Three controllers, PID, LQR, and LQI, have been evaluated for regulating rudder angle and thrust input. It was found that the LQR performs the best while the LQI was second best in simulation but not in the experimental tests. This may partly be due to the lack of a proportional action or an inadequate accuracy of the model. Nonetheless, it was found that an experimental platform is helpful for understanding coordination of boats and there are controllers that can fulfill this purpose. It may even be easier in some sense to apply this on larger vessels due to the formulation of the model and the access to other kinds of data.

Keywords: control, marine vessel, boat, formation, maritime dynamics.

Acknowledgements

Firstly, we would like to thank Balázs Kulcsár for your excellent supervision and constant encouragement. Your advice and input have been of great help throughout the whole project. We also want to thank Jonas Williamsson for the opportunity to perform this thesis at Consat and for all the positive support. Lastly, we would like to give a big thanks to all the great people at Consat for giving a hand when needed and for insightful discussions and feedback. Thank you.

Carl & Wendy, May 2019

Contents

List of Figures **xi**

List of Tables **xiii**

1 Introduction **1**
1.1 Project Aim . . . 2
1.2 Scope and Limitations . . . 2
1.3 Report Structure . . . 2

2 Technical Background **3**
2.1 Formation Control . . . 3
2.2 Coordinate Frames . . . 5
2.3 Nonlinear Mathematical Model of Ship . . . 5
2.4 Linearization of Nonlinear Model . . . 7
2.5 Control Systems . . . 7
2.5.1 PID . . . 7
2.5.2 LQR . . . 8
2.5.3 LQI . . . 9
2.6 Discretization of LQR and LQI . . . 10
2.7 Anti-Windup . . . 11

3 System Development **13**
3.1 Linearization of Model . . . 13
3.2 Control System Development . . . 14
3.2.1 PID Design . . . 14
3.2.2 LQR Design . . . 15
3.2.3 LQI Design . . . 15
3.3 Simulation Environment . . . 16
3.4 System Architecture . . . 17
3.5 Global Angle Estimation . . . 20
3.6 Speed Estimation . . . 21
3.7 Testing Methods . . . 22

4 Results **25**
4.1 Simulation of System . . . 25
4.2 Experimental Results . . . 29
4.2.1 Platform . . . 29

4.2.2 Dry Tests 30
4.2.3 Wet Tests 30

5 Discussion **33**
5.1 Mathematical Model and Real-Life Differences 33
5.2 Evaluation of Controllers 34
5.3 Evaluation of Platform 34
5.4 Future Work 35

6 Conclusion **37**

List of Figures

2.1 Block scheme of a decentralized leader-follower system. u is the manual input to the leader boat. Each follower has its own control system. 3
2.2 World frame with three boats with different coordinate frame locally. 4
2.3 Coordinate frame of ship with 3DOF. 5
2.4 Block diagram of PID regulator in a frequency domain. 8
2.5 Block diagram of the LQR. 9
2.6 Block diagram of the LQI without K_r. 10
2.7 Block diagram of conditional windup. 12

3.1 Comparison of the resulting angle and speed between the nonlinear and linear models. 14
3.2 Block diagram of system hardware on a follower boat. 18
3.3 RC boats used in the project, with a red bucket placed on the leader and a camera mounted on a follower. 18
3.4 Overview of ROS nodes and topics in the system. 19

4.1 Controllers follow the leader trajectory with different success. 25
4.2 Angle error of PID controller in simulation oscillates but stays stable. 26
4.3 Angle error of the LQR has peaks from the turnings in the beginning but remains stable later on. 26
4.4 When the heading is straight the angle error is smaller for the LQI than the other controllers. 27
4.5 Magnitude of PID error speed resultant. 28
4.6 Magnitude of LQR error speed resultant. The error never exceeds 0.7 m/s. 28
4.7 Magnitude of LQI error speed resultant. The LQI is more uncertain in the error damping than the LQR but better than the PID. 29
4.8 Angle error of the PID over time. Here it can be seen that the error drifts away and the controller does not manage to keep it around 0 over time. 31
4.9 LQR corrects the difference between the follower and leader angle. . 31
4.10 LQI tries to correct the difference between the follower and leader angle, but with little success. 32

List of Tables

3.1 Parameters for the PID controllers. 15
3.2 Description of each ROS topic message and type. 20

4.1 Accumulative error of angle and speed for the three controllers in simulation. 29
4.2 Accumulative angle error for the three controllers in wet test. 32

Nomenclature

δ	Rudder angle
γ	Experimental constant of longitudinal inflow velocity
$\Gamma_{\dot{\theta}}$	Threshold for $\dot{\theta}$
Γ_{Imax}	Top threshold for anti-windup
Γ_{Imin}	Bottom threshold for anti-windup
μ_R	Flow straightening coefficient
ρ	Density of water
τ	Propeller thrust
τ_0	Base thrust
I^2C	Inter-integrated circuit
θ	Global angle of ship
a_H	Increase factor of rudder force
A_R	Profile area of movable part of mariner rudder
f_α	Rudder lift gradient coefficient
J_z	Added moment of inertia
L_{pp}	Ship length
m_x	Added mass in x-axis
m_y	Added mass in y-axis
t_p	Deduction factor for thrust
t_R	Deduction factor for steering resistance
T_s	Sampling time of system
U_r	Resultant inflow velocity to rudder
x_G	Distance between center of gravity and center of ship.
x_H	Longitudinal coordinate of additional lateral forces
x_R	-0.5 L_{pp}, longitudinal coordinate of rudder position
d	Ship draft
I	Moment of inertia around center of gravity
m	Ship's mass
r	Yaw rate
u	Surge velocity at center of gravity
v	Lateral velocity at center of gravity
CI	Conditional integral
CSI	Camera serial interface
DOF	Degrees of freedom
ESC	Electronic speed controller
PWM	Pulse width modulation
ROS	Robot operating system

1
Introduction

Coordination and formation control are two relevant research topics for various types of vehicles [1] [2] [3] where the main objective is to ensure that objects keep a formation while moving. For these purposes it is common to have control laws that enforce feedback control [4] and the complexity of these can vary depending on the tasks.

Formation control is not only limited to suit objects on land, but control within marine settings is a rising topic as well [5]. Having marine vessels keeping a formation can be of interest in many application areas, with examples ranging from multi-coordinated vessels posing as an aqua rake for collecting marine debris to ice-breaker escorting to replenishment at sea.

These systems often require large crews on board, but resources may be saved by the rising popularity of automation and autonomy. Autonomous vessels can be implemented to take more energy efficient routes than humans, and if well-equipped with relevant sensors, the input data from them can be used to predict and counteract incoming disturbances. For this to work, good models of the system considering potential disturbances, such as wind and waves, and fine control of the autonomous vessels are needed for them to be able to take the most efficient routes.

To reach a fully autonomous system may take another while, but a good start would be to keep a manually steered leader and have followers autonomously maneuvered in a formation [6]. Here the goal of the followers would be to keep its position with a certain distance or angle in relation to the leader. This would in turn create the necessity of keeping the same speed as the leader and ensure that errors are accounted for.

For purposes such as in a leader-follower framework, a decentralized control system becomes relevant to allow more freedom in the scaling into larger networks. In such a system, several subsystems work together towards a global goal, but each is responsible for a its own control actions. By measuring states and computing control actions locally, the need of a global model is redundant and may also reduce computation complexity significantly [7].

A leader-follower system additionally requires good communication between the vessels in a way that ensures that they stay coordinated and synchronised throughout the journey all while avoiding collision. All of this, in addition to the concepts

mentioned above, may work well in theory but it is the physical implementation of this that is an interesting objective for this thesis project. A good start before applying this on large vessels is to test such a system on a prototype version first.

1.1 Project Aim

This master's thesis project aims to find suitable control strategies for formation of marine vessels. It is intended to create a development platform which may be used for testing and evaluating control systems in marine vessels.

1.2 Scope and Limitations

Different formations of marine vessels will be implemented and tested. Due to limitations in time and resources, this project will be evaluated by a small boat platform with restrictions upon controllers and sensors. All controllers that are evaluated should be easy to implement. To simplify the modelling a bit, the boats will be evaluated in three dimensions but two dimensional conditions will be assumed.

There will also be a limit to the number of vessels present in the system, even though the controller should work well if scaling-up is needed. Since the design of the controllers is the focus here, object avoidance will not be prioritized in this project. The system will only be made as a prototype and therefore not implemented as a finished commercial product.

1.3 Report Structure

Technical background is the first section of the report and will describe relevant theory for understanding the approaches used in the project. The following chapter, System Development, explains the design choices, methods, and tools used before the Results chapter which outlines the platform and test case findings. Towards the end the reader will find a section about discussion and evaluation of the project before finishing with a short conclusion.

2
Technical Background

This chapter briefly describes the relevant theory for this thesis. First formation control will be defined and then the mathematical model of a boat will be introduced in addition to the concepts of a few controllers. Since the leader and follower will operate in different frames, the coordinate frames will be outlined as well. Lastly the theory behind discretization of controllers and the concept of anti-windup will be briefly explained.

2.1 Formation Control

This project defines formation control as having boats following the same trajectory as a leader that is manually steered. It is not necessary to keep the formation in ways such that there has to be an explicit shape but rather that the follower should try to keep the same angle and speed as the leader. With this, a controller approach can be used and no implementation of path planning is needed. To avoid collision, the boats are initially placed with some distance between them.

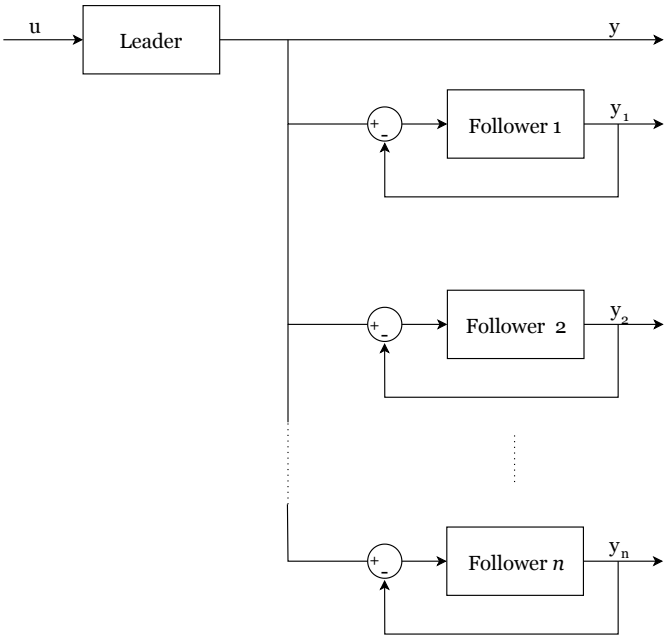


Figure 2.1: Block scheme of a decentralized leader-follower system. u is the manual input to the leader boat. Each follower has its own control system.

A general flowchart of the intended structure of the system is shown in Figure 2.1. Here it can be seen that the information from the leader, such as its heading, is sent out to all the followers and they are then able to use the data acquired in their local calculations. Each follower block can be viewed as the whole follower itself, with each containing a control system. The input to the followers are the difference between their own states and the leader's states.

The platform will use a decentralized system. Each follower will compute the error between its states and the leader's states with measurements from sensors. They will be equipped with both an IMU and a camera to acquire data from its environment. This allows calculations to take place locally on each boat and yield the control outputs directly instead of sending this information back and forth to a main computer. Decentralization is good for expanding the system to include more vessels later.

Figure 2.2 shows a system overview of how the three ships have different local coordinate systems in the world frame. It is used as a reference in order to be able to convert the angles and speed from the leader's frame to the follower ship's own frame.

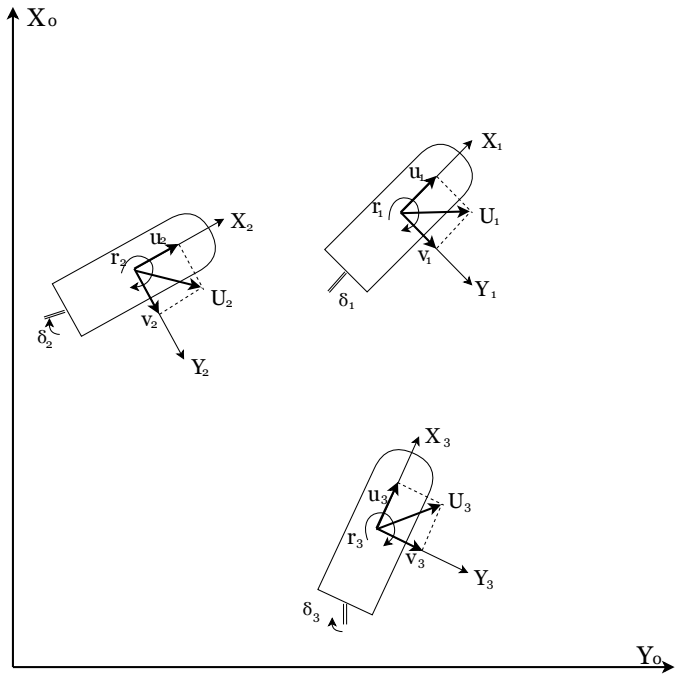


Figure 2.2: World frame with three boats with different coordinate frame locally.

For the follower to keep the formation the controllers need to generate a stable system. In this thesis the stability only needs to be local. For a solution to be considered locally stable [8], stability is needed for all initial conditions $x \in B_r(a)$, where $B_r(a)$ is defined by Equation 2.1. Only local stability is required since the boats start in a stable position and should only deviate from this starting formation by a little and this is a restriction chosen onto the model.

$$B_r(a) = \{x : ||x - a|| < r\} \tag{2.1}$$

2.2 Coordinate Frames

To be able to control the errors between a follower and the leader, the states need to be represented in the same coordinate frame. The measured speed and distance to the leader should be represented in the follower's own coordinate frame. Two rotation matrices are needed for the conversion and these use the follower's global angle θ_f and the leader's global angle θ_l. The first matrix converts the leader's speed to the world frame and the second translates the speed in the world frame into the follower's own frame. The final rotation matrix is the product of these two rotation matrices as found in Equation 2.2.

$$R = \begin{bmatrix} cos\theta_f & -sin\theta_f \\ sin\theta_f & cos\theta_f \end{bmatrix} \begin{bmatrix} cos\theta_l & -sin\theta_l \\ sin\theta_l & cos\theta_l \end{bmatrix} \tag{2.2}$$

2.3 Nonlinear Mathematical Model of Ship

Ship motion is usually described using six degrees of freedom, but for ship maneuvering it is common to simplify it and use only three degrees of freedom. A common mathematical model to use is the MMG standard for ship maneuvering predictions found in [9]. Notations are derived using the local coordinate frame as displayed in Figure 2.3.

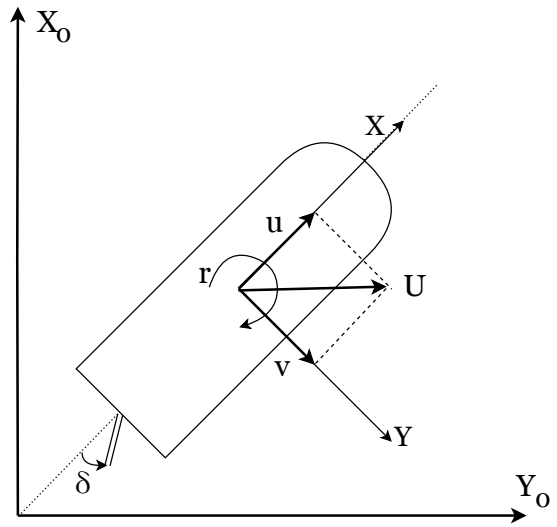


Figure 2.3: Coordinate frame of ship with 3DOF.

The mathematical model used for the boat in this thesis is therefore a three degrees of freedom model that consists of velocities in axes x and y, denoted u and v respectively, as well as the yaw rate, r [10]. Their relations are such as displayed in

Equations 2.3-2.5 and are based on Newton's second law.

$$(m + m_x)\dot{u} - (m + m_y)vr - x_g m r^2 = X_H + X_R + X_P \tag{2.3}$$

$$(m + m_y)\dot{v} + (m + m_x)ur + x_g m\dot{r} = Y_H + Y_R \tag{2.4}$$

$$(I + x_g^2 m + J_z)\dot{r} + x_g m(\dot{v} + ur) = N_H + N_R \tag{2.5}$$

X_P is the force generated by the propeller thrust, τ, acting on the boat and is described in Equation 2.6.

$$X_P = (1 - t_p)\tau \tag{2.6}$$

Equations 2.7-2.9 show X_H, Y_H, N_H, which are the hydrodynamic forces where $v' = \frac{v}{\sqrt{v^2+u^2}}$ and $r' = \frac{rL_{pp}}{\sqrt{v^2+u^2}}$:

$$X_H = \frac{\rho L_{pp} d}{2}(u^2 + v^2)(-R_0 + X_{vv}v'^2 + X_{vr}v'r' + X_{rr}r'^2 + X_{vvvv}v'^4) \tag{2.7}$$

$$Y_H = \frac{\rho L_{pp} d}{2}(u^2 + v^2)(Y_v v' + Y_r r' + Y_{vvv}v'^3 + Y_{vvr}r'v'^2 + Y_{vrr}v'r'^2 + Y_{rrr}r'^3) \tag{2.8}$$

$$N_H = \frac{\rho L_{pp}^2 d}{2}(u^2 + v^2)(N_v v' + N_r r' + N_{vvv}v'^3 + N_{vvr}r'v'^2 + N_{vrr}v'r'^2 + N_{rrr}r'^3) \tag{2.9}$$

X_R, Y_R, N_R are the forces created by the rudder acting on the boat, depending on the rudder angle (δ). Here $v_R = \sqrt{v^2 + u^2}\mu_R \cdot (tan^{-1}(\frac{-v}{u}) + \frac{r'}{2})$ and $u_R = u \cdot \gamma$. These are shown in Equations 2.10-2.12.

$$X_R = -\frac{1}{2}(1 - t_R)\rho A_R(v_R^2 + u_R^2) f_\alpha sin\left(\delta - \frac{v_R}{u_R}\right) sin(\delta) \tag{2.10}$$

$$Y_R = -\frac{1}{2}(1 + a_H)\rho A_R(v_R^2 + u_R^2) f_\alpha \left(\delta - \frac{v_R}{u_R}\right) cos(\delta) \tag{2.11}$$

$$N_R = -\frac{1}{2}(x_R + a_H x_H)\rho A_R(v_R^2 + u_R^2) f_\alpha \left(\delta - \frac{v_R}{u_R}\right) cos(\delta) \tag{2.12}$$

It is worth noting that these models have been derived from larger marine vessels and have been scaled down to suit the physical platform in this project.

2.4 Linearization of Nonlinear Model

In order to derive the controllers, the system needs to be represented in a linear format as shown in Equation 2.13.

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{2.13}$$

This is done using Jacobian Linearization [8]. The model is linearized around a point $\begin{bmatrix} x_0 & u_0 \end{bmatrix}^T$. The input to the system is split between the linearization point and the offset around and it is formulated as seen in Equation 2.14.

$$\begin{aligned} x(t) &= x_0 + \delta x(t) \\ u(t) &= u_0 + \delta u(t) \end{aligned} \tag{2.14}$$

To linearize the system, the equations need to be formulated according to Equation 2.15.

$$\begin{aligned} \frac{dx}{dt} &= f(x, u) \\ y &= h(x, u) \end{aligned} \tag{2.15}$$

The linearized matrices of the nonlinear system is then derived according to Equations 2.16. It is worth noting that the new linear system is only an approximation of the original system around the the linearization point.

$$A = \left.\frac{\partial f}{\partial x}\right|_{(x_0,u_0)}, \quad B = \left.\frac{\partial f}{\partial u}\right|_{(x_0,u_0)}, \quad C = \left.\frac{\partial h}{\partial x}\right|_{(x_0,u_0)}, \quad D = \left.\frac{\partial h}{\partial u}\right|_{(x_0,u_0)} \tag{2.16}$$

For the linearization of the system specifically, see later parts of the report.

2.5 Control Systems

The aim is seen from a control system approach and the following section will briefly describe which controllers are of relevance for the project. Three controllers have been chosen - PID, LQR, and LQI- for their simplicity and low computational cost.

2.5.1 PID

The PID (*proportional-integral-derivative controller*) is often considered to be one of the most basic controllers[8]. It works by continuously computing the difference between a reference value and the process output. It then tries to correct the error based on proportional, integral, and derivative coefficients, denoted K_p, K_i, and K_d respectively. PID controllers do not have any support for multiple inputs and multiple outputs. Equation 2.17 describes the controller mathematically.

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{2.17}$$

In Figure 2.4 a block diagram of the PID controller can be seen with its three different parts. The box called $P(s)$ is not part of the controller but rather the plant that is being controlled. If it is possible to describe the mathematical models of the plant, it can be used in the derivations of the control parameters. Even though the PID controller can be based on the mathematical model of the plant, it is not in any way a guarantee of an optimal control policy.

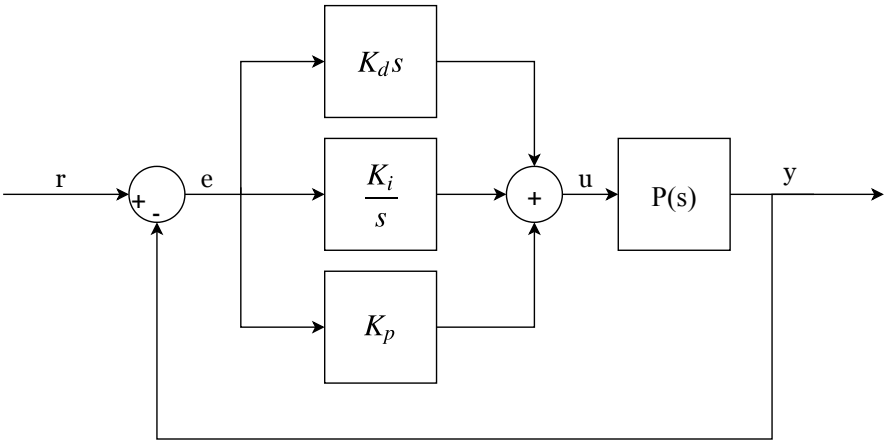


Figure 2.4: Block diagram of PID regulator in a frequency domain.

2.5.2 LQR

A *linear-quadratic regulator* (LQR) minimizes a cost function to select appropriate gains of a state feedback controller [8]. Given a linear system as expressed in its state-space form, Equation 2.13, it strives to optimize the quadratic cost function defined in Equation 2.18. *x(t)* denotes the states of the system and *u(t)* is the input to the system.

$$J = \frac{1}{2} \int_0^\infty (x^T Q x + u^T R u)dt \tag{2.18}$$

The LQR has optimal control law and solution as shown in Equation 2.19.

$$u(t) = -R^{-1} B^T P x(t) = -K x(t) \tag{2.19}$$

P is the solution to the continuous algebraic Riccati equation. It is defined in Equation 2.20 and is solved offline.

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{2.20}$$

In order for the controller to reach a setpoint, a K_r matrix is needed to convert the setpoint value to the corresponding value sent to the plant. The Equation for the K_r can be seen in Equation 2.21. In order to create the K_r matrix, the number of states in the system needs to be the same as the number of inputs to the system.

$$K_r = (D - (C - DK)(A - BK)^{-1}B)^{-1} \tag{2.21}$$

$Q \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{p \times p}$ are weight matrices which describe the convergence rate of the solution against the cost of the control [8]. These matrices can simply be designed to set the weights diagonally. This will allow each diagonal element to directly influence its corresponding state and also its impact on the overall cost. Furthermore, it is required that $Q \geq 0$ and $R > 0$ as well as the need to satisfy reachability and controllability conditions.

For continuous-time systems, reachability and controllability properties are equivalent and it is therefore enough for the LQR to meet either of the requirements. A linear system is reachable if there is an input that can steer the system from a given point x_0 to a desired final point x_f. The reachability matrix found in Equation 2.22 should be of full rank, i.e. $\text{rank}(\mathcal{R}_n) = n$ where n is number of states in the state vector.

$$\mathcal{R}_n = \begin{bmatrix} B & AB & ... & A^{n-1}B \end{bmatrix} \tag{2.22}$$

Figure 2.5 shows an overview of the feedback gain K in a closed loop of the LQR.

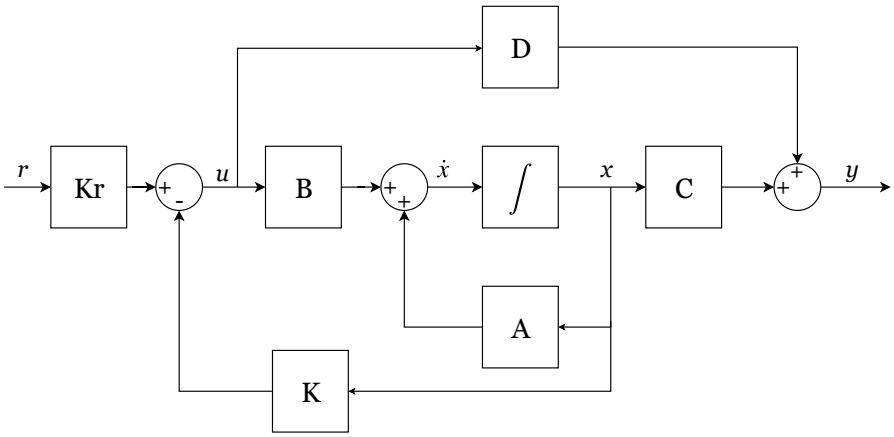


Figure 2.5: Block diagram of the LQR.

2.5.3 LQI

An alternative to the LQR is to add integral feedback to it, creating an LQI (*linear quadratic integral control*)[8]. The aim of the integrator is to zero the steady-state error which is done by augmenting the linear system into Equation 2.23. The reachability of the original state-space apply on the newly augmented one as well.

$$\begin{aligned}\begin{bmatrix}\dot{x}(t)\\ \dot{z}(t)\end{bmatrix} &= \underbrace{\begin{bmatrix}A & 0_{n\times q}\\ C & 0_q\end{bmatrix}}_{A_{LQI}}\begin{bmatrix}x(t)\\ z(t)\end{bmatrix} + \underbrace{\begin{bmatrix}B & 0_{n\times q}\\ D & -I_q\end{bmatrix}}_{B_{LQI}}\begin{bmatrix}u(t)\\ r(t)\end{bmatrix}\\ y(t) &= \underbrace{\begin{bmatrix}C & 0_q\end{bmatrix}}_{C_{LQI}}\begin{bmatrix}x(t)\\ z(t)\end{bmatrix} + Du(t)\end{aligned} \tag{2.23}$$

The new matrices A_{LQI}, B_{LQI}, C_{LQI}, and the old D matrix are used with the Riccati equation 2.20. The new control law then takes the form shown in Equation 2.24. K_z is the integral term and K_r is an optional parameter to set the desired steady state.

$$u(t) = -Kx(t) - K_z z(t) + K_r r(t) \tag{2.24}$$

With the addition of the integral action, matrix Q is expanded to accommodate the added states. As with the LQR, the matrices Q and R are modified to change the cost of the equations. A block diagram of the LQI is found in Figure 2.6.

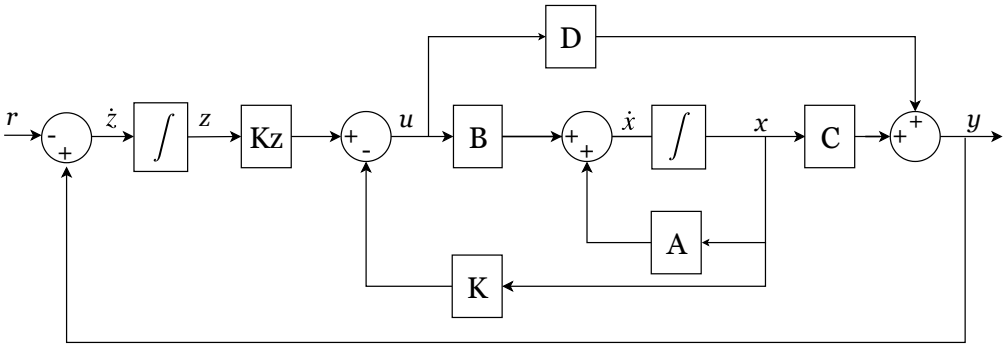


Figure 2.6: Block diagram of the LQI without K_r.

2.6 Discretization of LQR and LQI

In order to run the control loop on a digital system, the controllers need to be converted to discrete time [11] in a finite horizon. The continuous state-space should be represented in its discretized form found in Equation 2.25.

$$\begin{aligned}x(k+1) &= A_d x(k) + B_d u(k)\\ y(k) &= C_d x(k) + D_d u(k)\end{aligned} \tag{2.25}$$

The discrete matrices A_d, B_d, C_d, D_d are calculated using the Equations 2.26, and T_s is the sampling time of the discrete system. It is important to notice that matrix A must be invertible for this to be applicable.

$$\begin{aligned} A_d &= e^{A \cdot T_s} \\ B_d &= e^{A \cdot T_s}(-A^{-1}e^{-A \cdot T_s} + A^{-1}I_n)B \\ C_d &= C \\ D_d &= D \end{aligned} \tag{2.26}$$

The sampling time T_s needs to be chosen so that the system can execute as intended. The sampling rate needs to be at least two times faster then the highest frequency present in the system.
A new cost function is used, although it shows stark similarities to its continuous counterpart, and it is shown in Equation 2.27.

$$J = \frac{1}{2}\sum_{k=1}^{k_f-1}(x^T Q x + u^T R u) \tag{2.27}$$

In discrete time the algebraic Riccati equation found in Equation 2.20 needs to be replaced with the discrete time algebraic Riccati Equation in Equation 2.28. The calculated $\bar{P}$ matrix is then used to calculate the new control law using Equation 2.29. Both the LQR and the LQI use the same equations, the only difference is the added integration states in the LQI, similar as in the continuous time case.

$$A^T P A - P - A^T P B(Q_u + B^T P B)^{-1} B^T P A + Q_x = 0 \tag{2.28}$$

$$K = -(B^T P B + Q_u)^{-1} B^T P A \tag{2.29}$$

2.7 Anti-Windup

One possible issue that can be found in controllers with integral action is integral windup, which happens when the integral action accumulates a significant amount of error beyond the system's saturation limits and overshoots. The problem arises when sudden and large changes happen in the setpoint value. Because of the overshooting, it takes time to revert the error back to zero. This causes the controller to become slow and it will not be able give an appropriate response in time.

To counter integral windup, anti-windup may be used and a few techniques are described in [12]. A common factor between all these methods is that they improve the step response and dampens the overshooting.

One anti-windup is the *conditional integration* (CI) method which, depending on certain conditions, switches on or off the integral. It is common to terminate the integration component when the actuator reaches its saturation limit, and the same sign is found in the control signal and the error [13]. Since the integration is turned

off when the saturation is reached, it will be quicker in reacting to sign changes and therefore give a faster response time. A block diagram of conditional windup can be seen in Figure 2.7.

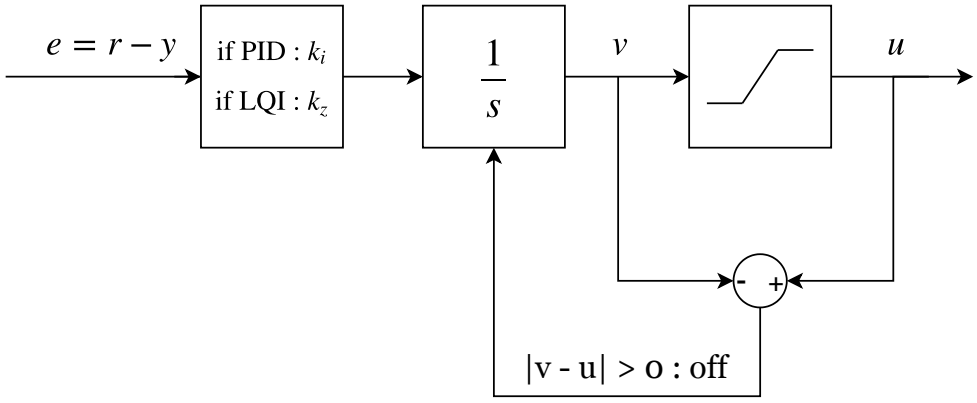


Figure 2.7: Block diagram of conditional windup.

It is worth noting that anti-windup in studies are often used in PIDs. However, LQIs also contain integral action and this project will therefore not only apply anti-windup to the PID but also LQI, although this combination is less common in theory.

A mathematical model for the discrete anti-windup can be found in Equation 2.30. In this model x_k is the current value of the integrator and Δx is the new value that is being added to the integration sum.

$$x_{k+1} = \begin{cases} x_k & \text{if } (x <= \Gamma_{Imin} \text{ and } \Delta x < 0) \text{ or } (x >= \Gamma_{Imax} \text{ and } \Delta x > 0) \\ x_k + \Delta x T_s & \text{otherwise} \end{cases} \tag{2.30}$$

3

System Development

This chapter explains how the project goal is approached and motivates the design choices, based on the methods described in Section 2.

3.1 Linearization of Model

Expressing the equations as a state-space gives an overview of relevant state and system dynamics. With the given states u, v, and r, in addition to the angle, θ, the equations in Section 2.3 are converted into the matrices in Equations 3.1, 3.2, and 3.3 by linearizing around the steady state point X_0= $[0\ u_0\ 0]^T$ and $U_0 = [0\ \tau_0]$. Rudder angle δ and propeller thrust τ are used as inputs. Equations 2.3, 2.4, and 2.5 correspond to f_1, f_2, and f_3 respectively.

$$
\begin{aligned}
A &= \begin{bmatrix} \frac{\partial f_1}{\partial u} & \frac{\partial f_1}{\partial v} & \frac{\partial f_1}{\partial r} & 0 \\ \frac{\partial f_2}{\partial u} & \frac{\partial f_2}{\partial v} & \frac{\partial f_2}{\partial r} & 0 \\ \frac{\partial f_3}{\partial u} & \frac{\partial f_3}{\partial v} & \frac{\partial f_3}{\partial r} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}_{X_0,U_0} \\
&= \begin{bmatrix} 0 & \frac{-2 \cdot C_1 \cdot R_0 \cdot u_0}{m+m_x} & 0 & 0 \\ \frac{Y_v \cdot u_0 \cdot C_1}{m+m_y} & 0 & \frac{u_0(Y_r \cdot L_{pp} \cdot C_1 - (m+m_x))}{m+m_y} & 0 \\ \frac{C_2 \cdot N_v \cdot u_0}{I+J_z} & 0 & \frac{C_2 \cdot N_r \cdot L_{pp} \cdot u_0}{I+J_z} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}
\end{aligned} \tag{3.1}
$$

$$
\begin{aligned}
B &= \begin{bmatrix} \frac{\partial f_1}{\partial \delta} & \frac{\partial f_1}{\partial \tau} \\ \frac{\partial f_2}{\partial \delta} & \frac{\partial f_2}{\partial \tau} \\ \frac{\partial f_3}{\partial \delta} & \frac{\partial f_3}{\partial \tau} \\ 0 & 0 \end{bmatrix}_{X_0,U_0} \\
&= \begin{bmatrix} 0 & \frac{1-t_p}{m+m_x} \\ \frac{u_0^2 \cdot C_3}{m+m_y} & 0 \\ \frac{u_0^2 \cdot C_4}{I+J_z} & 0 \\ 0 & 0 \end{bmatrix}
\end{aligned} \tag{3.2}
$$

$$
C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}
$$

To facilitate the reading of the equations, constants $C_1 - C_4$ have been defined as described in Equations 3.4-3.7.

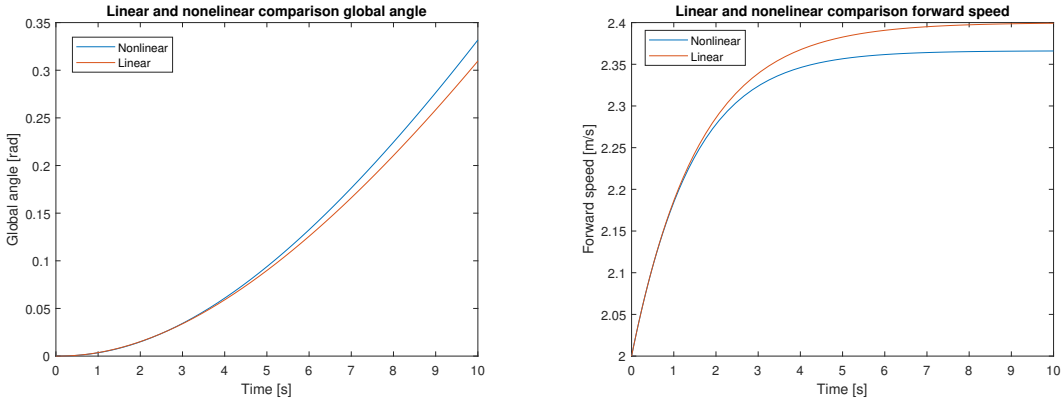
$$C_1 = \frac{1}{2}\rho L_{pp} d \tag{3.4}$$

$$C_2 = \frac{1}{2}\rho L_{pp}^2 d \tag{3.5}$$

$$C_3 = -(1 + A_H)\frac{1}{2}\rho A_R (v_R^2 + u_R^2) f_\alpha \tag{3.6}$$

$$C_4 = -(X_R + A_H + X_H)\frac{1}{2}\rho A_R (v_R^2 + u_R^2) f_\alpha \tag{3.7}$$

This linearized model is needed for development of some of the controllers used in the physical platform. There is a slight difference between the linear and nonlinear models and that is shown in Figure 3.1. As seen the discrepancies are however not too large and thus the linearization is considered acceptable.



(a) Comparison of angle. **(b)** Comparison of speed.

Figure 3.1: Comparison of the resulting angle and speed between the nonlinear and linear models.

3.2 Control System Development

The control systems are developed to generate suitable input values for thrust and rudder angle given the speed and angle of the leader in relation to each follower. For this to be possible it is necessary to use the transformation matrices to convert the leader's state into values that are suitable in the follower's own frame.

3.2.1 PID Design

As mentioned in Section 2.5.1, one PID controller cannot support multiple inputs and multiple outputs. This means that two PIDs are needed in total where one is for

the angle and another one for speed. The parameter values are found heuristically through trial and error and are displayed in Table 3.1.

Table 3.1: Parameters for the PID controllers.

	Thrust	Rudder angle
K_p	1	1
K_i	0.01	1
K_d	0.01	0

Anti-windup has been added to the integral component to prevent overshooting and this decreases the response time.

3.2.2 LQR Design

One of the advantages of the LQR is that the impact of each state variable can easily modified through the weight matrix Q_{lqr}. Similarly the user is also able to change input parameters through the R_{lqr}-matrix. The evaluated matrices Q_{lqr} and R_{lqr} are shown in Equations 3.8 and 3.9.

$$Q_{lqr} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix} \tag{3.8}$$

$$R_{lqr} = \begin{bmatrix} 1000 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.9}$$

These weight matrices have been chosen through trial and error and give a better controller performance than other combinations. The resulting discrete K_{lqr} matrix is as shown in Equation 3.10.

$$K_{lqr} = \begin{bmatrix} -0.0343 & 0 & 0.2708 & 0.3162 \\ 0 & 0.1725 & 0 & 0 \end{bmatrix} \tag{3.10}$$

Since there are four states and two inputs, it will not be possible to assign setpoints to all of them. This means that the matrix K_r cannot be created in a conventional way, and a possible solution would be to use a pseudo inversion method. However, for simplicity reasons, this project has chosen to replace K_r wholly with K_{lqr} instead.

3.2.3 LQI Design

In similar fashion as with the LQR, the LQI has two weight matrices which can be modified to alter the control law. Matrix Q_{lqi} is shown in Equation 3.11 and R_{lqi} in Equation 3.12. The values for matrices Q_{lqi} and R_{lqi} have been chosen heuristically.

$$Q_{lqi} = \begin{bmatrix} 100 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 100 \end{bmatrix} \tag{3.11}$$

$$R_{lqi} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.12}$$

Although there are not any problems with the weight matrices per se, there is a problem with the design of the LQI and there seems to be numerical problems which possibly cause the reachability matrix to lose full rank. To counter this, several methods have been tried such as scaling, preconditioning, and balance realization[14]. Additionally, Newton's method for Riccati calculations [11] was assessed as well as minimal realization[15] but none of these methods mentioned seem to give any improvement and the design remains ill-conditioned.

The LQI does however still work, albeit some values for the weight matrices yield unfeasible solutions due to the lacking reachability. With similar reasoning as with for the LQR, no K_r matrix has been implemented due to incompatible sizes between the states and inputs. Equation 3.13 shows the resulting K_{lqi} matrix in its discretized form acquired with the weight matrices above.

$$K_{lqi} = \begin{bmatrix} -7.0495 & 0 & 17.4333 & 22.2057 \\ 0 & 11.2468 & 0 & 0 \end{bmatrix} \tag{3.13}$$

Equation 3.14 shows the matrix K_{zlqi}.

$$K_{zlqi} = \begin{bmatrix} 0 & 0 & 10 \\ 0 & 10 & 0 \end{bmatrix} \tag{3.14}$$

To prevent overshooting of the angle error, CI anti-windup has been implemented. If the value sent to the rudder angle of the follower goes beyond the saturation limit +25°/-35° the integral component is turned off until the sign of the control error and the setpoint signal change.

3.3 Simulation Environment

In order to test the controller parameters before the boats are operated in water, a simulation environment has been created. This allows the controllers to be tested and to ensure stability before they are implemented on real boats. The environment

is created in MATLAB and Simulink. It contains the nonlinear model described in Section 2.3 together with the linearized system and the three controllers.

The leader's inputs, for thrust and rudder angle, can be defined alongside noise added into the mathematical models of the follower boats to stress the controllers in the simulation. This enables various situations to be evaluated to see how well the follower boats will follow the leader.

3.4 System Architecture

One of the aims in the project is to develop a physical platform and it is therefore crucial that the hardware works for this purpose. This thesis uses RC boats of the model *AquaCraft Lucas Oil FE Brushless Catamaran* and it measures 81.3 cm × 24.2 cm × 15.2 cm for length, width, and height.

Each boat can be controlled by changing its thrust for acceleration and rudder angle for steering. An *AquaCraft 60-Amp Brushless LiPo Ready Marine ESC* controls the motor. The ESC takes a PWM signal of 55 Hz with a duty cycle in the range from 5.8 % to 11.2 %, where the latter one gives max thrust of the motor. The boats are also equipped with a servo that controls the angle of the rudder. This servo takes PWM values from 5.8 % to 11.2 % of duty cycle.

Each boat is also equipped with two LiPo batteries with a total cell count of 4 and a total voltage of 14.8 V. In standard configuration, the boat is controlled with a *TacTic TTX300 remote controller* and a *TacTic TR325 Receiever* that receives the transmitted signals from the remote controller and creates the PWM signal for the Motor ESC and the Rudder servo.

A number of components have been added to control the boat. Raspberry Pi 3 B+ acts as the core and is equipped with a Raspberry Sense Hat that includes an IMU that is essential for acquiring yaw measurements. In order to power the Raspberry Pi a 5 V powerbank has been used.

How the Raspberry Pi is connected on the followers can be seen in Figure 3.2. In this case the Raspberry Pi has the control of both the thrust and the rudder angle. It can read the PWM signals sent from the remote control. The Raspberry Pi in the follower is also equipped with a Raspberry Pi Camera that is mounted on the front of the boat. Since the camera is located on the outside of the boat, it is placed in a zip bag to make it waterproof. A red bucket is placed on the leader for easy camera recognition.

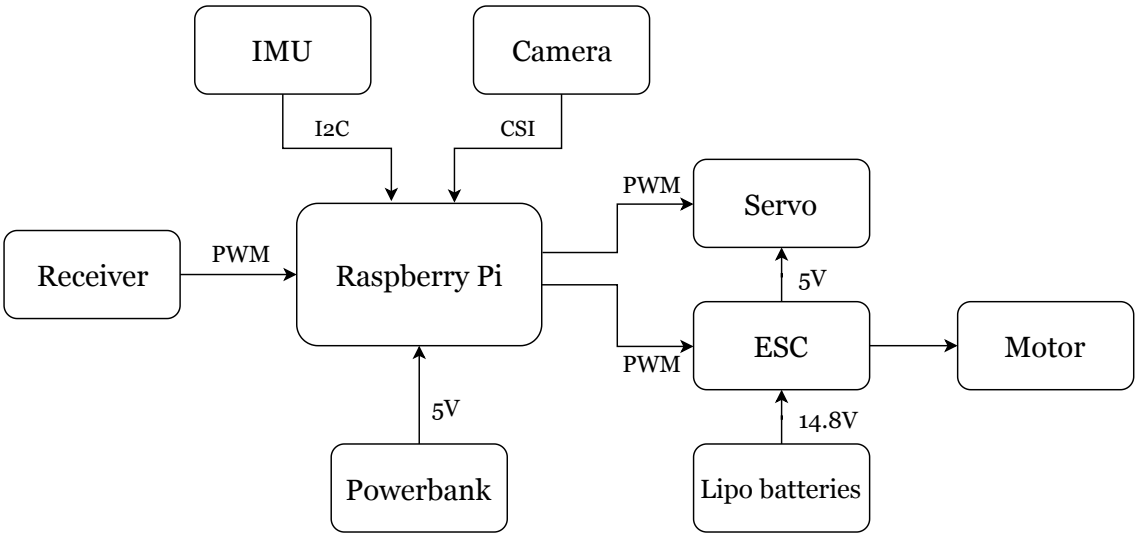


Figure 3.2: Block diagram of system hardware on a follower boat.

The final form of the boats with their modifications can be found in Figure 3.3.

Figure 3.3: RC boats used in the project, with a red bucket placed on the leader and a camera mounted on a follower.

Four things affect the choice of sensors: functionality, size, simplicity, and cost. At first it may seem obvious to use a GPS to get the speed and direction of the boats. However, the main problem with a cheap, small GPS is the accuracy [16]. Since these are small boats that are tested in a small area, such a GPS would not be

able to track the movements of the boats correctly and therefore other alternative sensors are needed.

As another option for obtaining directions of the boats, one could use a Raspberry Pi Sense Hat which is a shield that plugs in directly on the Raspberry Pi. It has and IMU and includes code libraries available to access the data acquired with ease. Similarly, the Raspberry Pi Camera also plugs directly into the Raspberry Pi and is of a small size. Both the shield and the Pi Camera have good compatibility with the microcontroller.

Furthermore, there are several components that need to communicate for the system to fully work. The platform operates in a network of nodes using *Robot Operating System*, ROS. In contrast to what the name suggests, ROS is not an operating system but rather a software framework used to facilitate communication between different components in a system. It is often used to develop robot applications. Tasks are called *nodes* and messages between them are sent through *topics*, virtual channels. Each node can either publish or subscribe to different topics in the network and this is also the backbone of how different components can communicate directly with each other.

ROS requires a network connection to properly work. In this project a mobile Wi-Fi hotspot has been used and all the components are connected to it. Figure 3.4 illustrates how the nodes relate to each other. The three nodes *rudder_sub*, *thrust_sub*, and *controller_node* are active in the follower boat, while *leader_angle*, *computer*, and *user_input* should be hosted on the leader.

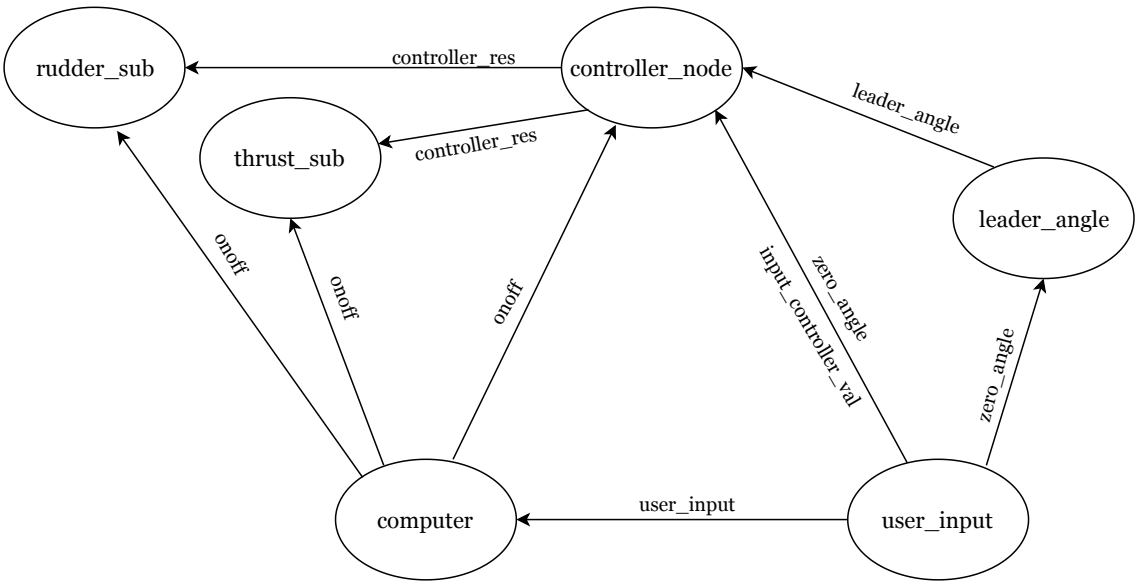


Figure 3.4: Overview of ROS nodes and topics in the system.

As it can be seen there are several topics present, each of them responsible for conveying one type of special message. A summary of what each topic is responsible for can be found in Table 3.2.

Table 3.2: Description of each ROS topic message and type.

Topic	Type	Description
onoff	String	Status of system. Decides active controller or manual mode.
controller_res	Float32MultiArray	Resulting [rudder, thrust] values from controller.
leader_angle	Float32	Leader's yaw angle
zero_angle	String	Resets the angle
input_controller_val	Float32	Edits the values of the K-matrix
user_input	String	User input can decide status or reset angles.

The *computer* node repeatedly publishes status messages and by default the status will enable manual steering of the boats. Through the node *user_input* the user is able to set the status to a controller mode for the follower boats to autonomously navigate. However, if the system for examples disconnects, the nodes listening to the topic will automatically switch back to a manual steering mode.

At first it would seem like *computer* and *user_input* could be the same node, but problems arise since reading messages from the command line the node will be halted until a new line is written. This would stop the continuous flow of the computer if they were the same node. The leader will continuously publish its own yaw angle, while *zero_angle* is only published when the user wishes to.

3.5 Global Angle Estimation

In order to control the heading of the follower, both the leader's angle and a follower's angle need to be estimated. Included in the IMU is a magnetometer. It measures the magnetic field and estimates the orientation of the sensor from these measurements. At first this seems like a good approach for finding the angle, but due to large disturbances coming from the magnetic field of the motor, which is located close to the sensor, the magnetometer becomes unusable. One solution to the magnetic disturbances would be to enclose the motor inside a metal box that stops the magnetic field. This would however add an extra cost and an extra weight to the boat. The extra weight might also change the dynamics of the boat.

The other approach is to use a gyroscope that measures the angular velocities of yaw, pitch, and roll. Only the yaw angular velocity is needed due to the assumption in this thesis that the boats are placed exactly horizontal in a plane. Since the gyroscope does not use magnetic fields, it will not have the same problems as a magnetometer has.

To get the angle from the yaw velocity, the measurements need to be integrated. This gives the potential problem that the noises add up and the angle starts to

drift over time. One solution to this is the optimal linear Kalman filter [17]. After analysing the measurements, the Kalman filter seems unnecessary. This is because the noise is really small in comparison to the real value when the sensor rotates. The final idea is therefore to integrate only when the measurements are larger than a specific threshold. This formula can be seen in Equation 3.15. This idea consists of a constant angle when the measurements is small and an Euler approximation[18] when the magnitude of the measurements is larger than the specific threshold.

$$\theta_{k+1} = \begin{cases} \theta_k & \text{if } |\dot{\theta}| < \Gamma_{\dot{\theta}} \\ \theta_k + \dot{\theta} T_s & \text{otherwise} \end{cases} \tag{3.15}$$

One problem arises with this method. The global angle of the ship is represented in how many radians the boat has turned since it was last reset. This means that if two boats are supposed to compare the angles, they need to have the same heading when reset.

3.6 Speed Estimation

Each follower needs to be able to determine where the leader is in relation to itself. The boat itself is harder to detect partly because of its color and partly because of its shape. A red bucket is placed on the leader in order to counter this problem. This means that the algorithm should instead find an object that is more rectangular in shape and red, something that is seldom found in nature.

To facilitate object detection OpenCV is used, which is an open library commonly used within computer vision. First of all the distance to the object is needed and it is the distances over time that make it possible to compute the speed later on. The camera needs to repeatedly feed new images to the method for it to have access to relevant data.

Algorithm 1 describes the method for estimating the leader's speed in relation to a follower's own speed. Focal length is denoted by `focal_length` and this is a ratio for understanding the camera's vision width and height in relation to real-life measurements. With this, we can for example know how much of a distance one pixel corresponds to. By then knowing large the bucket is in real life, one can then acquire a distance by counting how large it is in pixels.

The height of the bucket in real life is `bucket_height` and variables such as `old_distance` and state are included to be able to fully use this method. The computation of speed will be carried out as long as the the state of the system is not *off*.

Data: state, image, `focal_length`, `bucket_height`, `old_distance`
Result: Finding speed of leader in the follower's frame.
while *state is not off* **do**
1. Find all red parts of the given image;
2. Pick largest red rectangle and assume that this is the bucket;
3. Count pixels of said rectangle, store in variable `rect_pixels`;
4. Compute the distance by `focal_length` × `bucket_height` / `rect_pixels`;
5. Calculate difference between the distance and `old_distance`. Divide this over time between the two images;
6. Store new distance as `old_distance`;

end

Algorithm 1: How to find speed of leader boat in follower's coordinate frame.

To ensure that the whole bucket is visible there is an object ratio. This is computed by taking the width divided by the height and the ratio should be the same for the image capture as in real life. If the ratio differs too much from a reference value it is assumed that the image of the bucket is distorted and an average of the last previous five speed values is used as the new value instead. The distance of the new value is also considered invalid if the difference between the old and new distances is larger than a threshold value, since it might be assumed that the boat cannot travel a certain distance in the given time between when two pictures are taken.

3.7 Testing Methods

A few tests are designed to gather relevant data to evaluate if the system performs satisfactorily. Each controller will be tested for one minute and the path will include a few turns in both directions that are not too sharp. It is also assumed the boats are to be reset in terms of yaw rotation before the controller is activated.

Since the main goal of the follower is to minimize the differences in its trajectory in comparison to the leader, speed and angle are the data of interest. With the gathered data it is then possible to see how the error varies over time and also see an accumulative error for better comparison between the controllers, in case the controllers should give similar results. The controller with the smallest errors will be deduced to function the best out of the three.

Before launching boats into water, the performance of the controllers should be evaluated in simulation and at least one of the controllers should give good results before implementing it on a physical platform. Since the simulation should give indication of how well the controllers work, the simulation time is longer than for the actual tests runs and has been extended to 300 seconds instead.

The physical tests should also be divided into dry and wet tests, with the whole system activated in both cases. A for the dry tests it is interesting to see how the rudder control och motor control behave on land while placed on a stand. The whole

system is activated as if it is in water, but instead of the whole leader boat it is only its Raspberry Pi that is manually manipulated and the follower remains still. Since the boats do not move, it is only possible to see if the controllers work but not to what extent. This method is however good for finding other potential problems that may arise in the physical system such as network problems and akin.

First the quality of the estimations are tested, namely how well the global angle is estimated and how accurate the estimation of speed is. The dry tests are also to see if the rudder is moving in the correct direction in response to turns and how fast it responds. The thrust is tested by moving the microcontroller corresponding to the leader boat and thereafter see how the propeller on the follower adjusts its speed.

4

Results

With the developed platform, it should be possible to launch the boats into water and have them follow the leader. First however, the control systems need to be tested both in simulation and on dry land. This section will therefore be dedicated to the results both from the simulation and physical system.

4.1 Simulation of System

Before implementing the controllers on an actual system, they are tested in a simulation environment. Three boats each represent a different controller - PID, LQR, and LQI - while a fourth one is added as the leader boat. A plot of the simulated trajectory is illustrated in Figure 4.1. The simulation time is set to 300 seconds to ensure stability throughout the whole process and noise is added to stress the controllers.

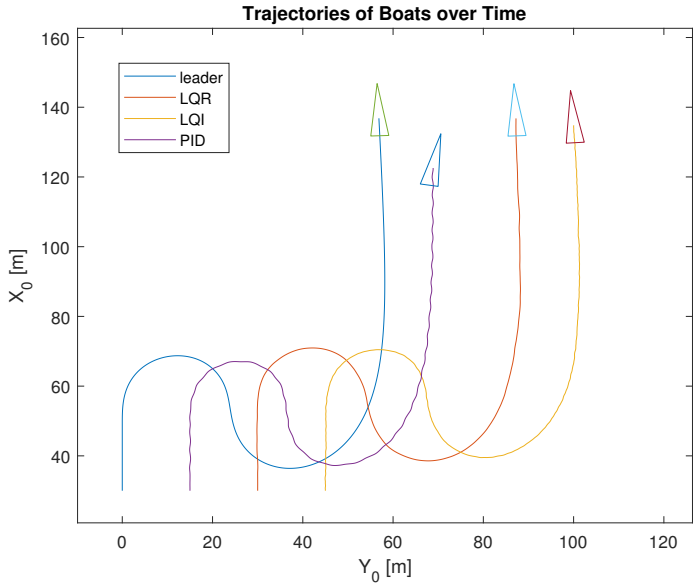


Figure 4.1: Controllers follow the leader trajectory with different success.

If only the trajectory is taken into consideration, it can be seen that it is the LQR that has kept its distance in relation to the leader the best. This is further confirmed

by looking at error over time as well. Figures 4.2, 4.3, and 4.4 show the angle errors for the different controllers.

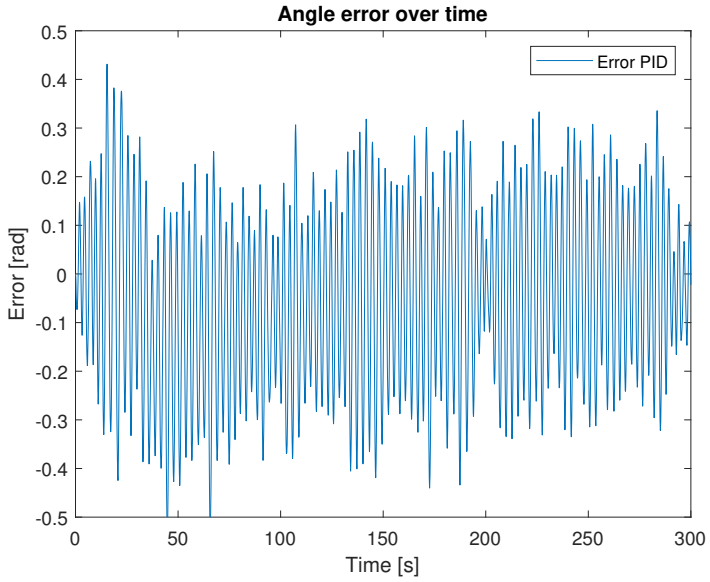


Figure 4.2: Angle error of PID controller in simulation oscillates but stays stable.

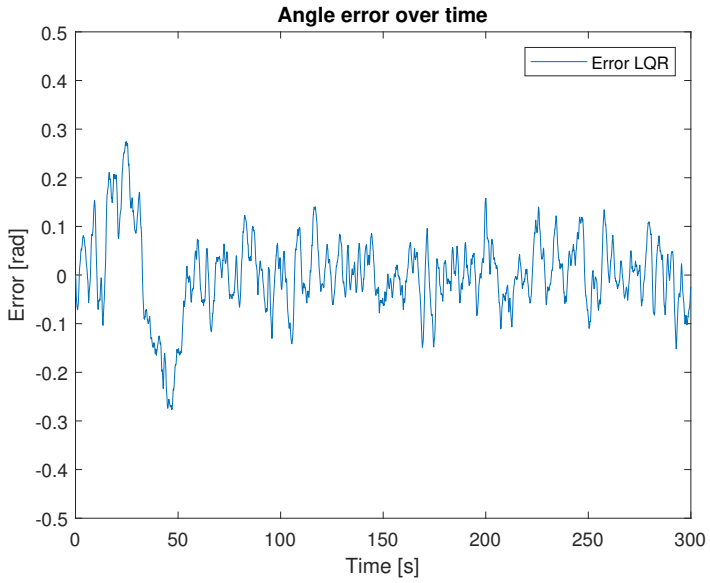


Figure 4.3: Angle error of the LQR has peaks from the turnings in the beginning but remains stable later on.

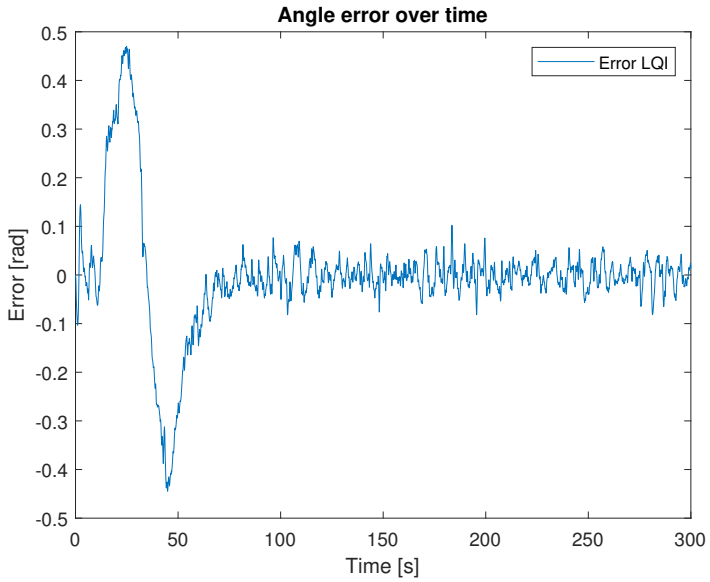


Figure 4.4: When the heading is straight the angle error is smaller for the LQI than the other controllers.

The angle error for the PID shows oscillations throughout the run and the peaks come in the beginning where the boat turns. Even if its peaks are not necessarily the largest, the PID has the highest angle error over time in average. Compared with the other controllers the PID has a high average error over the whole run, but the turnings in the beginning do not impact the difference more than when the path is straight.

The effects of the turns are also more prominent for the LQR and LQI as seen by the peaks before 50 s. Figure 4.4 shows that the peaks of LQI are reaching a maximum error of ± 0.5 rad while the LQR peaks at around ± 0.35 rad. The LQI manages to have the smallest error difference while driving forwards but has a hard time keeping up with turns, seen by the large deviations in the beginning.

Below are error plots as well, but they show the error speed resultant over time instead. The resultant of the forward error speed u and lateral error speed v has been calculated and gives an indication of the displacement. This is a speed that is represented in the follower's own coordinate frame. From Figure 4.5 it is once again seen that the PID varies a lot ranging mostly in between 0.3 m/s to 1.2 m/s.

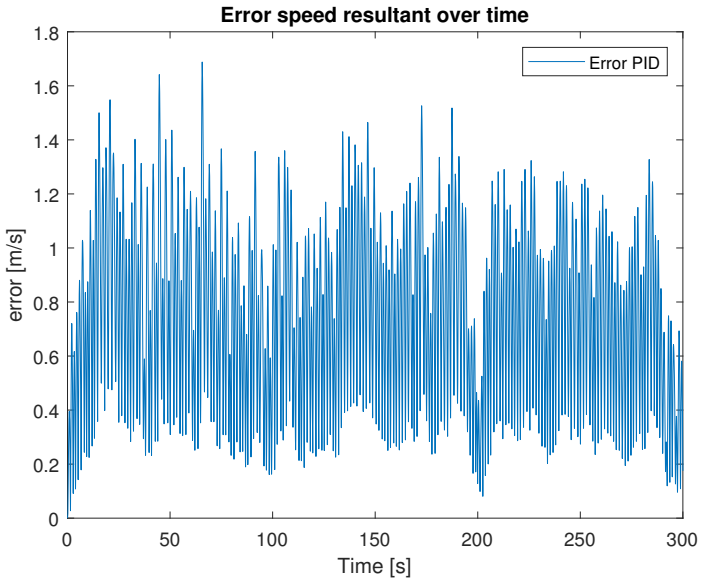


Figure 4.5: Magnitude of PID error speed resultant.

The magnitude of the LQR speed resultant is shown in Figure 4.6. It can be seen that the error peaks are not as high at all and has a max of around 0.6 m/s and there are times when the error is as low as zero. These are results that are more desirable than the PID. Notice once again that the resultant consists of the orthogonal u and v error speeds in the follower's coordinate frame.

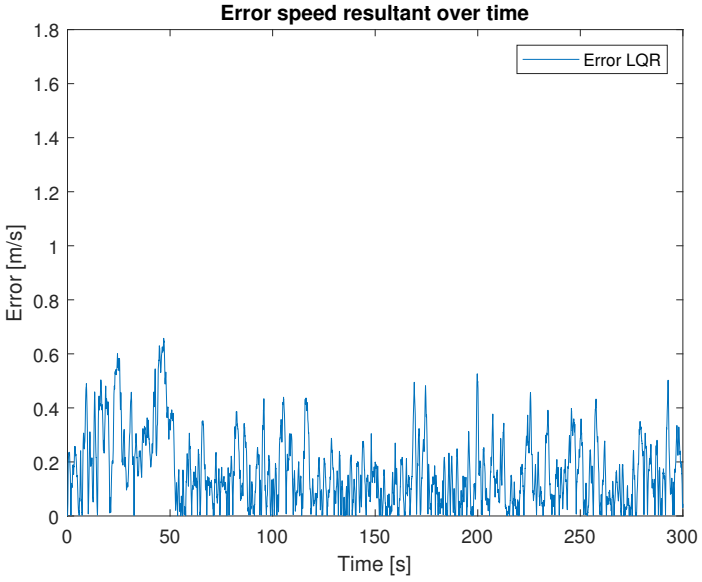


Figure 4.6: Magnitude of LQR error speed resultant. The error never exceeds 0.7 m/s.

Lastly the magnitude plot of the LQI is found in Figure 4.7 and once again the results are better than for the PID. The peaks are higher for this controller than in the LQR, but it manages to go down to zero error at times as well.

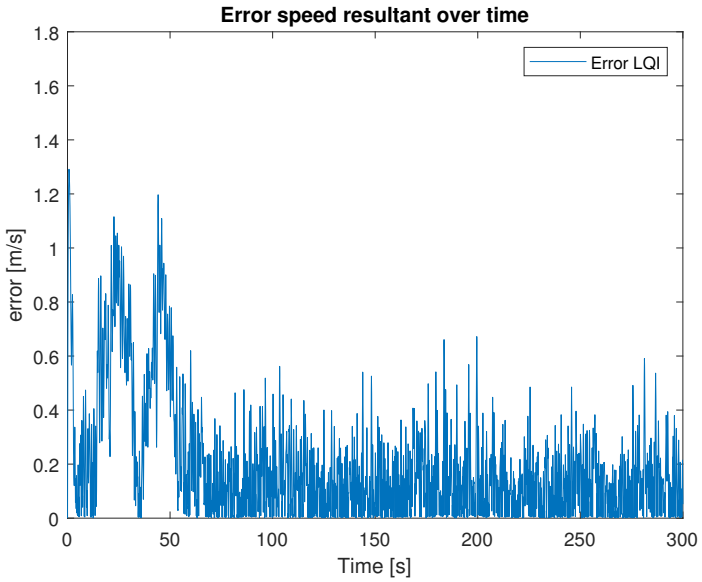


Figure 4.7: Magnitude of LQI error speed resultant. The LQI is more uncertain in the error damping than the LQR but better than the PID.

Accumulative error has been calculated using the same results as in the plots shown above. Table 4.1 describes the accumulative error for each of the controllers for speed and rudder angle. From the table it can be seen that the LQR controller has the lowest accumulated error and is thus the best to track the angle and speed. The LQI is close to the LQR in performance and the PID controller does not give a good result at all in comparison to the other two.

Table 4.1: Accumulative error of angle and speed for the three controllers in simulation.

	PID	LQR	LQI
Angle [rad · s]	96.58	26.69	27.65
Speed [m]	332.45	74.52	97.35

4.2 Experimental Results

Results from simulations reflect an ideal situation and to establish the actual performance of the system, it needs to be tested on the physical boats. The following section describes the functionality of the platform as well as the results from the tests that were executed on the physical system.

4.2.1 Platform

The system has been built to be able to switch between steering from controllers and manual steering. PWM signals that are read by the Raspberry Pi are sent from the remote controller and the user can then decide whether it is the controller

or the remote control that is supposed to steer the boat. The microcontroller also continuously reads status messages from the master node. If no messages arrive on time the mode of the boat changes to manual steering. This means that the boat can be steered also in the case of lost connection to the other devices.

There is additionally an input stream available for the user to change settings in the system. One main function of the user input is to reset the system by setting all angles to 0 rad and another is to switch between which controller to use. It is also possible to change controller parameters when the boats are running in water via this stream.

4.2.2 Dry Tests

Through the dry tests it is noted that the speed estimation does not work as well as it should. Even though the distance estimation returns an acceptable assessment, the variation over consecutive values is too large. This causes a certain unreliability in the speed controller and in turn a spasmodic behavior in the thrust. For this reason, this project has chosen to not implement the thrust controller in the wet tests.

In contrast, the control of the angle works well. It can be seen that the rudder changes in reaction to moving the leader boat albeit the different controllers are not equally as fast. Due to the integral windup both the PID and the LQI are too slow for any acceptable results and will, if tested in water, only go in circles. Some kind of anti-windup is therefore needed. With CI anti-windup the controllers respond to changes faster and it can be seen that they will be able to follow the leader in the wet tests with varying success.

4.2.3 Wet Tests

With the angle controllers passing the simulation and also the dry tests, it is interesting to see how they behave in a wet setting. Two boats are launched in water and each controller is set to be on for one minute each, while the error over time and the accumulated error are recorded. These tests are carried out on the same day and in wind conditions that do not exceed 5 m/s to minimize ambient differences as much as possible.

Figure 4.8 shows the results of the angle error for the PID. The errors here are larger than for the simulation, as expected, but here the controller is unable to keep the error around 0. This might be due to the fact that the integrator part is too slow for this purpose, making the process of correcting the error tardy and thus causing the drift. This happens even though anti-windup has been implemented.

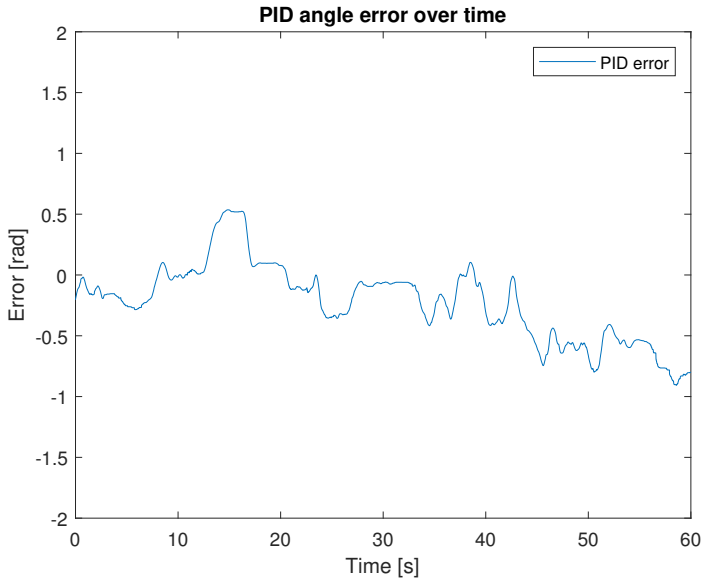


Figure 4.8: Angle error of the PID over time. Here it can be seen that the error drifts away and the controller does not manage to keep it around 0 over time.

In contrast, the LQR behaves more like the its simulation counterpart in terms of that it corrects the angle error and tries to keep it as small as possible. The resulting plot for the angle error of the LQR is found in Figure 4.9. Due to its ability to stabilize even in the wet tests, it is concluded that the LQR is robust.

The controllers are tested for the same amount of time, but the drift found in the PID cannot be found in the LQR. Even though the peaks are a bit larger here than in the simulation, it manages to correct its error and works well for its purpose.

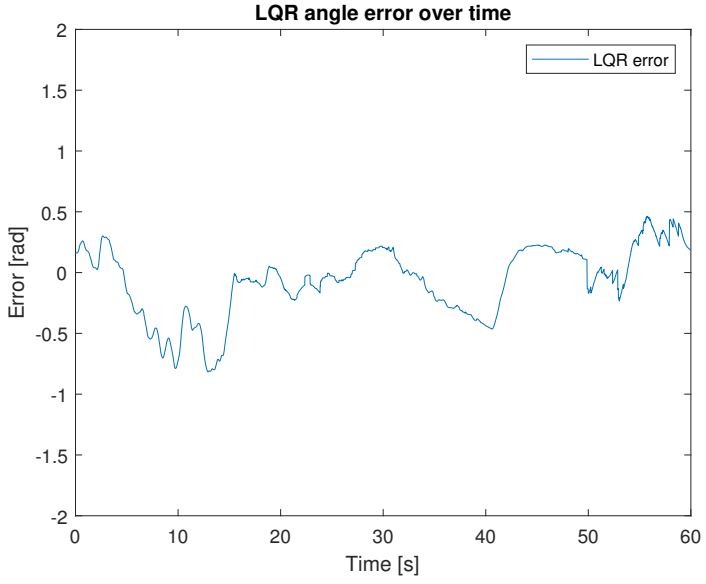


Figure 4.9: LQR corrects the difference between the follower and leader angle.

Angle error over time of the LQI from the wet tests are shown in Figure 4.10. It can be observed that the controller tries to keep the angle error as close to zero as possible, but it is still a bit too slow in the changes and this results in the drift which is seen towards the end of the run. In fact, it seems like the LQI performs the worst out of the three controllers, which is slightly different than the expected results based on the simulation. LQI may have reduced robustness based on these wet tests.

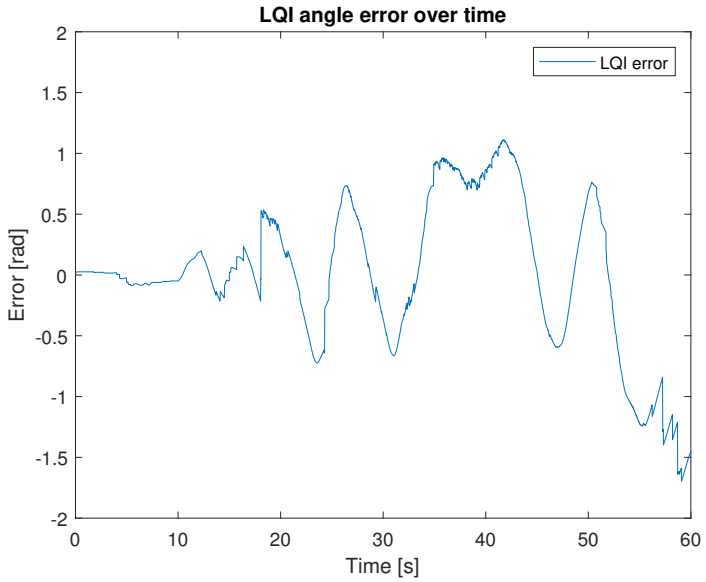


Figure 4.10: LQI tries to correct the difference between the follower and leader angle, but with little success.

The controller performance can further be compared in Table 4.2, which shows the accumulative error of angle for each of the controllers. It is noted that it is the LQR that has the lowest error over the duration of 60 seconds and from this it can be concluded that its performance is the most desired one. In contrast from the accumulated error in the simulation, the results from the wet tests show that the LQI has the highest error and has the hardest time to adjust to the leader's angle. It is worth taking into consideration that the simulation time for the controllers was longer (300 seconds) and the error will therefore be lower for the wet tests.

Table 4.2: Accumulative angle error for the three controllers in wet test.

	PID	LQR	LQI
Angle [rad · s]	17.98	13.73	27.83

5

Discussion

All in all, the controllers perform satisfactorily since they manage to manipulate the boats to follow the leader boat if the speed of them are set constant. There is however also room for improvement and this chapter will discuss and evaluate the selected design aspects of the project as well as suggest improvement and development areas for further work.

5.1 Mathematical Model and Real-Life Differences

There might be several reasons why the experimental results differ from the simulation. Firstly, it is hard to know what kind of disturbances will be present in the water. Currents, winds, and waves may be unaccounted and it is hard to know exactly how much the inertia in water will slow down the movement of the rudder. These have been approximated in the model but may not reflect the real life situation.

In simulation the speed is changed throughout the run and regulated, while the input thrust is kept constant in the wet tests. This might have affected the turnings of the boat since speed is related to how fast the boat maneuvers in both lateral and surge directions.

Another thing to keep in mind is that the model derived for this project is based on models used on larger marine vessels. There might have been problems in the translation into smaller boats, e.g. if some parameters are not linear in the scaling. Since the model itself is abstracted to 3DOF the final model used in the development of the controllers might not be entirely optimal even though it meets the requirements in this project.

Furthermore, the simulation has not been developed to take all potential physical limitations into consideration. The rudder cannot rotate equally as far in both directions, making it harder for the boat to turn right than left. The integral windup was therefore not a problem in simulation but rather something that appeared in the physical implementation.

5.2 Evaluation of Controllers

Mostly the experimental results reach expectations of the simulations, but there is a difference in which controller performed the worst. It can for example be seen from the results that the PID gave a better performance than the LQI which may depend on the fact that it has a derivative action and that could have sped up the controller response.

The PID should also have a slight disadvantage due its structure. It is in reality two controllers - one for rudder angle and one for thrust- and these have no communication between them and this makes these controllers unaware of each other. The unawareness might be a problem since the rotation rate depends on the speed, this might be one of the reasons the PID lacks accuracy in the simulations when compared to the other controllers.

On the other hand the LQR and LQI have been developed specifically for this model and the states are intertwined. They may therefore suit this purpose better than the PID, but they are not entirely free of problems either. Due to the number of states and inputs chosen for this model, K_r has not been implemented for either of the controllers. Additionally, the LQI is ill-conditioned even though it works and it might therefore be a good idea to re-evaluate the state and input matrices for a better structure. The LQR however is quite robust and manages to perform quite well.

One other thing to take into consideration regarding the controllers is also that the weight matrices have been determined heuristically. Because both the tuning and the simulation are both two time-consuming processes, parameters have been selected to give a satisfactorily response. This means that the tuning of them may or may not have been optimal which can affect the performance.

Since both the PID and the LQI have problems with drifting it could be relevant to evaluate if CI is the best anti-windup for this purpose. CI is easy to implement, but perhaps other techniques such as tracking anti-windup or limited integrator could have given similar or even better results than those now.

All three controllers were able to control the thrust input in simulations. Since the speed measurements did not yield a good result in the dry tests it is not possible to say if the controllers could control the speed in comparison to the leader in the wet tests.

5.3 Evaluation of Platform

The platform works well, but since it is dependent on a mobile hotspot, the boats cannot be located too far away from the WiFi source. This limits the size of the

laps and can disrupt the network connection in cases when they are located further away which in turn turns off the controller mode. One solution for this would be to use include WiFi repeaters in the network to expand the range. Using a device with larger range than a cellphone might also be a possible solution.

A feature that does not work quite as well as intended is the estimation of speed. The images captured by the Raspberry Pi camera are slightly blurred since it is placed in a zip bag to protect it from water. Additionally, the colors are dampened and are not as vibrant as without the bag, making it harder for the algorithm to actually detect the defined color range of red. These aspects combined make it hard to estimate the distance to the object since the number of red pixels can vary a lot between consecutive images just depending on lightning, even if the movement is minimal.

One solution of the speed problem could be to test the system on larger boats. Larger boats have access to more information and will be able to use a GPS with higher accuracy which can give more reliable speed data. A GPS would in this case be more useful than for smaller boats, since an inaccuracy matters less for a large vessel than what the same error would do for a smaller one.

The same goes for turning. While it takes time for larger boats to change direction, turning for smaller boats happens more rapidly and thus changes may go unnoticed. This means that there will be more time to correct the turning path for larger boats and this can be advantageous for correcting errors in time.

5.4 Future Work

This project has set up the foundations for a platform used in testing of controllers in marine settings. With this basis it is easy to do further testing with other boats, models or controllers.

Testing other boats would be simple and perhaps a good alternative for evaluating the current controllers. Since the model is scaled down to suit this scenario it could be interesting to see how well the controllers perform with larger boats. There is also the incentive to include more boats to the current platform to ensure that the decentralization is possible for more units in the system as well.

If time allows, it would be even better to try to revise or develop a better model. It has worked well - the developed controllers have been able to control the boats to follow a leader- but there is always room for improvement. Only 3DOF is used but 6DOF, as commonly used in marine settings, can be implemented to make the model more reliable for the price of increased complexity.

It is possible to redefine what formation control means and implement an even stricter formation control. A certain degree of path planning is needed since the followers will have to alter their speed in turns for this. Of course, this depends on a

working method for finding speed in order for the controllers to have all the relevant data needed.

Speed estimation is hard to solve with the current hardware but replacing the camera for another one with higher resolution might help. It might also be wise to discard the current algorithm and replace it with another method. An improvement could be to implement actual object detection instead of just trying to find a rectangle or add object tracking, which could help to deduce if consecutive distances are reasonable or not.

In the case of testing on larger boats, the speed problem might be solved. Larger boats have other and often more complex sensors that yield better measurements, allowing the user to know the driving speed and this information could then be used in the control system.

Further development of the controllers would be to investigate the ill-conditioning of the LQI. This might best be done by re-evaluating the model from the beginning and ensure that it is for example not too simplified. It may also be of interest to test other controllers and their performance. Should path planning be enforced it might for example be relevant to use a model predictive controller or a robust controller, although this might also increase the computational cost.

6
Conclusion

With the project aims in mind it can be concluded that it is possible to develop a platform used in testing different controllers for keeping formations in boats. Although the nonlinear model adapted for this is simplified to only use 3DOF it still provides a successful base for developing simple controllers on. It has been found that all three controllers that have been tested - PID, LQR, and LQI- do all serve the purpose for smaller boats, even if the results vary.

It can be argued how well the applicability of this formation control translates to larger vessels and trough the discussion it would seem like due to the nature of the formulated model as well as the access to relevant data from the GPS, it might even be easier to apply these controllers and methods on larger boats. With this in mind, a decentralized control system for navigating boats is definitely an idea worth exploring further since it might help save time and resources out at real-life sea applications.

Bibliography

[1] J. Shao, G. Xie, J. Yu, and L. Wang, "Leader-following formation control of multiple mobile robots," *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005.*, 2005.

[2] J. Shao, G. Xie, and L. Wang, "Leader-following formation control of multiple mobile vehicles," *IET Control Theory & Applications*, vol. 1, pp. 545–552, 2007.

[3] E. Carvalho, M. P. Silva, and C. Cardeira, "Decentralized formation control of autonomous mobile robots," *2009 35th Annual Conference of IEEE Industrial Electronics*, 2009.

[4] J. P. Desai, J. Ostrowski, and V. Kumar, "Controlling formations of multiple mobile robots," in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, vol. 4, pp. 2864–2869 vol.4, May 1998.

[5] R. Skjetne, S. Moi, and T. I. Fossen, "Nonlinear formation control of marine craft," *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, 2002.

[6] D. J. Stilwell and B. E. Bishop, "Platoons of underwater vehicles," *IEEE Control Systems Magazine*, vol. 20, pp. 45–52, Dec 2000.

[7] I. Necoara, V. Nedelcu, and I. Dumitrache, "Parallel and distributed optimization methods for estimation and control in networks," *Journal of Process Control*, vol. 21, no. 5, pp. 756 – 766, 2011. Special Issue on Hierarchical and Distributed Model Predictive Control.

[8] K. Åström and R. Murray, *Feedback Systems : An Introduction for Scientists and Engineers*. Princeton University Press, Princeton and Oxford, 2008.

[9] C.-Y. Tzeng and J.-F. Chen, "Fundamental properties of linear ship steering dynamic models," *Journal of Marine Science and Technology*, vol. 7, pp. 79–88, 1999.

[10] H. Yasukawa and Y. Yoshimura, "Introduction of mmg standard method for ship maneuvering predictions," *Journal of Marine Science and Technology*, vol. 20, pp. 37–52, Mar 2015.

[11] H. Kwakernaak and R. Sivan, *Linear optimal control systems*, vol. 1. Wiley-interscience New York, 1972.

[12] C. Bohn and D. Atherton, "An analysis package comparing pid anti-windup strategies," *IEEE/IFAC Joint Symposium on CACSD*, pp. 34–40, Apr. 1995.

[13] R. F. Garcia and F. J. P. Castelo, "Conditional integration on pid for types 1 and 2 control systems," in *Proceedings of the International Conference on Control Applications*, vol. 1, pp. 577–581 vol.1, Sep. 2002.

[14] D. F. Enns, "Model reduction with balanced realizations: An error bound and a frequency weighted generalization," in *The 23rd IEEE Conference on Decision and Control*, pp. 127–132, Dec 1984.

[15] R. Williams and D. Lawrence, *Linear State-Space Control Systems*. John Wiley & Sons, 2007.

[16] W. Yan, L. Wang, Y. Jin, and G. Shi, "High accuracy navigation system using gps and ins system integration strategy," in *2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pp. 365–369, June 2016.

[17] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[18] L. Euler, *Institutionum calculi integralis*. No. v. 2 in Institutionum calculi integralis, imp. Acad. imp. Saènt., 1769.