



CHALMERS
UNIVERSITY OF TECHNOLOGY



Autonomous docking of a small electric ferry

Boat maneuvering using model predictive control in parallel with a high-frequency linear quadratic regulator for disturbance rejection

Master's thesis in Systems, Control and Mechatronics

RASMUS BÄCKSTRÖM
MARCUS ODENMARCK

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

MASTER'S THESIS 2026

Autonomous Docking of a Small Electric Ferry

Boat maneuvering using model predictive control in parallel with a high-frequency linear quadratic regulator for disturbance rejection

RASMUS BÄCKSTRÖM
MARCUS ODENMARCK



CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Systems and Control
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026

Autonomous Docking of a Small Electric Ferry
Boat maneuvering using model predictive control in parallel with a high-frequency
linear quadratic regulator for disturbance rejection
RASMUS BÄCKSTRÖM
MARCUS ODENMARCK

© RASMUS BÄCKSTRÖM AND MARCUS ODENMARCK, 2026.

Supervisors: Tobias Husberg, CStrider, and
Torsten Wik, Department of Electrical Engineering
Examiner: Torsten Wik, Department of Electrical Engineering

Master's Thesis 2026
Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
SE 412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Celia One, CStrider's first autonomous ferry prototype.

Typeset in L^AT_EX
Printed by Chalmers Teknologtryck
Gothenburg, Sweden 2026

Autonomous Docking of a Small Electric Ferry
Boat maneuvering using model predictive control in parallel with a high-frequency linear quadratic regulator for disturbance rejection
Rasmus Bäckström
Marcus Odenmarck
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Docking is one of the most challenging and error-prone tasks in marine transportation. This project presents a controller that autonomously docks a small, dual-motor catamaran ferry using exclusively Global Navigation Satellite System (GNSS) and Inertial Measurement Unit (IMU) data. The controller consists of four subsystems: an extended Kalman filter for state estimation; a basic path planner; a Model Predictive Controller (MPC) assisted by a Linear Quadratic Regulator (LQR) for path following; and a thrust allocator that converts desired forces into angle and thrust commands for the motors. Running at 100 Hz, the LQR assists the 5 Hz MPC by attenuating high-frequency disturbances and model errors.

Through system identification and physics modeling, a hydrodynamic maneuvering model is developed to predict the motion of the ferry. The model accounts for water damping, Coriolis and centripetal forces in a rotating body frame, rigid-body acceleration forces, and added mass effects. All development, system identification experiments, and performance evaluation tests are carried out in the Virtual RobotX (VRX) simulator.

The system is evaluated under zero, medium (4 m/s wind, 0.5 m waves), high (7 m/s wind, 1.5 m waves), and high-wind (11 m/s wind, no waves) disturbances, with two runs each, to compare the combined MPC-LQR controller with a standalone MPC. Both architectures successfully dock the ferry under zero and medium disturbances. Under high disturbances, both succeed to dock in one of the runs, while both fail in the other. Under high-wind, the MPC-LQR controller outperforms the standalone MPC by successfully docking in both runs while the standalone MPC only succeeds in one. While both controllers reach the correct position across all disturbance levels, tracking heading remains the primary challenge; both respond slowly to heading changes, though the MPC-LQR provides slightly better tracking. These results demonstrate the feasibility of autonomous docking for small catamaran ferries using an MPC-LQR controller and exclusively onboard sensing under environmental disturbances.

Keywords: Automated Docking, Unmanned Surface Vessel (USV), Hydrodynamical Maneuvering Model, Model Predictive Control (MPC), Linear Quadratic Regulator (LQR), LQR assisted MPC, Extended Kalman Filter.

Acknowledgements

We would like to thank Viktor Lindström, a PhD student in our department, for his support. He shared insights and unpublished details from a similar project of his [11], and helped us find relevant literature and research, and let us borrow an inertial navigation unit (INU) from his project.

We thank the staff of Chalmers Autonomous Systems and Electronics (CASE) Lab for providing access to their facilities. We also extend special thanks to the simulation PC, Linus, for executing excellent simulations.

Finally, we owe CStrider, and especially Tobias Husberg, our gratitude for trusting us with this project, for enabling us to work with their prototype ferry Celia One and for being welcoming into the CStrider team.

Rasmus Bäckström and Marcus Odenmarck, Gothenburg, June 2026

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AI	Artificial Intelligence
COG	Center of Gravity
DES	Desired
DOF	Degrees of Freedom
EKF	Extended Kalman Filter
ENU	East-North-Up
FLU	Forward-Left-Up
GNSS	Global Navigation Satellite System
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
INU	Inertial Navigation Unit
LQR	Linear Quadratic Regulator
MARV	Marine Autonomous Research Vehicle
MIN	Minimize or Minimum
MPC	Model Predictive Control
OSQP	Operator Splitting Quadratic Program
PREV	Previous
QP	Quadratic Programming
ROS 2	Robot Operating System 2
RQt	ROS Qt-Based GUI Framework
SITAW	Situated Awareness
SSRS	Swedish Sea Rescue Society
VRX	Virtual RobotX (Simulation Environment)
WAMV	Wave Adaptive Modular Vessel

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Marine terminology

Surge	Speed in positive x direction, forward
Sway	Speed in positive y direction, to one side (depends on frame)
Heave	Speed in positive z direction, up or down (depends on frame)
Roll	Rotation around x
Pitch	Rotation around y
Yaw	Rotation around z , equivalent to heading
Bow	The front of a boat
Stern	The back of a boat

Variables and Parameters

Use this as a look up table rather than reading it in its entirety.

$\mathbf{A}$	Continuous-time linearized system matrix
$\mathbf{A}_1$	Linear inequality bounds matrix for single-step constraints
$\mathbf{A}_d$	Discrete-time system state matrix
$\mathbf{A}_{ineq}$	Stacked inequality constraint matrix over the horizon
$\mathbf{a}_{ENU}$	Global acceleration vector transformed to the ENU frame
$\mathbf{a}_{FLU}$	Acceleration vector expressed in the body frame
$\varphi_i / \varphi_1, \varphi_2$	Azimuth steering angles for individual thrusters
φ	Actuator matrix array packing thruster steering angles
φ_{cmd}	Absolute target angle output vector command sent to motors
$\Delta\varphi_{max}$	Maximum allowable angular steering change per sample step

$\mathbf{B}$	Continuous-time linearized input mapping matrix
$\mathbf{B}_d$	Discrete-time system input mapping matrix
B	Beam width of the vessel's hull
$\mathbf{b}_1$	Upper bound threshold column vector for single-step constraints
$\mathbf{b}_{ineq}$	Stacked upper bound threshold vector over the horizon
$\mathbf{C}$	Continuous-time linearization constant offset vector
$\mathbf{C}_d$	Discrete-time system constant offset vector
$\mathbf{C}_A(\mathbf{v}_r)$	Hydrodynamic added mass Coriolis and centripetal matrix
$\mathbf{C}_{RB}(\mathbf{v})$	Rigid-body Coriolis and centripetal matrix
$\mathbf{C}(\mathbf{v})$	Total system Coriolis and centripetal matrix
$\mathbf{D}_\Delta$	Difference operator block matrix for control inputs
$\mathbf{D}(\mathbf{v}_r)$	Hydrodynamic damping matrix
d_{draft}	Draft depth of the marine vessel
d_m	Lateral distance spacing layout separating the two thrusters
$\Delta t / dt$	Time step size or sampling interval width
$\Delta \mathbf{U}$	Vector bundling control input rate-of-change
$\mathbf{e}$	Global tracking error vector over the prediction horizon
$\mathbf{e}_0$	Initial tracking error vector over the horizon
E	East position coordinate in the global ENU frame
ε	Small regularization parameter guarding against division-by-zero
$\boldsymbol{\eta}$	Full global state vector tracking pose and body velocities
$\boldsymbol{\eta}_{op}$	Operating point state vector used for linearization
$\boldsymbol{\eta}_{pose}$	Reduced vehicle kinematic state tracking pose matrix vector
$\hat{\boldsymbol{\eta}}_{(1:N)}$	Predicted future state trajectory vector over horizon N
$\mathbf{f}$	Condensed quadratic programming linear objective vector
$\mathbf{f}(\cdot)$	Nonlinear process model function vector
$f(x, y)$	Mathematical function outputting target thrust command
$f_N(\mathbf{v})$	Nonlinear damping torque function around the z-axis
$f_X(\mathbf{v})$	Nonlinear damping force function along the x-axis
$f_Y(\mathbf{v})$	Nonlinear damping force function along the y-axis
$\mathbf{F}$	State transition Jacobian matrix
$\mathbf{F}$	Actuator matrix array packing individual thruster forces
$\mathbf{F}_{\text{max}}$	Physical maximum force bounds constraint vector for thrusters
$\mathbf{F}_{\text{cmd}}$	Absolute target force output vector command sent to motors

$\mathbf{F}_{\text{prev}}$	Actuator thruster force magnitudes from the previous cycle
$F_i / F_1, F_2$	Thrust forces generated individually by outboard motors
$F_{i,\text{cmd}}$	Individual target allocated command force variable for motor i
F_{damping}	Calculated total damping force scalar value
$\mathbf{G}$	Input transition Jacobian matrix
$\mathcal{G}$	Condensed MPC offset transition matrix
g	Acceleration due to gravity
$\mathbf{g}_d$	Discrete constant offset or disturbance vector
$\mathbf{\Gamma}$	Condensed MPC input mapping matrix
$\mathbf{H}$	Measurement Jacobian matrix
$\mathbf{H}$	Condensed quadratic programming Hessian matrix
$\mathbf{h}(\cdot)$	Nonlinear measurement function
$\mathbf{I}$	Identity matrix
I_z	Moment of inertia about the z-axis
J	Non-linear scalar cost function for thrust allocation
J_{lin}	Objective cost function inside linearized thrust allocation
J_{tot}	Total integrated multi-objective MPC cost function
J_{track}	Tracking cost function scalar value
$J_{\Delta U}$	MPC penalty cost term for control input rate-of-change
$\mathbf{K}$	Calculated Kalman gain matrix
K	Kalman gain matrix
L	Total length of the vessel's hull
l	Distance from vessel centerline to individual hull centerline
m	Total physical mass of the vessel
m_h	Mass supported by an individual hull
$\mathbf{M}$	Total system mass and inertia matrix
$\mathbf{M}_A$	Hydrodynamic added mass inertia matrix
$\mathbf{M}_{RB}$	Rigid-body inertia matrix
N	Prediction or control horizon window length
N_{horizon}	Horizon length step count parameter inside the MPC configuration
N	North position coordinate in the global ENU frame
N	Total net propulsion yaw torque
$N_{\dot{r}}$	Added mass rotational moment coefficient in yaw
N_r	Linear hydrodynamic damping coefficient in yaw

N_{max}	Maximum physical limit bound for propulsion torque in yaw
Ω	Weight matrix penalizing rapid thruster steering angle changes
p	Vector containing numerical coefficients for motor force polynomial
p_{lin}	Coefficient of the linear damping polynomial fit
p_1^{quad}	First-order coefficient of the quadratic damping polynomial fit
p_2^{quad}	Second-order coefficient of the quadratic damping polynomial fit
p_1, p_2, p_3, p_4, p_5	Tuned coefficients of the second-order thrust mapping polynomial
$\bar{P}$	Positive weighting scalar penalizing power consumption
$\mathbf{P}$	Matrix weight penalizing force deltas within linearized allocation
$\mathbf{P}_{k k-1}$	Predicted state error covariance matrix
$\mathbf{P}_{k-1 k-1}$	Updated state error covariance matrix from the previous step
$\mathbf{P}_{k k}$	Updated state error covariance matrix
$\mathbf{P}_f$	Weighting matrix for the terminal tracking error
ϕ_1	Azimuth steering angle for thruster 1 alternative notation variant
ϕ	Roll angle of the vessel
Φ	Condensed MPC state transition matrix
$Poly(\cdot)$	Non-linear empirical polynomial converting force into motor commands
ψ	Yaw heading angle representing global orientation
$\mathbf{Q}$	Process noise covariance matrix
$\mathbf{Q}$	Weighting matrix penalizing state deviations
$\bar{\mathbf{Q}}$	Block diagonal state penalty matrix
Q	Weight penalty matrix minimizing thrust allocation slack errors
$\mathbf{R}$	Sensor/measurement noise covariance matrix
$\mathbf{R}$	Weighting matrix penalizing control input magnitude
$\bar{\mathbf{R}}$	Block diagonal control input penalty matrix
$\mathbf{R}_u$	Covariance matrix of input measurements
$\mathbf{r}_{ref}(0 : N)$	Target reference state trajectory sequence
r	Yaw rate
$\hat{r}$	Radius of the approximated submerged hull section
r_{hull}	Physical radius of the catamaran hull
ρ	Mass density of the water
ϱ	Scaling penalty weight factor tracking proximity to singularities
$\mathbf{R}_{E2F}(\psi)$	Rotation matrix mapping ENU frame to FLU frame
$\mathbf{R}_{F2E}(\psi)$	Rotation matrix mapping FLU frame to ENU frame

$\mathbf{p}_{\text{ENU}}$	General vector expressed in the ENU frame
$\mathbf{p}_{\text{FLU}}$	General vector expressed in the FLU frame
$\mathbf{S}$	Innovation covariance matrix
$\mathbf{S}$	Step-wise penalty weight matrix for control rate-of-change
$\bar{\mathbf{S}}$	Block diagonal weight penalty matrix for control rate-of-change
s	Feasibility slack variable column vector
$T(\varphi)$	Geometry-dependent thruster configuration mapping matrix
ThrustCMD	Operational motor metric command output from empirical polynomials
$\boldsymbol{\tau}$	Generalized force and torque vector
$\boldsymbol{\tau}_{\text{rigid body}}$	Generalized force vector from rigid-body dynamics
$\boldsymbol{\tau}_{\text{hydrodynamics}}$	Generalized hydrodynamic force vector
$\boldsymbol{\tau}_{\text{hydro statics}}$	Generalized hydrostatic force vector
$\boldsymbol{\tau}_{\text{motors}}$	Generalized force and torque vector from the motors
$\boldsymbol{\tau}_{\text{wind}}$	Environmental force vector from wind
$\boldsymbol{\tau}_{\text{wave}}$	Environmental force vector from waves
$\boldsymbol{\tau}_{\text{prev}}$	Generalized force control vector from the previous step
$\boldsymbol{\tau}_{\text{max}}$	Maximum generalized control forces and moments boundary vector
$\boldsymbol{\tau}_{\text{des}}$	Combined final target desired generalized force vector
$\boldsymbol{\tau}_{\text{mpc}}$	Generalized force vector computed by the MPC loop
$\boldsymbol{\tau}_{\text{lqr}}$	Dynamic stabilization correction force vector computed by the LQR loop
$\boldsymbol{\tau}_{\text{cmd}}$	Generalized force vector allocated to and realized by actuators
$\boldsymbol{\tau}_{\text{error}}$	Generalized force tracking error vector
$\boldsymbol{\tau}_{\text{op}}$	Operating point control force vector used for linearization
θ	Pitch angle of the vessel
$\mathbf{u}(0 : N - 1)$	Sequence of optimized future control inputs
$\mathbf{u}(k)$	Control input vector applied at look-ahead step k
$\mathbf{u}_{\text{min}}$	Vector setting lower bound constraints on control signals
$\mathbf{u}_{\text{max}}$	Vector setting upper bound constraints on control signals
u	Surge velocity
U_x	Surge damping parameter variant used in results tables
$\mathbf{U}$	Vector bundling sequenced future control inputs
$\mathbf{u}_0$	Control rate-of-change offset vector
$\mathbf{u}_{\text{FLU}}$	Raw input vector containing body-frame measurements

$\mathbf{u}$	Final EKF input vector transformed to the world frame
$\mathbf{u}_k$	EKF system input vector at step k
$\mathbf{v}$	Body-frame velocity vector
$\mathbf{v}_r$	Relative velocity vector
$\mathbf{v}_c$	Water current velocity vector
$\mathbf{V}$	Innovation vector
$V_{i,\text{prop}}$	Estimated water fluid velocity passing through propeller disk i
$V_N^*(x)$	Minimum value of the cost function optimized over horizon N
$V_N(\cdot)$	Multi-step objective cost function
W	Directional weight tracking matrix used inside singularity check
$\mathbf{x}(0 : N)$	Predicted future state trajectory over the horizon
$\mathbf{x}(k + 1)$	Predicted system state vector at the next interval
$\mathbf{x}_{min}$	Vector setting lower bound constraints on system states
$\mathbf{x}_{max}$	Vector setting upper bound constraints on system states
x	Longitudinal coordinate relative to the yaw center
$\mathbf{x}_{k k-1}$	Predicted state estimate vector at time step k
$\mathbf{x}_{k-1 k-1}$	Updated state estimate vector from the previous step
$\mathbf{x}_{k k}$	Updated optimal state estimate vector
$\mathbf{x}_k$	EKF state estimate vector at step k
$\mathbf{x}_{ENU}$	State estimator vector defined in the ENU frame
x_{cog}	Distance to the center of gravity
X	Total net propulsion force along the x-axis
$X_{\dot{u}}$	Added mass force coefficient in surge
X_u	Linear hydrodynamic damping coefficient in surge
X_{max}	Maximum physical limit bound for propulsion force in surge
y	Lateral coordinate relative to the yaw center
Y	Total net propulsion force along the y-axis
$Y_{\dot{v}}$	Added mass force coefficient in sway
$Y_{\dot{r}}$	Cross-coupling added mass force coefficient
Y_v	Linear hydrodynamic damping coefficient in sway
Y_r	Sway damping parameter variant used in results tables
Y_{max}	Maximum physical limit bound for propulsion force in sway
$\mathbf{z}_k$	Actual sensor measurement vector at step k

Contents

List of Acronyms	ix
Nomenclature	xi
List of Figures	xxi
List of Tables	xxv
1 Introduction	1
1.1 Background	1
1.2 Purpose	1
1.3 Scope and Limitations	1
1.4 The use of AI Tools	2
1.5 An Interview With a Deck Officer	2
2 Theory	5
2.1 Previous Work	5
2.1.1 Marine Autonomous Research Vehicle (MARV)	5
2.1.2 Project from Trondheim, Norway	5
2.2 Marine Vessel Modeling	6
2.2.1 Sea-keeping Model	6
2.2.2 Maneuvering Model	7
2.3 Controller	9
2.3.1 Extended Kalman Filter	9
2.3.2 Model Predictive Control (MPC)	10
2.3.3 Robot Operating System 2 (ROS 2)	12
2.4 Virtual RobotX (VRX)	12
3 Method	15
3.1 Hardware	15
3.1.1 Hull and Actuators	15
3.1.2 Sensors	15
3.2 System Identification	16
3.2.1 Marine Vessel Modeling	16
3.2.2 Estimating Added Mass With Strip Theory	17

3.2.3	State Space Modeling	19
3.2.4	Experiment Setup	20
3.2.4.1	Experiment 1	21
3.2.4.2	Experiment 2	21
3.2.4.3	Experiment 3	21
3.2.4.4	Experiment 4	22
3.2.4.5	Experiment 5	23
3.2.4.6	Experiment 6	23
3.2.5	Parameter Estimation	23
3.2.5.1	Estimating the Dampening Parameters	24
3.2.5.2	Estimating the Force Produced by the Motors	24
3.3	Controller	25
3.3.1	Global Frame and Body Frame	26
3.3.2	State-Estimator	27
3.3.3	Path-Planner	28
3.3.4	MPC	29
3.3.5	Condensed form MPC	29
3.3.6	OSQP Solver	32
3.3.7	LQR	32
3.3.8	Thrust-Allocator	33
3.4	Simulation in VRX	37
3.5	Docking Tests in VRX	37
4	Results and Discussion	39
4.1	System Identification	39
4.1.1	Estimating the Dampening Parameters	39
4.1.1.1	Comments on the Inaccuracy of the Estimations	39
4.1.1.2	Comments on the Oscillations in the Data	40
4.1.2	Estimating the Force Produced by the Motors	42
4.2	Subsystems	45
4.2.1	State-Estimator	46
4.2.2	Path-Planner	46
4.2.3	LQR	48
4.3	Docking	49
4.3.1	Sequence 1: Scenario 1	49
4.3.2	Sequence 1: Scenario 2	54
4.3.3	Sequence 1: Scenario 3	58
4.3.4	Sequence 1: Scenario 4	61
4.3.5	Sequence 2: Scenario 1	64
4.3.6	Sequence 2: Scenario 2	67
4.3.7	Sequence 2: Scenario 3	70
4.3.8	Sequence 2: Scenario 4	73
4.3.9	Summary of Docking Results	78
5	Conclusions and Future Work	79
5.1	Future Work	79
5.1.1	Within This Project	79

5.1.1.1	Modeling	80
5.1.1.2	Controller	80
5.1.2	Not Within the Project	81
Bibliography		83

List of Figures

2.1	Visual illustration of generalized forces (left) and the state vector (right).	7
2.2	MPC illustrated	11
2.3	Screenshot from VRX.	13
3.1	Image of Celia One from	16
3.2	Cross section of submerged hull in gray, the approximated submerged hull in orange, different dimensions illustrated with green arrows and the waterline illustrated in blue.	18
3.3	How to drive a catamaran sideways using a dual motor setup. Point both motors outwards approximately 25 degrees so that their thrust vector intersects the axis of rotation in yaw. Then reverse one motor and drive the other one forward. This will result in a combined force acting in sway.	22
3.4	Experiment 4. This is an example of how the thrust command and surge can look when following the sequence explained for Experiment 4.	23
3.5	Controller architecture. r_{pp} is the reference path from the Path-Planner, τ_{mpc} and τ_{lqr} are the generalized forces calculated by the MPC and LQR respectively, τ_{des} is the combined desired generalized control force, ThrustCMD is the motor control forces and angles, and x is the state vector. In this figure, x represents both x and η . η is a state vector in another frame explained bellow.	26
3.6	The ferry in the FLU body frame relative to the ENU global frame. The heading angle is measured relative to the east axis of the global frame.	27
3.7	Conceptual illustration of the function of the Thrust-Allocator	34
3.8	Geometric setup of the thrusters on a dual azimuth thruster catamaran, as Celia One.	34
4.1	Raw surge and surge acceleration data from six runs of Experiment 1.	41
4.2	Raw sway and sway acceleration data from four runs of Experiment 2.	41
4.3	Raw yaw rate and calculated yaw acceleration data from four runs of Experiment 3. The data series in this plot has been shifted in time for better visualization.	41
4.4	Different fits to force-surge data from Experiment 1	42

4.5	Different fits to force-sway data from Experiment 2	42
4.6	Different fits to moment-yaw rate data from Experiment 3	42
4.7	Raw data from one of the four runs of Experiment 4.	43
4.8	Raw data from one of the four runs of Experiment 4, where more steps were taken than was initially planned when the experiment was invented.	44
4.9	Raw data from one of the four runs of Experiment 4, when the ferry was driving backwards instead.	44
4.10	The black points are estimated from the Experiment 4 data. The surface is the function estimated from the data. Lastly, the red points are points taken at places in the data with zero acceleration as validation points. View 1 show the graph with the origin in the foreground and the surface sloping upwards as it moves away. View 2 instead, has the origin in the background and the surface is seen decreasing down toward zero as it approaches the origin. View 2 also indicates that the surface is a good estimation of the points since it follows them quite well.	45
4.11	The estimated states versus ground truth value during docking Sequence 1, Scenario 1. The initial position pose of the ferry is $\boldsymbol{\eta}_{pose} = [39.2, -38.3, 0.99]$	46
4.12	The planned path in Scenario 1 with each planned state plotted separately.	47
4.13	The planned path in Scenario 1 from a top view.	48
4.14	Figure 4.14a shows a step performance carried out on the ferry in VRX. The EKF estimated states are the blue lines. The dashed red lines denote the reference. Figure 4.14b illustrates the forces that are delivered by the motors via the Thrust-allocator.	48
4.15	Bird's-eye view of docking Sequence 1: Scenario 1. The ferry trajectory (blue) is the estimated path traversed by the ferry. The planned trajectory, denoted by the dashed red line, is the Path-Planner reference. The ground truth is the true position according to the VRX environment. The twist point (green) is where the ferry rotates into position to reverse into the dock.	50
4.16	A comparison of the two different controllers tracking the reference during the docking sequence. The state estimate (blue) is the EKF filtered measurement. The red line is the Path-Planner reference. The horizontal axis denotes time.	51
4.17	A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize. The blue lines denote $\boldsymbol{\tau}_{cmd}$ and the red ones denote $\boldsymbol{\tau}_{des}$	52
4.18	Thrust and rudder angle commands based on the allocated forces and moments. The rudder angles and motor forces are computed directly by the Thrust-Allocator from Chapter 3.3.8. The thrust commands applied, F_i , are computed from the motor forces using Equation 3.30.	52
4.19	Illustration of the ratio of controller contribution in $\boldsymbol{\tau}_{des}$ in the case where both controllers are used.	53

4.20	Bird's-eye view of docking (Sequence 1, Scenario 2).	55
4.21	A comparison of the two different controllers tracking the reference during the docking sequence.	56
4.22	A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.	57
4.23	Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.	57
4.24	Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.	57
4.25	Bird's-eye view of docking (Sequence 1, Scenario 3).	58
4.26	A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 1, Scenario 3).	59
4.27	A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.	60
4.28	Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.	60
4.29	Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.	60
4.30	Bird's-eye view of docking (Sequence 1, Scenario 4).	61
4.31	A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 1, Scenario 4).	62
4.32	A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.	63
4.33	Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.	63
4.34	Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.	63
4.35	Bird's-eye view of docking from the second start position (Sequence 2: Scenario 1).	64
4.36	A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 2: Scenario 1).	65
4.37	A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.	66
4.38	Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.	66
4.39	Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.	66
4.40	Bird's-eye view of docking (Sequence 2: Scenario 2).	67
4.41	A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 2: Scenario 2).	68
4.42	A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.	69
4.43	Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.	69
4.44	Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.	69

4.45	Bird's-eye view of docking (Sequence 2: Scenario 3).	70
4.46	A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 2: Scenario 3).	71
4.47	A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.	72
4.48	Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.	72
4.49	Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.	72
4.50	Bird's-eye view of docking (Sequence 2, Scenario 4).	74
4.51	A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 2, Scenario 4).	75
4.52	A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.	76
4.53	Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.	76
4.54	Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.	76

List of Tables

3.1	Differing initial positions for the ferry docking sequence	38
3.2	Environmental Parameters by Scenario	38
4.1	Estimated parameters for the dampening in the hydrodynamical model along with the ground truth parameters used by VRX	40
4.2	Docking Performance of the MPC controller for Sequence 1.	78
4.3	Docking Performance of the MPC-LQR controller for Sequence 1.	78
4.4	Docking Performance of the MPC controller for Sequence 2.	78
4.5	Docking Performance of the MPC-LQR controller for Sequence 2.	78

1

Introduction

1.1 Background

Electric marine vessels can offer significant environmental benefits, compared to vessels powered by internal combustion engines, by reducing emissions and air pollution. CStrider proposes an innovative, sustainable option for traversing urban waterways using electric ferries [2]. This solution provides an alternative transit method while reducing traffic congestion along main routes. Because the CStrider vessel is a small commercial ferry, minimizing onboard personnel is paramount for economic sustainability. Concurrently, the rise of automation throughout society makes autonomous maneuvering an excellent option to reduce operational expenses. However, these autonomous solutions must maintain high quality without compromising safety or drastically increasing energy consumption.

This project addresses the problem of autonomous docking. Human error accounts for approximately 70 to 80% of maritime accidents [3]. Ferry docking is widely perceived as a tedious and accident prone part of marine travel. Turbulent wind and currents cause vessel drift, which complicates navigation, particularly at the low speeds of docking procedures. Thus, the autonomous solution requires meticulous trajectory planning to achieve the desired final pose while maximizing energy efficiency. To save energy, propulsion must be minimized, which underscores the importance of planning both an effective geometric path and a sequence of actuator inputs to guide the ferry from its starting position to the dock.

1.2 Purpose

The purpose of this project is to develop a proof-of-concept controller capable of autonomously docking a small, dual-motor ferry. The controller executes a stern-first docking maneuver, guiding the vessel from a distance of 30–40 m to the berth.

1.3 Scope and Limitations

- As a proof of concept, many aspects of the controller are suboptimal and require further development to be fully functional.
- Although designed with energy efficiency in mind, it is not prioritized due to

time constraints.

- Environmental forces from wind, waves and currents are only handled through feedback, and not feed forward through disturbance models.
- Systems for detecting and avoiding obstacles or other vessels during docking are not considered.
- Variations in vessel mass and inertia due to passenger count are ignored, and passenger weight is assumed to be negligible.
- All tests and experiments are conducted in a digital simulation environment; physical implementation is omitted due to time constraints. However, the system is developed with future real-world implementation in mind.

1.4 The use of AI Tools

Throughout this project, development follows specific guidelines regarding the utilization of AI tools. These boundaries are established based on personal experience and Chalmers' institutional regulations for AI usage in thesis defense [14]:

- It does not serve as direct informational sources in the report.
- It can serve as a development assistant for tasks where answers are easily verified or where the risk of trusting an incorrect response is negligible.
- It can assist with basic software engineering tasks, such as generating plots, writing simple scripts, and debugging.
- It can act as educational aid when learning new simulation frameworks, toolboxes, or software libraries.
- It is used to proof read and refine the writing in the report.

1.5 An Interview With a Deck Officer

To get an insight into what people from the marine industry think of unmanned and autonomous vessels, Matilda Hallman, who also is a friend of Marcus Odenmarck, was interviewed on the topic.

Hallman recently graduated as a class VII deck officer and was interviewed on her opinion about unmanned and autonomous vessels reaching her field and the consequences, good and bad, that she is able to identify [15]. With her approval, her answers were translated into english and slightly reformulated. She has read through and approved the final versions of the answers.

How interested are you in working on board a ferry doing short routes on channels

and rivers in an urban environment?

It's not my dream job, no. However, I see that type of work as a perfect way into the maritime industry. It's a relatively straightforward job with normal working hours where you, unlike many other jobs at sea, get to go home in the evening. There are a lot of routines and passenger service, which is a great start for a recent graduate. You are often no more than two crew members on board, which implies a lot of responsibility. It might look like you're just standing there staring at the passengers, but once you get an insight into a workday, it suddenly becomes much more interesting. Engine checks, docking, departure, safety, and in poor weather, you're on the bridge as a lookout.

What are your feelings towards the fact that automation is reaching your profession?

There are two perspectives there. Firstly, I don't feel particularly threatened when it comes to losing my job. Shipping accounts for an incredibly large part of the world economy, and skippers will always be needed. There is already a labor shortage within the maritime industry and more automation could therefore help. On the other hand, I am quite skeptical about replacing a seafarer with a robot or software. Sure, a large percentage of accidents at sea occur due to human error, but what percentage of accidents have been avoided because of humans?

What advantages do you see in semi and/or fully autonomous vessels?

I definitively see an economic advantages. This is because it will take less training to educate new people and fewer staff are needed on a autonomous vessel. This also goes back to what I said before about the possible solution to the labor shortage. Automation is also a useful tool, specifically on long voyages, cargo ships across the Atlantic and such. Then you can spend more time on maintenance instead of navigation.

What disadvantages do you see in semi and/or fully autonomous vessels?

On short trips, like urban traffic, there are many external factors like weather, currents, winds, and other traffic like sailboats, kayaks, and other ferries, among others, that is hard for a computer to always correctly compensate for. Perhaps cable ferries are another viable option; they travel on a predetermined route and are easy to dock, etc.

Whenever you work with people, problems occur. The autonomous ferry has to handle tourists, bicycles, and other 'junk' people bring along, but how does it do that? An example is alcohol; many evenings, there are people who want to come aboard and drink their alcohol, but you as a deckhand have to stop them as there is zero tolerance for that on the ferries I work on. This is due to the safety risks and out of respect for other passengers.

Then there is also the question of responsibility. If there is no one on board who is in charge, who is responsible if someone falls overboard or get crushed during

1. Introduction

docking? Accidents do happen and there needs to be someone there to handle it. There is a reason why the education to become an officer is several years long; it is not easy to replace.

2

Theory

This chapter highlights the theories, concepts, and software used in this project. It provides an overview of how these tools function rather than detailing their specific implementation. It also explores similar previous work.

2.1 Previous Work

This section highlights examples of previous work conducted in the same field.

2.1.1 Marine Autonomous Research Vehicle (MARV)

In 2021, the Department of Electrical Engineering at Chalmers hosted a project in collaboration with the Swedish Sea Rescue Society (SSRS) [11]. The project aimed to create a testbed that simplified the implementation and testing of autonomous navigation software for students and researchers, while demonstrating how autonomous systems could assist SSRS operations. Both hardware and software were developed to facilitate the integration of high-level control software and algorithms into MARV. To verify the testbed, a demonstrational waypoint mission control system was developed and implemented.

As part of the demonstrational control system, a dynamical maneuvering model was created to describe how the hull interacts with the water. Several experiments were designed and conducted to estimate the required model parameters. One such test, a coast-down test, approximated the damping force generated by the water as a function of surge.

2.1.2 Project from Trondheim, Norway

Sensor fusion algorithms combined with a control system showed promising results for autonomous docking. In 2021, the fully autonomous urban electric ferry, milliAmpere2, was launched [4]. Developed by the Norwegian University of Science and Technology in Trondheim, milliAmpere2 was designed to carry up to 12 passengers, matching the capacity of the CStrider ferry.

To navigate busy waterways safely, the ferry utilized a diverse sensor suite for docking and obstacle avoidance. A situational awareness computer integrated twelve

cameras (eight RGB and four IR), two lidars, one radar, and four ultrasonic distance sensors to estimate the position and velocity of nearby objects. This data enabled the ferry to reactively adjust its path and velocity, which ensured safety in high-traffic zones. Additionally, primary and backup global navigational satellite system (GNSS) units, an inertial measurement system (IMU), and a wind sensor provided the autonomy system with essential state and environmental information. Whenever sensor performance declined, the environmental assessment became inaccurate. To mitigate this, the milliAmpere2 utilized a safety mechanism that evaluated measurement quality; if the data proved inadequate, the ferry entered a minimum-risk state and halted.

The ultrasonic sensors guided the ferry during the final stages of docking. These sensors were mounted in pairs on both the bow and the stern. Because the ferry was symmetric, it docked bidirectionally, eliminating the need for substantial rotation around the yaw axis to achieve a correct heading prior to docking. This design simplified the control problem by reducing deviations from the desired final heading angle. Consequently, the ferry tracked the path planner and utilized ultrasound measurements exclusively to fine-tune its final position relative to the docking station.

In contrast, the CStrider ferry is designed with a large cabin for passenger comfort, and its docking mechanism is located solely on the stern. This layout requires rotation around the yaw axis along its path, introducing complexities for the control and path-planning algorithms, which has to account for these maneuvers to achieve the desired final pose. Furthermore, the docking sequence this project develops for CStrider relies entirely on onboard sensors, reducing infrastructure requirements and improving versatility. However, docking without external guidance requires highly accurate position and orientation estimates derived solely from onboard instrumentation.

2.2 Marine Vessel Modeling

Marine vessel theory comprises two main branches: sea-keeping and maneuvering theory [17]. They differ fundamentally in their physical derivations due to distinct operating conditions.

2.2.1 Sea-keeping Model

Sea-keeping models predicts vessel behavior under normal operating conditions out at sea involving wave and current disturbances. In these models, velocity and heading is assumed to be constant and hydrodynamic coefficients are dynamic functions of wave frequency [17, p. 81].

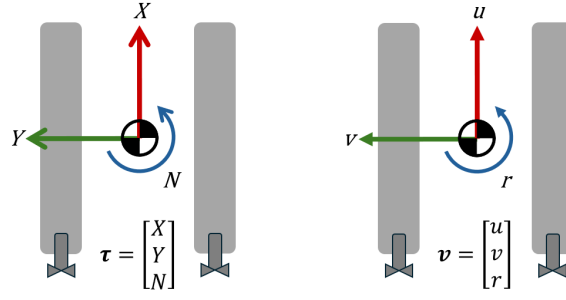


Figure 2.1: Visual illustration of generalized forces (left) and the state vector (right).

2.2.2 Maneuvering Model

Maneuvering models assume operation in calm waters without high-frequency wave disturbances. This assumption yields static hydrodynamic coefficients [17, p. 109]. This project uses a maneuvering model because docking occurs at low velocities and normally in calm waters. The model is defined in the horizontal plane; therefore, pitch, roll, and heave are neglected. The state vector is defined as the body-frame velocity vector,

$$\mathbf{v} = [u, v, r]^\top, \quad (2.1)$$

where u is surge, v is sway, and r is yaw rate in the body frame (shown in Figure 2.1).

The 3-DoF maneuvering model is expressed as a sum of several terms:

$$\tau_{\text{rigid body}} + \tau_{\text{hydrodynamics}} + \tau_{\text{hydro statics}} = \tau_{\text{motors}} + \tau_{\text{wind}} + \tau_{\text{wave}}, \quad (2.2)$$

where $\tau = [X, Y, N]^\top$ is a generalized force vector (shown in Figure 2.1), in which X and Y are the forces along the x - and y -axes, and N is the yaw torque. The six different τ terms are described below.

The rigid-body forces, $\tau_{\text{rigid body}}$, arise from the motion of a body with mass and inertia. They are defined as

$$\tau_{\text{rigid body}} = \mathbf{M}_{RB} \dot{\mathbf{v}} + \mathbf{C}_{RB}(\mathbf{v})\mathbf{v} \quad (2.3)$$

The rigid-body inertia matrix $\mathbf{M}_{RB}$ represents the body's inertia, and the rigid-body Coriolis and centripetal matrix $\mathbf{C}_{RB}$ represents the Coriolis and centripetal effects in the rotating body coordinate system. They are defined as

$$\mathbf{M}_{RB} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & mx_{\text{cog}} \\ 0 & mx_{\text{cog}} & I_z \end{bmatrix} \quad (2.4)$$

and

$$\mathbf{C}_{RB}(\mathbf{v}) = \begin{bmatrix} 0 & 0 & -m(x_{\text{cog}}r + v) \\ 0 & 0 & mu \\ m(x_{\text{cog}}r + v) & -mu & 0 \end{bmatrix}$$

where m is the mass, I_z is the moment of inertia about the z -axis, and x_{cog} is the distance between the rotation axis and the center of gravity. Note that $\mathbf{C}_{RB}(\mathbf{v})$ depends on the state vector.

The hydrodynamic forces, $\boldsymbol{\tau}_{\text{hydrodynamics}}$, consist of added mass forces and hydrodynamic damping forces

$$\boldsymbol{\tau}_{\text{hydrodynamics}} = \underbrace{\mathbf{M}_A \dot{\mathbf{v}}_r + \mathbf{C}_A(\mathbf{v}_r) \mathbf{v}_r}_{\text{added mass forces}} + \underbrace{\mathbf{D}(\mathbf{v}_r) \mathbf{v}_r}_{\text{damping forces}} \quad (2.5)$$

where $\mathbf{v}_r = \mathbf{v} - \mathbf{v}_c$ and $\mathbf{v}_c$ is the velocity of the current. Under ideal conditions, the velocity of the current is zero, such that $\mathbf{v}_r = \mathbf{v}$. The added mass phenomenon represents pressure-induced forces and moments caused by the movement of the hull in the water [16, p. 32]. The added mass inertia matrix, $\mathbf{M}_A$, and the added mass Coriolis and centripetal matrix, $\mathbf{C}_A$, are given by:

$$\mathbf{M}_A = \begin{bmatrix} -X_{\dot{u}} & 0 & 0 \\ 0 & -Y_{\dot{v}} & -Y_{\dot{r}} \\ 0 & -Y_{\dot{r}} & -N_{\dot{r}} \end{bmatrix}$$

and

$$\mathbf{C}_A(\mathbf{v}) = \begin{bmatrix} 0 & 0 & Y_{\dot{v}}v + Y_{\dot{r}}r \\ 0 & 0 & -X_{\dot{u}}u \\ -Y_{\dot{v}}v - Y_{\dot{r}}r & X_{\dot{u}}u & 0 \end{bmatrix}.$$

The entries in $\mathbf{M}_A$ and $\mathbf{C}_A(\mathbf{v})$, when multiplied by the state vector or its derivative (see Equation 2.5), represent the pressure-induced forces and moments caused by hull acceleration in surge, sway, and yaw. These parameters are often estimated using strip theory (see Section 3.2.2).

The damping force is defined as the damping matrix $\mathbf{D}(\mathbf{v}_r)$ multiplied by the state vector $\mathbf{v}_r$ (see Equation 2.5), where the damping matrix is defined as

$$\mathbf{D}(\mathbf{v}) = [f_X(\mathbf{v}), f_Y(\mathbf{v}), f_N(\mathbf{v})]^\top, \quad (2.6)$$

where each function f_i is nonlinear and is typically determined experimentally and approximated as a linear or polynomial function.

Under the assumption of zero current, $\mathbf{v}_c = \mathbf{0}$, the hydrostatic force in the 3-DoF horizontal plane is zero [17, p. 133]:

$$\boldsymbol{\tau}_{\text{hydrostatics}} = \mathbf{0}. \quad (2.7)$$

The project scope excludes environmental disturbances; therefore, wind and wave forces are neglected:

$$\boldsymbol{\tau}_{\text{wind}} = \boldsymbol{\tau}_{\text{wave}} = \mathbf{0}. \quad (2.8)$$

The control forces and torque generated by the motors are represented as

$$\boldsymbol{\tau}_{\text{motors}} = [X, Y, N]^\top. \quad (2.9)$$

From Equations 2.3–2.9 (excluding Equation 2.6), the model is expressed as

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}(\mathbf{v})\mathbf{v} + \mathbf{D}(\mathbf{v})\mathbf{v} = \boldsymbol{\tau}_{\text{motors}}, \quad (2.10)$$

where

$$\begin{aligned} \mathbf{M} &= \mathbf{M}_{RB} + \mathbf{M}_A \\ \mathbf{C}(\mathbf{v}) &= \mathbf{C}_{RB}(\mathbf{v}) + \mathbf{C}_A(\mathbf{v}). \end{aligned}$$

2.3 Controller

Autonomous navigation controllers are typically divided into subcomponents. This section describes the algorithms and control theory used across these components. Their specific implementation and functionality are detailed in Section 3.3.

2.3.1 Extended Kalman Filter

The Kalman filter is a powerful sensor-fusion tool that enhances measurement accuracy by combining a process model with a measurement model. While the true state remains unknown, the filter tracks uncertainties in both models via a covariance matrix. By fusing these components, the filter calculates a gain $\mathbf{K}$ that minimizes the posterior error covariance, yielding a statistically optimal state estimate with minimum variance.

While the classic Kalman filter assumes linear models, the Extended Kalman Filter (EKF), handles non-linear systems by linearizing the non-linear model around the current state estimate. This recursive filter predicts the next state using a process model and updates this prediction using new measurements. These operations comprise the prediction and correction steps. The prediction step is expressed as [18, p. 113]:

$$\begin{aligned}
\mathbf{x}_{k|k-1} &= \mathbf{f}(\mathbf{x}_{k-1|k-1}, \mathbf{u}_{k-1}) \\
\mathbf{P}_{k|k-1} &= \mathbf{F}\mathbf{P}_{k-1|k-1}\mathbf{F}^\top + \mathbf{G}\mathbf{R}_u\mathbf{G}^\top + \mathbf{Q} \\
\mathbf{F} &= \left. \frac{d\mathbf{f}}{d\mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}_{k-1|k-1}} \\
\mathbf{G} &= \left. \frac{d\mathbf{f}}{d\mathbf{u}} \right|_{\mathbf{u}=\mathbf{u}_{k-1}}
\end{aligned} \tag{2.11}$$

where $\mathbf{Q}$ is the process noise matrix and $\mathbf{P}$ is the state uncertainty matrix. While measurements are sometimes used as input [18, p. 82], adding their covariance $\mathbf{R}_u$ via the term $\mathbf{G}\mathbf{R}_u\mathbf{G}^\top$ is not standard in the literature, but mirrors the addition of the state covariance $\mathbf{P}_{k-1|k-1}$ via $\mathbf{F}\mathbf{P}_{k-1|k-1}\mathbf{F}^\top$. The correction step equations are:

$$\mathbf{V} = \mathbf{z}_k - \mathbf{h}(\mathbf{x}_{k|k-1}) \tag{2.12}$$

$$\mathbf{H} = \left. \frac{d\mathbf{h}}{d\mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}_{k|k-1}} \tag{2.13}$$

$$\mathbf{S} = \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^\top + \mathbf{R} \tag{2.14}$$

$$\mathbf{K} = \mathbf{P}_{k|k-1}\mathbf{H}^\top\mathbf{S}^{-1} \tag{2.15}$$

$$\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + \mathbf{K}\mathbf{V} \tag{2.16}$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P}_{k|k-1}, \tag{2.17}$$

where $\mathbf{h}(\mathbf{x}_{k|k-1})$ maps the state to the measurements, $\mathbf{z}_k$ represents the measurement at step k , and $\mathbf{R}$ is the measurement covariance matrix. Equations 2.14 and 2.15 define the innovation covariance $\mathbf{S}$ and the Kalman gain $\mathbf{K}$. A large $\mathbf{S}$ yields a small $\mathbf{K}$, causing the filter to rely more on the process prediction and reducing the measurement update step. A small Kalman gain indicates low confidence in the measurement. Conversely, a small $\mathbf{S}$ results in a large $\mathbf{K}$, signaling high confidence and producing a larger update step. As shown in Equation 2.17, the updated state uncertainty matrix $\mathbf{P}$ shrinks proportionally to $\mathbf{K}$.

2.3.2 Model Predictive Control (MPC)

Model Predictive Control (MPC) is a method that can be used for autonomous marine docking [12]. The MPC algorithm computes optimal control inputs subject to defined constraints and the current state. It generates a control sequence that tracks a reference trajectory while satisfying all system constraints. At each time step, only the first control input from this sequence is applied. For the next iteration, the newest state estimates serves as initial states.

Figure 2.2 illustrates what happens before and during the prediction horizon, which defines how far into the future the controller predicts the system states. At each step, predicted control inputs propagate the state estimates across this horizon. The predicted output converges to the reference trajectory within this window, and the

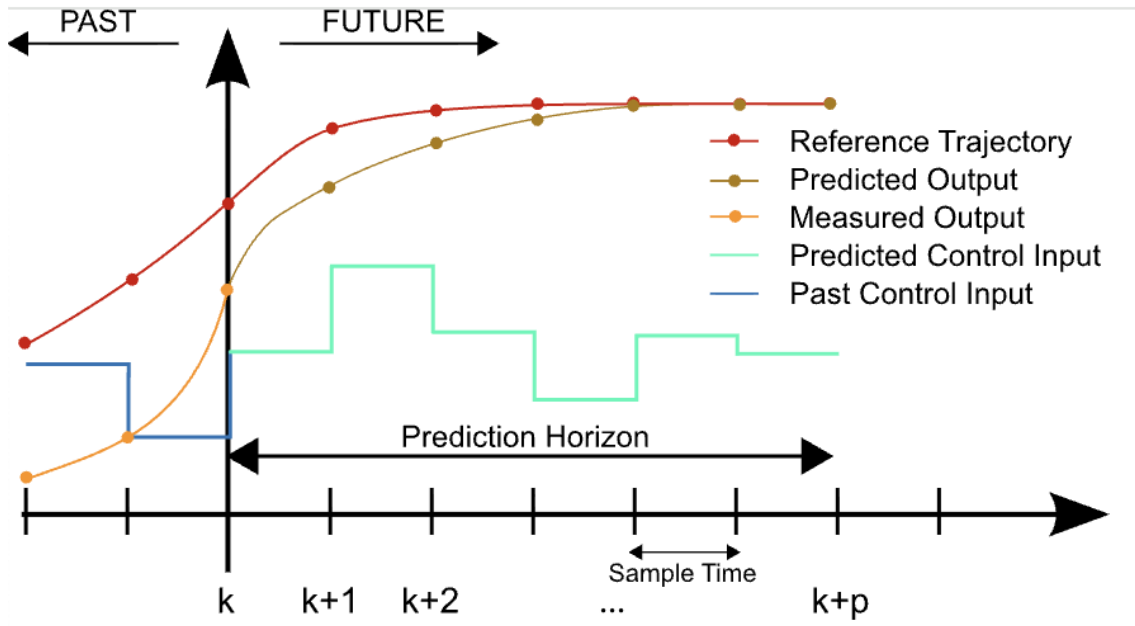


Figure 2.2: MPC illustrated, By Martin Behrendt - Own work, CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=7963069>

terminal point represents the final state. MPC identifies the optimal path to this final state. Optimization is governed by a cost function subject to state and input constraints, formulated as

$$V_N^*(x) = \min_{\mathbf{u}(0:N-1)} V_N(u(0:N-1), x(0:N)) \quad (2.18)$$

where

$$V_N(u(0:N-1), x(0:N)) = \mathbf{e}^\top(N) \mathbf{P}_f \mathbf{e}(N) + \sum_{k=0}^{N-1} (\mathbf{e}^\top(k) \mathbf{Q} \mathbf{e}(k) + \mathbf{u}^\top(k) \mathbf{R} \mathbf{u}(k))$$

and

$$\mathbf{e}(0:N) = \mathbf{x}(0:N) - \mathbf{r}_{ref}(0:N)$$

subject to

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) \\ \mathbf{x}_{min} &\leq \mathbf{x}(k) \leq \mathbf{x}_{max} \\ \mathbf{u}_{min} &\leq \mathbf{u}(k) \leq \mathbf{u}_{max} \end{aligned} \quad (2.19)$$

where N is the horizon length, $\mathbf{A}$ and $\mathbf{B}$ are system state-space matrices, tunable penalty matrix $\mathbf{Q}$ penalizes state error during the entire horizon, the matrix $\mathbf{R}$ penalizes control signals, and the matrix $\mathbf{P}_f$ penalizes the final state error at the end of the horizon [5].

MPC can enforce explicit constraints on translational motion, rotational motion, and actuator inputs. The constraints are paramount for both safety and actuator feasibility during a docking maneuver. What separates MPC from the Linear Quadratic Regulator (LQR) is the explicit handling of constraints. The LQR is the unconstrained, infinite-horizon special case of the same quadratic optimal control problem, yielding a static feedback gain $\mathbf{u} = -\mathbf{K}\mathbf{x}$ obtained from the discrete algebraic Riccati equation [6]. Because the LQR carries no notion of constraints, input limits must be enforced post-optimization by clipping.

Furthermore, MPC constraints and penalty matrices bound controller aggressiveness, directly influencing energy consumption. The objective is to maximize tracking accuracy while minimizing control effort. For a single-ended ferry, terminal heading is critical. MPC allows the system to plan ahead, ensuring the correct orientation before reaching the dock.

2.3.3 Robot Operating System 2 (ROS 2)

Robot Operating System 2 (ROS 2) [8] is a framework of tools and libraries for robotic software development. It operates using nodes and topics; a node is a modular software unit executed on an event-driven basis. Events include incoming sensor data or timer-triggered control loops. Topics act as communication channels where nodes publish data for subscriber nodes to consume.

For example, this project features a state estimator node. It subscribes to sensor topics, estimates the state vector, and publishes the results to an estimation topic. Subscribed controller nodes then consume these updates synchronously.

2.4 Virtual RobotX (VRX)

Virtual RobotX (VRX) is a simulation environment for unmanned surface vessels [7]. Figure 2.3 shows a screenshot of this open-source extension to the Gazebo simulator. Key features include:

- Ocean-spectra wave models that affect vessel dynamics and simulated sensors.
- Stochastic wind disturbances.
- Approximated buoyancy forces acting on geometric shapes to enable a floating behavior.
- A parameterized, non-linear, configurable propulsion thrust model.

The simulation runs in real time at the expense of absolute physical accuracy. It aims to replicate qualitative real-world behaviors, though quantitative magnitudes may vary. Its design philosophy dictates that controllers that fail in simulation will fail in reality and controllers that works in reality will work in simulation, making simulations in VRX a suitable testing phase prior to physical deployment.

The simulator interfaces via ROS 2 by listening to thrust command topics, publishing sensor data, and broadcasting simulation metrics like simulation time. This allows interaction via standard ROS 2 tools like RQt alongside custom controller nodes.



Figure 2.3: Screenshot from VRX.

3

Method

This chapter covers the implementation of the theory described in the previous chapter, along with motivations behind the methodology. It begins with a description of the ferry hardware, followed by an explanation of system identification and dynamic modeling. The subsequent section details the controller architecture, and the chapter concludes with a brief overview of how AI tools are utilized throughout the project.

3.1 Hardware

Although the controller is never physically deployed, it is designed to be compatible with the hull geometry, sensor configuration, and actuators described in this section.

3.1.1 Hull and Actuators

CStrider's first prototype vessel Celia One, shown in Figure 3.1, is a catamaran equipped with two independently steerable electric outboard motors attached to the aft of each hull. It also features a bow thruster in each hull, with all actuators managed by a centralized onboard computer.

3.1.2 Sensors

The ferry features a dual-antenna Inertial Navigation System (INS) with an integrated IMU and state estimator. The dual-antenna configuration allows heading estimation solely from GNSS measurements. The State-Estimator provides the following kinematic states:

- Position
- Velocity
- Acceleration
- Orientation
- Angular Velocities



Figure 3.1: Image of Celia One from .

It is also equipped with other sensors, such as cameras and radars, but those are not utilized in this project.

3.2 System Identification

Modeling the system is a critical step in designing the controller. This section explains how the maneuvering model is applied, how its parameters are estimated, and how the physical motor thrust is quantified.

3.2.1 Marine Vessel Modeling

To implement the 3-DoF maneuvering model [17, p. 109], the original equations are simplified. Assuming that the axis of rotation is located at the geometric center of the ferry, the rigid-body mass matrix from Equation 2.4 becomes

$$\mathbf{M}_{RB} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & mx_{\text{cog}} \\ 0 & mx_{\text{cog}} & I_z \end{bmatrix} \approx \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I_z \end{bmatrix}. \quad (3.1)$$

Furthermore, assuming that the off-diagonal elements are small [17, p. 119], the added-mass matrix simplifies to

$$\mathbf{M}_A = \begin{bmatrix} -X_{\dot{u}} & 0 & 0 \\ 0 & -Y_{\dot{v}} & -Y_{\dot{r}} \\ 0 & -Y_{\dot{r}} & -N_{\dot{r}} \end{bmatrix} \approx \begin{bmatrix} -X_{\dot{u}} & 0 & 0 \\ 0 & -Y_{\dot{v}} & 0 \\ 0 & 0 & -N_{\dot{r}} \end{bmatrix}. \quad (3.2)$$

These two assumptions also reduces the Coriolis matrices from Equations 2.3 and 2.5 to

$$\mathbf{C}_{RB}(\mathbf{v}) = \begin{bmatrix} 0 & 0 & -m(x_{\text{cog}}r + v) \\ 0 & 0 & mu \\ m(x_{\text{cog}}r + v) & -mu & 0 \end{bmatrix} \approx \begin{bmatrix} 0 & 0 & -mv \\ 0 & 0 & mu \\ mv & -mu & 0 \end{bmatrix} \quad (3.3)$$

and

$$\mathbf{C}_A(\mathbf{v}) = \begin{bmatrix} 0 & 0 & Y_{\dot{v}}v + Y_{\dot{r}}r \\ 0 & 0 & -X_{\dot{u}}u \\ -Y_{\dot{v}}v - Y_{\dot{r}}r & X_{\dot{u}}u & 0 \end{bmatrix} \approx \begin{bmatrix} 0 & 0 & Y_{\dot{v}}v \\ 0 & 0 & -X_{\dot{u}}u \\ -Y_{\dot{v}}v & X_{\dot{u}}u & 0 \end{bmatrix}. \quad (3.4)$$

For the controller, the hydrodynamic damping from Equation 2.6 is assumed to be linear:

$$\mathbf{D}(\mathbf{v}) = [f_X(\mathbf{v}) \quad f_Y(\mathbf{v}) \quad f_N(\mathbf{v})] \approx \text{diag} [-X_u \quad -Y_v \quad -N_r] = \mathbf{D}. \quad (3.5)$$

Note that $X_u \neq X_{\dot{u}}$, $Y_v \neq Y_{\dot{v}}$ and $N_r \neq N_{\dot{r}}$. This assumption is justified by the results of Experiments 1, 2, and 3 in Section 4.1.1. The non-linearities in the data are small within the operational domain of the controller; thus, a linear approximation of the damping coefficients is sufficient for the docking procedure. Nonetheless, these experiments are performed to evaluate the extent of the non-linearities and to accurately determine the parameters of $\mathbf{D}$. However, the assumption of linearity is not made when using the hydrodynamical model to estimate the physical motor thrust. More on that in Section 3.2.5.2.

Equations 3.6–3.12 define the simplified hydrodynamic equations used by the controller in this project:

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}(\mathbf{v})\mathbf{v} + \mathbf{D}(\mathbf{v})\mathbf{v} = \boldsymbol{\tau}_{motors} \quad (3.6)$$

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}(\mathbf{v})\mathbf{v} + \mathbf{D}(\mathbf{v})\mathbf{v} = \mathbf{0} \quad (3.7)$$

$$\mathbf{D}(\mathbf{v})\mathbf{v} = -\mathbf{M}\dot{\mathbf{v}} - \mathbf{C}(\mathbf{v})\mathbf{v} \quad (3.8)$$

$$\mathbf{M} = \begin{bmatrix} m - X_{\dot{u}} & 0 & 0 \\ 0 & m - Y_{\dot{v}} & 0 \\ 0 & 0 & I_z - N_{\dot{r}} \end{bmatrix} \quad (3.9)$$

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & Y_{\dot{v}}v - mv \\ 0 & 0 & mu - X_{\dot{u}}u \\ mv - Y_{\dot{v}}v & X_{\dot{u}}u - mu & 0 \end{bmatrix} \quad (3.10)$$

$$\mathbf{D} = \text{diag} [-X_u \quad -Y_v \quad -N_r] \quad (3.11)$$

$$\boldsymbol{\tau}_{motors} = [X, Y, N]^\top \quad (3.12)$$

From these formulations, the non-linear equations of motion are derived. The accelerations in surge, sway, and yaw are expressed as

$$\dot{\mathbf{v}} = \mathbf{M}^{-1}(\boldsymbol{\tau}_{motors} - \mathbf{C}(\mathbf{v})\mathbf{v} - \mathbf{D}(\mathbf{v})\mathbf{v}) = \begin{bmatrix} \frac{-(X + X_u u - Y_{\dot{v}} r v + m r v)}{X_{\dot{u}} - m} \\ -\frac{Y + Y_v v + X_{\dot{u}} r u - m r u}{Y_{\dot{v}} - m} \\ \frac{N + N_r r - X_{\dot{u}} u v + Y_{\dot{v}} u v}{I_z - N_{\dot{r}}} \end{bmatrix}. \quad (3.13)$$

3.2.2 Estimating Added Mass With Strip Theory

In the maneuvering model, pitch and roll are neglected, meaning the coefficients of the $\mathbf{M}_A$ matrix are static. Because the hull is slender, the hydrodynamic derivatives

in the $\mathbf{M}_A$ matrix from Equation 3.2 are estimated using strip theory [16, pp. 37–40]. The added-mass coefficients depend on the geometric properties of the submerged hull. In strip theory, the submerged hull is divided into cross-sectional slices, or strips, along its length to estimate the two-dimensional added mass of each section. These 2D values are then integrated over the hull length to compute the total added mass, a method applied here for sway. Fossen proposes estimating the added mass in surge as 5% of the total mass when the submerged hull is approximated as a half cylinder [16, p. 39]. The added mass for yaw rotation combines both approaches, as shown later in Equation 3.20.

For the derivation, the geometry of the submerged hull is first approximated. The actual and approximated cross-sections of one pontoon of the catamaran hull are visualized in Figure 3.2. The gray area represents the physical hull, while the orange semicircle depicts the approximated submerged hull. Let r_{hull} be the radius of the hull, $l_{\text{waterline}}$ the distance from the centerline to the waterline, d_{draft} the draft, and $\hat{r}$ the radius of the approximated semicircular submerged hull. The radius $\hat{r}$ is calculated as the mean of d_{draft} and $l_{\text{waterline}}$. Because $l_{\text{waterline}}$ is difficult to measure directly, it is approximated as $l_{\text{waterline}} \approx r_{\text{hull}}$, yielding:

$$\hat{r} = \frac{d_{\text{draft}} + r_{\text{hull}}}{2}. \quad (3.14)$$

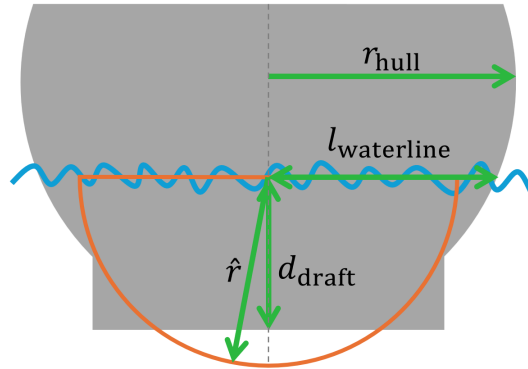


Figure 3.2: Cross section of submerged hull in gray, the approximated submerged hull in orange, different dimensions illustrated with green arrows and the waterline illustrated in blue.

For a half circle approximation of a monohull, the added masses for surge, sway, and yaw are estimated as

$$-X_{\dot{u}} = 0.05 m \quad (3.15)$$

$$-Y_{\dot{v}} = \frac{1}{2} \int_{-L/2}^{L/2} \rho \pi \hat{r}^2 dx = \frac{1}{2} \rho \pi \hat{r}^2 L \quad (3.16)$$

$$\begin{aligned} -N_{\dot{r}} &= \frac{1}{2} \int_{-B/2}^{B/2} y^2 \frac{0.1m}{B} dy + \frac{1}{2} \int_{-L/2}^{L/2} x^2 \rho \pi \hat{r}^2 dx \\ &= \frac{1}{24} (0.1 m B^2 + \rho \pi \hat{r}^2 L^3) \end{aligned} \quad (3.17)$$

where m is the mass of the ferry, ρ is the water density, L is the hull length, B is the beam width, and x and y are coordinates relative to the center of rotation in yaw and $\hat{r}$ is assumed to be the same throughout the length of the hull. The minus sign notation is from the literature.

As no literature was found regarding the adaptation of this method for catamarans, doubling the monohull added mass for surge and sway is assumed to be a sufficient approximation. For $-N_{\dot{r}}$, the integral over y is evaluated from $l - \frac{B}{2}$ to $l + \frac{B}{2}$, where l represents the distance from the vessel centerline to the centerline of an individual hull. This value is then doubled to account for both hulls. Thus, the added masses for the catamaran are assumed to be

$$-X_{\dot{u}} = 2 \times 0.05 m_h = 0.05 m, \quad (3.18)$$

$$-Y_{\dot{v}} = 2 \times \frac{1}{2} \int_{-L/2}^{L/2} \rho \pi \hat{r}^2 dx = \rho \pi \hat{r}^2 L, \quad (3.19)$$

and

$$\begin{aligned} -N_{\dot{r}} &= 2 \times \left(\frac{1}{2} \int_{l-B/2}^{l+B/2} y^2 \frac{0.1m_h}{B} dy + \frac{1}{2} \int_{-L/2}^{L/2} x^2 \rho \pi \hat{r}^2 dx \right) \\ &= \frac{1}{12} \left(0.1 \frac{m}{2} (B^2 + 12l^2) + \rho \pi \hat{r}^2 L^3 \right), \end{aligned} \quad (3.20)$$

where $m_h = \frac{m}{2}$ is the mass supported by each hull, and the remaining variables match the monohull definition.

3.2.3 State Space Modeling

The 3-DoF maneuvering model for trajectory tracking is formulated as a six-state model to track both pose and velocity. The complete state vector is defined as

$$\boldsymbol{\eta} = \begin{bmatrix} E & N & \psi & u & v & r \end{bmatrix}^\top, \quad (3.21)$$

which combines the global position and body-frame velocities.

The full non-linear dynamics are expressed by combining Equation 3.13 with the velocity in the world coordinate frame.

$$\dot{\boldsymbol{\eta}} = \mathbf{f}(\boldsymbol{\eta}, \boldsymbol{\tau}_{\text{motors}}) = \begin{bmatrix} \mathbf{R}(\psi)\mathbf{v} \\ \mathbf{M}^{-1}(\boldsymbol{\tau}_{\text{motors}} - \mathbf{C}(\mathbf{v})\mathbf{v} - \mathbf{D}(\mathbf{v})\mathbf{v}) \end{bmatrix}, \quad (3.22)$$

where

$$\mathbf{R}(\psi) = \begin{bmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

and $\mathbf{v}$ is the body-frame velocities defined by Equation 2.1.

The model is linearized using a first-order Taylor expansion:

$$\mathbf{f}(\boldsymbol{\eta}, \boldsymbol{\tau}) = \mathbf{f}(\boldsymbol{\eta}_{op}, \boldsymbol{\tau}_{op}) + \frac{d\mathbf{f}(\boldsymbol{\eta}, \boldsymbol{\tau})}{d\boldsymbol{\eta}}(\boldsymbol{\eta} - \boldsymbol{\eta}_{op}) + \frac{d\mathbf{f}(\boldsymbol{\eta}, \boldsymbol{\tau})}{d\boldsymbol{\tau}}(\boldsymbol{\tau} - \boldsymbol{\tau}_{op}), \quad (3.23)$$

where the subscript op denote the operating point. This is rewritten as

$$\mathbf{f}(\boldsymbol{\eta}, \boldsymbol{\tau}) = \mathbf{f}(\boldsymbol{\eta}_{op}, \boldsymbol{\tau}_{op}) + \mathbf{A}(\boldsymbol{\eta} - \boldsymbol{\eta}_{op}) + \mathbf{B}(\boldsymbol{\tau} - \boldsymbol{\tau}_{op}) \quad (3.24)$$

$$= \mathbf{A}\boldsymbol{\eta} + \mathbf{B}\boldsymbol{\tau} + \mathbf{C}, \quad (3.25)$$

where

$$\mathbf{C} = \mathbf{f}(\boldsymbol{\eta}_{op}, \boldsymbol{\tau}_{op}) - \mathbf{A}\boldsymbol{\eta}_{op} - \mathbf{B}\boldsymbol{\tau}_{op}. \quad (3.26)$$

By using the current state estimate as the operating point, the introduction of an offset variable $\mathbf{C}$ eliminates the need for state deviation variables, serving instead as a direct offset [22]. Finally, the system is discretized using a zero-order hold, and $\boldsymbol{\tau}_{op} = \mathbf{0}$, yielding the discrete state-space model:

$$\boldsymbol{\eta}(k+1) = \mathbf{A}_d\boldsymbol{\eta}(k) + \mathbf{B}_d\boldsymbol{\tau}(k) + \mathbf{C}_d \quad (3.27)$$

3.2.4 Experiment Setup

Six experiments are designed for the project. Experiments 1 through 3 estimate the hydrodynamic parameters, Experiments 4 and 5 characterize the motor parameters, and Experiment 6 is intended to fine-tune and validate the hydrodynamic model.

Due to time limitations, physical testing on the actual ferry was not possible; consequently, all experimental data are collected via simulated sensors in VRX. However, all tests and parameter estimation methods are fully compatible with real-world ferry data.

The same data set is logged across all trials consisting of both raw sensor measurements and state estimations:

- Timestamp

- Raw longitude and latitude from the GNSS
- Estimated yaw
- Estimated surge and sway
- High-pass filtered yaw rate from gyro
- High-pass filtered linear accelerations in surge and sway from the accelerometer
- Thrust and steering angle commands sent to the motors

Experiment 5 is an exception to this, as explained below. Each experiment is conducted multiple times to ensure repeatability, to verify that the data remains untainted by uncontrolled variables, and to mitigate sensor noise through averaging to approach the true mean values.

To minimize the impact of environmental variables like wind and currents, specific testing orientations are chosen. For instance, during surge tests, the vessel is oriented so that environmental disturbances hit the hull laterally, isolating the effects to sway. Additionally, trials are conducted an even number of times, alternating directions (e.g., swapping between easterly and westerly courses during a northerly wind) to cancel out directional bias.

3.2.4.1 Experiment 1

This experiment gathers data to estimate water drag in surge. The vessel accelerates to maximum surge before all engine thrust is cut, allowing water drag to decelerate the ferry. The logged surge data are used to approximate $f_X(\mathbf{v})$ from Equation 2.6. In reality on Celia One, this test would be performed in reverse, as the hull geometry generates higher damping when moving astern. However, the hull in VRX is very similar in the bow and stern, thus the dampening in positive and negative surge is assumed to be equal and the test is only performed forward.

3.2.4.2 Experiment 2

This experiment replicates the procedure of Experiment 1 but focuses on sway to approximate $f_Y(\mathbf{v})$ from Equation 2.6. Sideways translation is achieved using the dual-motor configuration illustrated in Figure 3.3. However, due to the dynamic and unpredictable nature of marine maneuvering, it is not possible to achieve pure surge motion using static motor angles. Feedback through computer or human intervention is needed to produce accurate test data.

3.2.4.3 Experiment 3

The third experiment evaluates damping in yaw rotation to approximate $f_N(\mathbf{v})$ from Equation 2.6. Pure yaw rotation is generated by aligning both motors straight, running one forward while reversing the other.

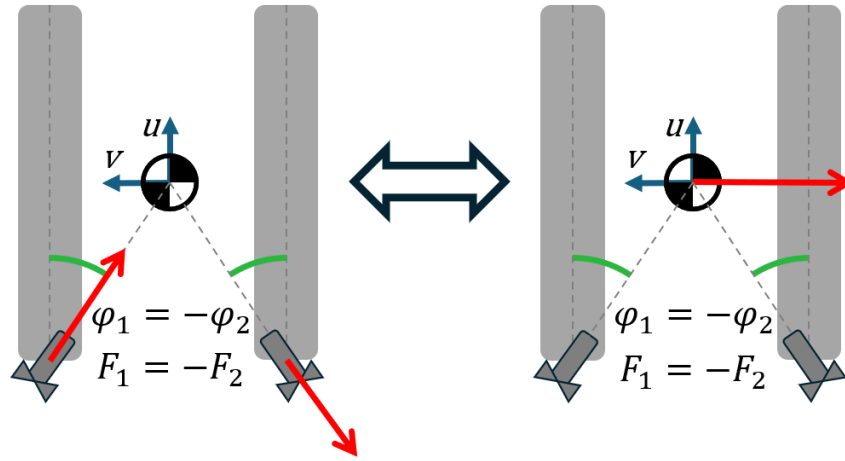


Figure 3.3: How to drive a catamaran sideways using a dual motor setup. Point both motors outwards approximately 25 degrees so that their thrust vector intersects the axis of rotation in yaw. Then reverse one motor and drive the other one forward. This will result in a combined force acting in sway.

3.2.4.4 Experiment 4

This experiment measures accelerations of varying intensities at different surge velocities across four phases, each defined by a specific baseline thrust level:

- Phase 1: 0%
- Phase 2: 25%
- Phase 3: 50%
- Phase 4: 75%

Each phase consists of four steps. Each step is performed by executing these four sub-steps:

1. Apply the phase base thrust and wait for a steady-state velocity.
2. Increase thrust by 5, 10, 15, or 20 percentage points above the baseline (for steps 1, 2, 3, and 4, respectively).
3. Wait for a new steady-state velocity.
4. Proceeding to the next step.

For example, during Step 3 of Phase 2, the motors stabilize at a 25% baseline, step up by 15 percentage points to 40%, reach a stable speed, and return to 25% before initiating the final 20 percentage step. Figure 3.4 illustrates typical surge and thrust command profiles during this experiment.

The objective is to derive a three-dimensional relationship mapping acceleration

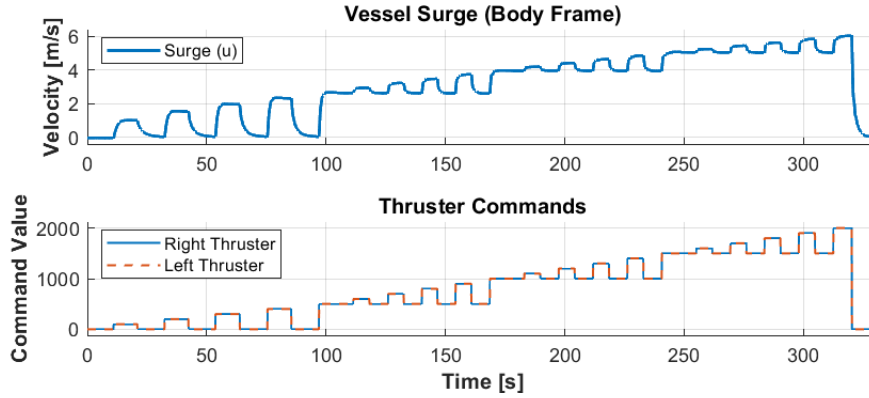


Figure 3.4: Experiment 4. This is an example of how the thrust command and surge can look when following the sequence explained for Experiment 4.

against thrust increments across multiple speeds. By applying the dynamic model parametrized in Experiments 1–3, these accelerations are converted into physical driving forces. The final output yields a three-dimensional relationship estimated by an equation relating force and water velocity over the propeller to thrust commands.

3.2.4.5 Experiment 5

To determine the maximum steering angle and angular velocity, the motors are commanded to cycle between full port and full starboard deflection. The actual motor angle feedback is recorded and differentiated to find the steering actuation speed. This test is evaluated at multiple surge velocities to account for potential friction increases induced by propeller thrust.

In VRX, where default angular speeds are unrealistically high, the controller is capped at a steering rate of $18^\circ/s$ to approximate the physical limits of Celia One.

3.2.4.6 Experiment 6

The final proposed experiment involves navigating a sharp zig-zag or S-shaped pattern. The data is intended to be used to fine-tune and validate the dynamic model parameters, as explained in in the next section. However, due to time constraints, this test was omitted.

3.2.5 Parameter Estimation

Data from Experiments 1, 2, and 3 are utilized to estimate the damping coefficients X_u , Y_v , and N_r for the matrix $\mathbf{D}$ in Equation 3.6. Because the estimation procedures for Y_v and N_r mirror that of X_u , only the derivation for X_u is detailed below. This methodology is inspired by the work of Danielsson and Lindström [11]. The estimation of motor forces using Experiment 4 data is discussed subsequently, as it requires an estimation of $\mathbf{D}$.

Initial parameter estimates can be refined by logging vessel trajectories during highly

dynamic maneuvers, such as the zig-zag patterns outlined in Experiment 6. By running a parallel simulation of the dynamic model using identical thrust and steering inputs, the logged and simulated trajectories are overlaid. The parameters are then manually adjusted until the tracks align closely, making the simulated boat model match the behavior of the real vessel. Due to time constraints, this optimization phase is omitted.

3.2.5.1 Estimating the Dampening Parameters

The first row of the complete system dynamics (Equation 3.6), which represents surge dynamics, is rearranged to isolate the total damping force $X_u u$:

$$X_u u = (m - X_{\dot{u}})\dot{u} + (Y_{\dot{v}} - mv)r - X \quad (3.28)$$

Because $X_u u$ describes the damping forces in surge, and all variables on the right-hand side are known from Experiment 1, mapping the logged $(u, \dot{u})$ data points yields a set of discrete (u, F_{damping}) pairs. These pairs are plotted to visualize the relationship between surge velocity and water resistance.

Because data collection occurs at a fixed frequency, the dataset contains significantly more points at low velocities than at higher velocities. To prevent low-speed bias, data points are weighted inversely to their local density during fitting.

The resulting dataset is fitted to four separate polynomials. Two of them are on linear form, $p_{\text{lin}} u$, one with weighted points and one with unweighted points. The other two are on quadratic form, $p_1^{\text{quad}} u + p_2^{\text{quad}} u^2$, one with weighted points and one with unweighted points. The linear models are designated for the controller architecture, while the more precise quadratic fits—which capture non-linear hydrodynamic features—are used to estimate the motor forces in the next section.

Crucially, the polynomials directly estimate the total force $X_u u$ rather than the isolated coefficient X_u , meaning $X_u \approx p_{\text{lin}}$ and $X_u \approx p_1^{\text{quad}} + p_2^{\text{quad}} u$. Although isolating X_u directly from the outset simplifies the process, both mathematical formulations are valid. Thus, it will not be recalculated using the simpler method.

To calculate Y_v and N_r , the second and third rows of the system dynamics (Equation 3.6) are used and $Y_v v$ and $N_r r$ is isolated and approximated using sway data from Experiment 2 and yaw rate data from Experiment 3. Because no physical sensor measures yaw acceleration ($\dot{r}$), it is approximated by numerically differentiating the estimated yaw rate.

3.2.5.2 Estimating the Force Produced by the Motors

A mathematical mapping is required to convert the force demands computed by the controller (in Newtons) into unitless motor commands.

The thrust generated by a given command depends heavily on the advance velocity

of the water passing through the propeller disk. At a constant rotational speed, net thrust decreases as the water velocity through the propeller increases, moving toward a condition where propeller blade pitch matches fluid velocity and no fluid is accelerated rearward.

To characterize this, a three-dimensional point cloud composed of surge velocity, thrust commands, and measured accelerations is extracted from the Experiment 4 dataset. Surge is equal to the water velocity passing through the propeller disk when the motors are not angled. Using the dynamical model, these entries are converted into discrete points relating surge, thrust command, and physical motor force, to which a second-order polynomial is fitted.

By extracting the surge equation from the system dynamics (Equation 3.6) and substituting the quadratic damping model, the net motor force X is isolated directly:

$$X = (m - X_{\ddot{u}})\dot{u} + (Y_{\dot{v}} - mv)r - (p_1^{quad} + p_2^{quad}u)u \quad (3.29)$$

where X represents the sole unknown variable. The structured steps in Experiment 4 ensure an even distribution of data across a broad envelope of accelerations, velocities, and input commands, which are processed through this equation to generate the force dataset.

Finally, a second-order polynomial is fitted to the data, using surge velocity and requested force as inputs to output the corresponding thrust command. The polynomial omits a constant intercept term, enforcing the physical constraint that zero thrust command corresponds to zero generated force when stationary:

$$f(x, y) = p_1x + p_2y + p_3x^2 + p_4xy + p_5y^2 \quad (3.30)$$

where x is the surge velocity, y is the requested force, and $f(x, y)$ is the target thrust command. The thrust command is used as the output of this polynomial, instead of the motor force, because that is how it is implemented in the controller. A motor force is requested, the surge is estimated and these are then used to calculate a thrust command.

3.3 Controller

The controller architecture prioritizes modularity because the project scale is large relative to the available development time. The primary goal is to build a functional and upgradable system rather than a perfectly optimized one. A modular design allows independent module upgrades without altering the entire controller structure. It is also easier for a new team to understand how individual sub-components work than a large complex single-structure controller.

The modules are implemented as ROS 2 nodes for two primary reasons. First, ROS 2 fits a modular architecture; nodes run independently while communicating seamlessly via the publisher/subscriber paradigm. Second, it enables direct integration with the Virtual RobotX (VRX) simulation software, which interfaces natively via

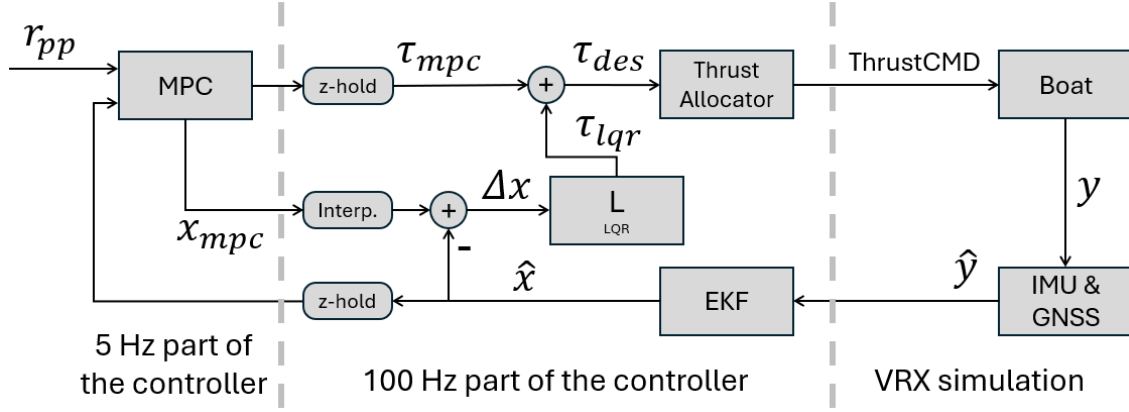


Figure 3.5: Controller architecture. r_{pp} is the reference path from the Path-Planner, τ_{mpc} and τ_{lqr} are the generalized forces calculated by the MPC and LQR respectively, τ_{des} is the combined desired generalized control force, ThrustCMD is the motor control forces and angles, and x is the state vector. In this figure, x represents both x and η . η is a state vector in another frame explained bellow.

ROS 2. Additionally, implementing nodes in Python leverages team familiarity and saves valuable development time.

Figure 3.5 shows the controller architecture that is be explained in this section.

3.3.1 Global Frame and Body Frame

The controller subsystems use different reference frames. This project employs two primary frames: the global East-North-Up (ENU) frame and the body-fixed Forward-Left-Up (FLU) frame (Figure 3.6). Heading is the angle between the global east axis and the body forward axis. Because the rotation axis remains the same under coordinate transformations, heading is identical in both frames. Rotation matrices translate vectors between these frames:

$$\mathbf{R}_{F2E}(\psi) = \begin{bmatrix} \cos \psi & -\sin \psi \\ \sin \psi & \cos \psi \end{bmatrix} \quad (3.31)$$

and

$$\mathbf{R}_{E2F}(\psi) = \mathbf{R}_{F2E}(\psi)^\top = \begin{bmatrix} \cos \psi & \sin \psi \\ -\sin \psi & \cos \psi \end{bmatrix} \quad (3.32)$$

where ψ is the heading and $\mathbf{R}_{F2E}(\psi)$ and $\mathbf{R}_{E2F}(\psi)$ are rotational matrices with subscripts denoting FLU-to-ENU and ENU-to-FLU transforms, respectively. For example:

$$\mathbf{p}_{\text{ENU}} = \mathbf{R}_{F2E}(\psi) \mathbf{p}_{\text{FLU}} \quad (3.33)$$

where $\mathbf{p}_{\text{FLU}}$ and $\mathbf{p}_{\text{ENU}}$ represent vectors in the FLU and ENU frames, respectively. These vectors typically contain velocities or accelerations.

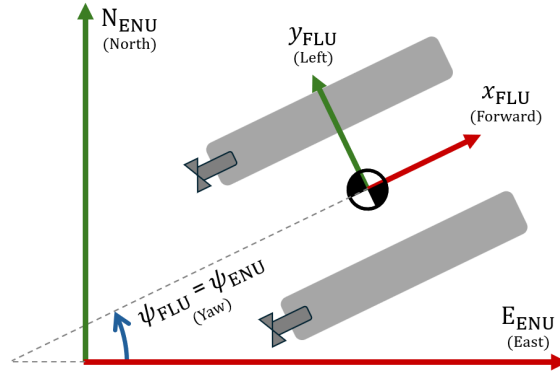


Figure 3.6: The ferry in the FLU body frame relative to the ENU global frame. The heading angle is measured relative to the east axis of the global frame.

3.3.2 State-Estimator

The State-Estimator within the ferry's Inertial Navigation Unit (INU) estimates all required states (see Section 3.1.2). However, its position estimates update at only 10Hz, which is slower than the intended controller operating frequency. Consequently, implementing the controller on the physical ferry requires a separate higher frequency Kalman filter or a prediction model, which falls outside the scope of this project.

The Virtual RobotX (VRX) simulation environment lacks an integrated INU filter, providing only independent simulated GNSS and IMU sensors. Therefore, running the controller in VRX requires a custom State-Estimator. This project implements an Extended Kalman Filter (EKF) based on a kinematic process model. This model assumes constant acceleration during each sample period and does not model the hydrodynamical forces. While less accurate than a hydrodynamic process model, low-magnitude measurement noise allows sensor updates to correct the estimated states. The kinematic EKF is computationally efficient, ensuring robustness at high frequencies, and its low complexity simplifies maintenance and updates. The primary trade-off is reduced prediction accuracy due to process model bias [13].

The state vector for the EKF is,

$$\mathbf{x}_{\text{ENU}} = [E \quad N \quad \dot{E} \quad \dot{N} \quad \psi]^\top, \quad (3.34)$$

where (E, N) are position coordinates in the East-North-Up (ENU) inertial frame and ψ denotes the heading angle. The input vector combines accelerometer readings and gyroscope yaw rate measurements:

$$\mathbf{u}_{\text{FLU}} = [\dot{u} \quad \dot{v} \quad r]^\top, \quad (3.35)$$

where $(\dot{u}, \dot{v})$ represent forward and lateral linear accelerations relative to the body-fixed Forward-Left-Up (FLU) coordinate frame. However, waves affect pitch and

roll, corrupting accelerometer measurements. Therefore, the measurements require gravity compensation to prevent roll and pitch angles from corrupting the readings. Because the ground-truth orientation is accessible in VRX, the ground-truth pitch (θ) and roll (ϕ) angles are used directly to compensate for these gravity-induced acceleration components without adding complexity to the state estimator:

$$\mathbf{u}_{FLU} = \begin{bmatrix} \dot{u} - g \sin \theta & \dot{v} + g \sin \phi \cos \theta & r \end{bmatrix}^\top. \quad (3.36)$$

This use of ground-truth values is deemed okay because this State-Estimator will never be implemented on the real ferry. Since the state vector is expressed in the world frame (ENU) and the inputs in Equation 3.35 are measured in the body frame (FLU), the acceleration measurements must be transformed to ENU, as discussed in Section 3.3.1:

$$\mathbf{a}_{ENU} = \begin{bmatrix} \ddot{E} \\ \ddot{N} \end{bmatrix} = \mathbf{R}_{F2E}(\psi) \mathbf{a}_{FLU} = \mathbf{R}_{F2E}(\psi) \begin{bmatrix} \dot{u} - g \sin \theta \\ \dot{v} + g \sin \theta \cos \phi \end{bmatrix} \quad (3.37)$$

Since the z-axis remain the same in both frames, r is also identical and the input vector in the world coordinate frame (ENU) is then expressed as

$$\mathbf{u} = \begin{bmatrix} \ddot{E} & \ddot{N} & r \end{bmatrix}^\top. \quad (3.38)$$

The kinematic process model used in the EKF is defined as

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) = \begin{bmatrix} E_k + \Delta t \dot{E}_k + \frac{1}{2} \Delta t^2 \ddot{E}_k \\ N_k + \Delta t \dot{N}_k + \frac{1}{2} \Delta t^2 \ddot{N}_k \\ \dot{E}_k + \Delta t \ddot{E}_k \\ \dot{N}_k + \Delta t \ddot{N}_k \\ \psi_k + r_k \Delta t \end{bmatrix} \quad (3.39)$$

Finally, the state estimate updates according to Equations 2.11–2.17, with data from the GNSS as measurement data in z_k .

3.3.3 Path-Planner

The Path-Planner generates a sequence of rotations and straight-line trajectories. The vessel first rotates to face a position immediately outside the dock and then travels in a straight line to this waypoint. Upon arrival, it rotates to orient its aft toward the dock and finally reverses in a straight line to reach the destination. This means the Path-Planner cannot avoid static or dynamic obstacles and that if its start position is at the dock, it still drives to a position outside the dock before reversing back in.

The path consists of discrete points separated by a time step of dt seconds. Each point features five states: x and y positions, surge velocity, yaw angle, and yaw rate. This trajectory serves as a reference for these five states. Consequently, the

integral of the planned surge velocity must equal the total distance traveled, and the integral of the yaw rate must equal the total rotation. A max velocity of 1 m/s is enforced and all accelerations and de-accelerations to and from this velocity is made at a constant acceleration of 0.12 m/s^2 acceleration and 0.08 m/s^2 for retardation. These values are tunable.

3.3.4 MPC

For control, the Path-Planner reference is transformed to align with the state-vector $\boldsymbol{\eta}$ as described in Equation 3.21. This is done by expanding the 5-state Path-Planner reference to include the sway-velocity and setting it to a constant 0.

Running at 5 Hz and guided by references from the Path-Planner, the linearized MPC optimizes control inputs to maintain an accurate course. While the Path-Planner generates kinematic trajectories subject to conservative geometric constraints, it does not account for vehicle dynamics or guarantee dynamic feasibility. The MPC translates these kinematic references into control signals that satisfy the dynamics and operational constraints of the ferry. Furthermore, tracking discrete reference points reduces the computational complexity of the MPC, as it avoids optimizing the entire global trajectory over a single horizon. This enables prediction horizons that are long enough for the ferry to anticipate upcoming trajectory changes, yet short enough to ensure accurate tracking. Assigning a heavy weight to the terminal state of each horizon ensures that the MPC enforces strict compliance with the reference path.

3.3.5 Condensed form MPC

Because the path planner reference can be trimmed into a localized reference windows, the MPC is formulated in a condensed form. This formulation restricts the optimization variables solely to the control input sequence. While the condensed form generally executes slower than the uncondensed version for long horizons, since the computational complexity of the uncondensed form scales linearly with horizon length [20], it remains highly efficient for short horizons absent state variable constraints. The condensed MPC prediction model is formulated as

$$\begin{aligned}
 \hat{\boldsymbol{\eta}}_{(1:N)} = & \underbrace{\begin{bmatrix} \mathbf{A}_d \\ \mathbf{A}_d^2 \\ \vdots \\ \mathbf{A}_d^N \end{bmatrix}}_{\boldsymbol{\Phi}} \boldsymbol{\eta}(0) + \underbrace{\begin{bmatrix} \mathbf{B}_d & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{A}_d \mathbf{B}_d & \mathbf{B}_d & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{A}_d^2 \mathbf{B}_d & \mathbf{A}_d \mathbf{B}_d & \mathbf{B}_d & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_d^{N-1} \mathbf{B}_d & \mathbf{A}_d^{N-2} \mathbf{B}_d & \mathbf{A}_d^{N-3} \mathbf{B}_d & \cdots & \mathbf{B}_d \end{bmatrix}}_{\boldsymbol{\Gamma}} \underbrace{\begin{bmatrix} \boldsymbol{\tau}^{(0)} \\ \boldsymbol{\tau}^{(1)} \\ \vdots \\ \boldsymbol{\tau}^{(N-1)} \end{bmatrix}}_{\mathbf{U}} \\
 & + \underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{A}_d & \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{A}_d^2 & \mathbf{A}_d & \mathbf{I} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_d^{N-1} & \mathbf{A}_d^{N-2} & \mathbf{A}_d^{N-3} & \cdots & \mathbf{I} \end{bmatrix}}_{\boldsymbol{\mathcal{G}}} \begin{bmatrix} \mathbf{g}_d \\ \mathbf{g}_d \\ \vdots \\ \mathbf{g}_d \end{bmatrix}.
 \end{aligned} \tag{3.40}$$

Introducing a trajectory error variable and its input-independent component, the MPC cost function is expressed as

$$J_{track} = \mathbf{e}^\top \bar{\mathbf{Q}} \mathbf{e} + \mathbf{U}^\top \bar{\mathbf{R}} \mathbf{U} \tag{3.41}$$

$$\mathbf{e}_0 = \boldsymbol{\Phi} \boldsymbol{\eta}(0) + \boldsymbol{\mathcal{G}} \mathbf{g}_d - \boldsymbol{\eta}_{ref} \tag{3.42}$$

where

$$\mathbf{e} = \mathbf{e}_0 + \boldsymbol{\Gamma} \mathbf{U}. \tag{3.43}$$

Substituting $\mathbf{e} = \mathbf{e}_0 + \boldsymbol{\Gamma} \mathbf{U}$ and expanding Equation 3.42 gives

$$J_{track} = \mathbf{e}_0^\top \bar{\mathbf{Q}} \mathbf{e}_0 + 2\mathbf{U}^\top \boldsymbol{\Gamma}^\top \bar{\mathbf{Q}} \mathbf{e}_0 + \mathbf{U}^\top \boldsymbol{\Gamma}^\top \bar{\mathbf{Q}} \boldsymbol{\Gamma} \mathbf{U} + \mathbf{U}^\top \bar{\mathbf{R}} \mathbf{U}. \tag{3.44}$$

Discarding the constant term $\mathbf{e}_0^\top \bar{\mathbf{Q}} \mathbf{e}_0$ yields

$$J_{track} = \mathbf{U}^\top (\boldsymbol{\Gamma}^\top \bar{\mathbf{Q}} \boldsymbol{\Gamma} + \bar{\mathbf{R}}) \mathbf{U} + 2\mathbf{U}^\top \boldsymbol{\Gamma}^\top \bar{\mathbf{Q}} \mathbf{e}_0, \tag{3.45}$$

where $\bar{\mathbf{R}}$ penalizes the control sequence $\mathbf{U}$, and $\bar{\mathbf{Q}}$ penalizes state deviations from the reference trajectory. Furthermore,

$$\bar{\mathbf{Q}} = \text{blkdiag}(\mathbf{Q}, \mathbf{Q}, \dots, \mathbf{P}_f) \tag{3.46}$$

$$\bar{\mathbf{R}} = \text{blkdiag}(\mathbf{R}, \dots, \mathbf{R}) \tag{3.47}$$

where $\mathbf{P}_f$ is the terminal state cost penalty.

Because the controller operates on a physical, passenger-carrying electric ferry, large control signal fluctuations are undesirable. Rapid oscillations produce jerky motions that degrade passenger comfort. To mitigate this, an additional cost term penalizes control rate-of-change. This penalty smooths the vehicle response and reduces mechanical stress on the motors. Furthermore, the internal predictive model does not account for actuator dynamics, such as physical limits on thruster rotation speeds.

Because the MPC only commands generalized forces rather than explicit actuator angles, penalizing rapid control variations ensures that the commanded forces vary slowly enough for the physical thrusters to track them. This rate-of-change cost term is expressed as

$$\Delta \mathbf{U} = \begin{bmatrix} \boldsymbol{\tau}(0) - \boldsymbol{\tau}_{prev} \\ \boldsymbol{\tau}(1) - \boldsymbol{\tau}(0) \\ \boldsymbol{\tau}(2) - \boldsymbol{\tau}(1) \\ \vdots \\ \boldsymbol{\tau}(N-1) - \boldsymbol{\tau}(N-2) \end{bmatrix} = \mathbf{D}_\Delta \mathbf{U} - \underbrace{\begin{bmatrix} \boldsymbol{\tau}_{prev} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}}_{\mathbf{u}_0} \quad (3.48)$$

where $\boldsymbol{\tau}_{prev}$ is the previous applied control input and

$$\mathbf{D}_\Delta = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ -\mathbf{I} & \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & -\mathbf{I} & \mathbf{I} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & -\mathbf{I} & \mathbf{I} \end{bmatrix}. \quad (3.49)$$

The additional cost term is

$$J_{\Delta U} = (\mathbf{D}_\Delta \mathbf{U} - \mathbf{u}_0)^\top \bar{\mathbf{S}} (\mathbf{D}_\Delta \mathbf{U} - \mathbf{u}_0) \quad (3.50)$$

$$= \mathbf{U}^\top \mathbf{D}_\Delta^\top \bar{\mathbf{S}} \mathbf{D}_\Delta \mathbf{U} - 2\mathbf{u}_0^\top \bar{\mathbf{S}} \mathbf{D}_\Delta \mathbf{U} + \mathbf{u}_0^\top \bar{\mathbf{S}} \mathbf{u}_0. \quad (3.51)$$

Removing the constant term, which does not influence the optimal solution, yields

$$J_{\Delta U} = \mathbf{U}^\top \mathbf{D}_\Delta^\top \bar{\mathbf{S}} \mathbf{D}_\Delta \mathbf{U} - 2\mathbf{u}_0^\top \bar{\mathbf{S}} \mathbf{D}_\Delta \mathbf{U}, \quad (3.52)$$

where

$$\bar{\mathbf{S}} = \text{blkdiag}(\mathbf{S}, \dots, \mathbf{S}). \quad (3.53)$$

The total cost function is therefore

$$J_{tot} = J_{track} + J_{\Delta U}, \quad (3.54)$$

which is structured for compatibility with a quadratic programming (QP) solver. Furthermore, the control inputs are bounded by physical force limits at each time step:

$$-\boldsymbol{\tau}_{max} \leq \boldsymbol{\tau}(k) \leq \boldsymbol{\tau}_{max}, \quad k = 0, 1, \dots, N-1, \quad (3.55)$$

where $\boldsymbol{\tau}_{max} = [X_{max} \ Y_{max} \ N_{max}]^\top$ aggregates the surge force, sway force, and yaw moment limits. Stacking these box constraints forms a pair of linear inequalities:

$$\underbrace{\begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \end{bmatrix}}_{\mathbf{A}_1} \boldsymbol{\tau}(k) \leq \underbrace{\begin{bmatrix} \boldsymbol{\tau}_{max} \\ \boldsymbol{\tau}_{max} \end{bmatrix}}_{\mathbf{b}_1}. \quad (3.56)$$

Stacking these constraints over the entire prediction horizon yields the following form:

$$\mathbf{A}_{ineq} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_1 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_1 \end{bmatrix}, \quad \mathbf{b}_{ineq} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_1 \end{bmatrix}. \quad (3.57)$$

These constraints only bound the generalized forces (surge, sway, and yaw moments), meaning no explicit state constraints are implemented. For urban operations, the Cstrider ferry must not exceed a speed limit of 5 knots. This constraint is implicitly enforced by choosing conservative bounds for τ_{max} and speed limits in the Path-Planner. The specific values for τ_{max} can be found in Equation 3.69.

The complete condensed quadratic programming problem is formulated as

$$\begin{aligned} \min_{\mathbf{U}} \quad & \frac{1}{2} \mathbf{U}^\top \mathbf{H} \mathbf{U} + \mathbf{f}^\top \mathbf{U} \\ \text{subject to} \quad & \mathbf{A}_{ineq} \mathbf{U} \leq \mathbf{b}_{ineq}, \end{aligned} \quad (3.58)$$

where

$$\mathbf{H} = 2 \left(\mathbf{\Gamma}^\top \bar{\mathbf{Q}} \mathbf{\Gamma} + \bar{\mathbf{R}} + \mathbf{D}_\Delta^\top \bar{\mathbf{S}} \mathbf{D}_\Delta \right), \quad (3.59)$$

$$\mathbf{f} = 2 \left(\mathbf{\Gamma}^\top \bar{\mathbf{Q}} \mathbf{e}_0 - \mathbf{D}_\Delta^\top \bar{\mathbf{S}} \mathbf{u}_0 \right). \quad (3.60)$$

3.3.6 OSQP Solver

The Operator Splitting Quadratic Program (OSQP) solver executes the MPC optimization. OSQP is highly suited for linear systems as it efficiently solves convex quadratic programs [21]. Because the underlying system dynamics are linearized, the resulting optimization problem remains convex. Additionally, OSQP features native Python compatibility, making it highly compatible with the ROS2 implementation.

3.3.7 LQR

The low-frequency MPC optimizes a control sequence based on the initial state estimate. However, several simplifications were made during the modeling of the ferry, as explained in Section 3.2.1). This creates an inherent mismatch between the model used by the MPC and the actual vessel dynamics. Furthermore, the system is linearized only once per optimization cycle. A first order linearization is only exact at the linearization point. Outside of this point, the linearization carries an error, where the nonlinear dynamics no longer matches the linearized plant [19, p. 549]. This error is the motivation for the linear time-varying MPC scheme, which re-linearize around the previous predicted state at each step of the horizon to limit an error that otherwise grows as the prediction deviates from the linearization point [23, p. 223-228]. The existence of this scheme indicate that the single first-order linearization (see Equation 3.23) will result in significant model-error as the prediction approaches the terminal state. This is however, even more computationally heavy than normal

MPC. In the docking scenario, the ferry's velocity and heading evolve substantially over the maneuver, so the predicted state moves far from the linearization point and the prediction accuracy degrades accordingly across the horizon.

Because the MPC optimizes inputs against state predictions of decaying quality, the resulting control-sequence is optimal for the linearized prediction model given its initial operating point, but suboptimal for the true nonlinear plant. However, the first control-inputs and predictions of the sequence are the least corrupted, since they are computed in closest proximity to the linearization point. Furthermore, unmodeled external disturbances, such as wind, waves and currents also cause the vessel to drift from both its predicted trajectory and the global path reference. A low-frequency, 5 Hz, MPC loop could be insufficient to reject such rapidly varying disturbances.

Inspired by "Tube-MPC" , a high-frequency, 100 Hz, Linear Quadratic Regulator (LQR) operates in parallel to the MPC [24]. The LQR is designed to minimize the deviation of the actual state from the MPC's predicted reference trajectory. However, since the MPC is running at a lower frequency and generates a state prediction per horizon step, the prediction has to be scaled such that it fits the frequency of the LQR. By interpolating between the current and predicted states, the controller generates a reference for the LQR. The MPC-LQR control law can thus be expressed as

$$\boldsymbol{\tau}_{des} = \boldsymbol{\tau}_{mpc} + \boldsymbol{\tau}_{lqr}, \quad (3.61)$$

where

$$\boldsymbol{\tau}_{lqr} = -\mathbf{K}_{lqr}(\boldsymbol{\eta}_{interpolated} - \boldsymbol{\eta}). \quad (3.62)$$

While the MPC forces ($\boldsymbol{\tau}_{mpc}$) are bounded via optimization constraints, the combined control command ($\boldsymbol{\tau}_{des}$) must remain feasible. The total command is therefore saturated using the same force and moment limits as the MPC. Consequently, the LQR correction is realized only when the MPC command leaves headroom within the saturation limits. If the MPC is already saturated, the LQR contribution is clipped.

3.3.8 Thrust-Allocator

The MPC and LQR produce a desired generalized force vector, the function of the Thrust-Allocator is to translate this into forces and angles for each motor, as illustrated in Figure 3.7. The ferry features two azimuth thrusters with a steering angle range of ± 45 degrees. The motors provide bidirectional thrust (forward and backward), enabling differential thrust capabilities.

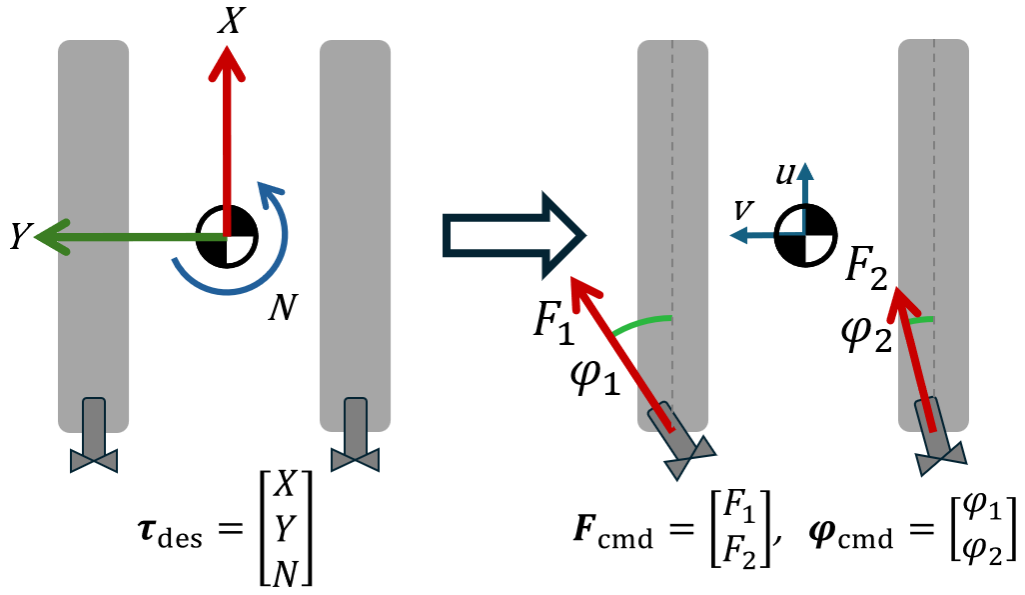


Figure 3.7: Conceptual illustration of the function of the Thrust-Allocator

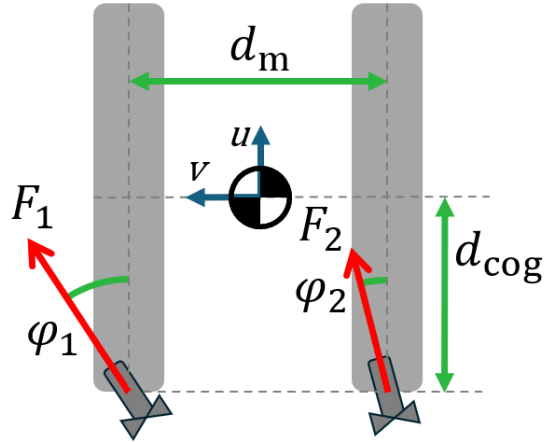


Figure 3.8: Geometric setup of the thrusters on a dual azimuth thruster catamaran, as Celia One.

Figure 3.8 illustrates this setup where φ_i denotes the steering angle for each thruster and F_i represents the generated thrust. The parameters d_{cog} and d_m define the distance from the center of gravity to the thrusters and the distance between the two thrusters, respectively. Under a coordinate system where u represents surge velocity and v represents sway velocity, the resulting generalized forces and moments $\boldsymbol{\tau}$ are modeled as

$$\begin{aligned}
\boldsymbol{\tau} = \begin{bmatrix} X \\ Y \\ N \end{bmatrix} &= \begin{bmatrix} F_1 c(\varphi_1) + F_2 c(\varphi_2) \\ F_1 s(\varphi_1) + F_2 s(\varphi_2) \\ \frac{d_m}{2} \left(-F_1 c(\varphi_1) + F_2 c(\varphi_2) \right) + d_{\text{cog}} \left(-F_1 s(\varphi_1) - F_2 s(\varphi_2) \right) \end{bmatrix} \\
&= \begin{bmatrix} c(\varphi_1) & c(\varphi_2) \\ s(\varphi_1) & s(\varphi_2) \\ -\frac{d_m}{2} c(\varphi_1) - d_{\text{cog}} s(\varphi_1) & \frac{d_m}{2} c(\varphi_2) - d_{\text{cog}} s(\varphi_2) \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \end{bmatrix} = \mathbf{T}(\boldsymbol{\varphi}) \mathbf{F} \quad (3.63)
\end{aligned}$$

where $s(\cdot) = \sin(\cdot)$ and $c(\cdot) = \cos(\cdot)$. The high-level controller commands a desired generalized force vector $\boldsymbol{\tau}_{\text{des}}$, which the thrust allocator maps into individual thrust commands $\mathbf{F}$ and steering angles $\boldsymbol{\varphi}$. This allocation is challenging because $\mathbf{F}$, $\boldsymbol{\varphi}$, and the steering rate $\dot{\boldsymbol{\varphi}}$ are bounded. Furthermore, the thruster configuration matrix $\mathbf{T}(\boldsymbol{\varphi})$ can encounter singular configurations where its pseudoinverse is undefined. Fossen [17, pp. 408-410] frames this allocation as an optimization problem, customized for this system as follows:

$$J = \min_{\mathbf{F}, \boldsymbol{\varphi}, \mathbf{s}} \left\{ \bar{P} |F_1|^{3/2} + \bar{P} |F_2|^{3/2} + \mathbf{s}^\top \mathbf{Q} \mathbf{s} + (\boldsymbol{\varphi} - \boldsymbol{\varphi}_{\text{prev}})^\top \boldsymbol{\Omega} (\boldsymbol{\varphi} - \boldsymbol{\varphi}_{\text{prev}}) + \frac{\varrho}{\varepsilon + \det(\mathbf{T}(\boldsymbol{\varphi}) \mathbf{W}^{-1} \mathbf{T}^\top(\boldsymbol{\varphi}))} \right\} \quad (3.64)$$

subject to:

$$\begin{aligned}
\mathbf{T}(\boldsymbol{\varphi}) \mathbf{F} &= \boldsymbol{\tau}_{\text{des}} + \mathbf{s} \\
-\mathbf{F}_{\text{max}} &\leq \mathbf{F} \leq \mathbf{F}_{\text{max}} \\
-\boldsymbol{\varphi}_{\text{max}} &\leq \boldsymbol{\varphi} \leq \boldsymbol{\varphi}_{\text{max}} \\
-\Delta\boldsymbol{\varphi}_{\text{max}} &\leq \boldsymbol{\varphi} - \boldsymbol{\varphi}_{\text{prev}} \leq \Delta\boldsymbol{\varphi}_{\text{max}}
\end{aligned} \quad (3.65)$$

where

- $\bar{P} |F_i|^{3/2}$ penalizes power consumption, where $\bar{P} > 0$ is a positive weighting scalar.
- $\mathbf{s}^\top \mathbf{Q} \mathbf{s}$ penalizes the slack variable error $\mathbf{s}$ between the commanded forces $\boldsymbol{\tau}_{\text{cmd}}$ and the desired forces $\boldsymbol{\tau}_{\text{des}}$. This ensures that the optimization problem retains a feasible solution for any combination of $\boldsymbol{\tau}$ and $\boldsymbol{\varphi}_{\text{prev}}$. The weight matrix $\mathbf{Q} > \mathbf{0}$ is selected sufficiently large to drive $\mathbf{s} \approx \mathbf{0}$ whenever feasible.
- The term

$$\frac{\varrho}{\varepsilon + \det(\mathbf{T}(\boldsymbol{\varphi}) \mathbf{W}^{-1} \mathbf{T}^\top(\boldsymbol{\varphi}))}$$

prevents singular thruster configurations where $\det(\mathbf{T}(\boldsymbol{\varphi}) \mathbf{W}^{-1} \mathbf{T}^\top(\boldsymbol{\varphi})) = 0$. $\mathbf{W}$ is a weighting matrix, in this project chosen to be $\mathbf{I}_{2 \times 2}$. To prevent division by zero, $\varepsilon > 0$ acts as a small regularization parameter, while $\varrho > 0$ scales

the penalty. A larger value of ϱ prioritizes maneuverability at the expense of power efficiency.

- $-\mathbf{F}_{\max} \leq \mathbf{F} \leq \mathbf{F}_{\max}$ defines the actuator saturation bounds.
- $-\varphi_{\max} \leq \varphi \leq \varphi_{\max}$ enforces the allowable angular sector limits for the azimuth thrusters.
- $-\Delta\varphi_{\max} \leq \varphi - \varphi_{\text{prev}} \leq \Delta\varphi_{\max}$ restricts the maximum angular change per sample step, which implicitly bounds the thruster turning rate $\dot{\varphi}$. Here, φ_{prev} represents the steering angles from the previous control cycle.

The optimization problem is presented in its non-linear form for conceptual clarity, though the real-time controller implements a linearized version. This linearized formulation iteratively computes the incremental adjustments $\Delta\mathbf{F}$ and $\Delta\varphi$. Consequently, the controller tracks absolute actuator states across iterations by accumulating these deltas onto the previous state values. The linearized optimization problem is formulated as:

$$J_{\text{lin}} = \min_{\Delta\mathbf{F}, \Delta\varphi, \mathbf{s}} \left\{ (\mathbf{F}_{\text{prev}} + \Delta\mathbf{F})^\top \mathbf{P} (\mathbf{F}_{\text{prev}} + \Delta\mathbf{F}) + \mathbf{s}^\top \mathbf{Q} \mathbf{s} + \Delta\varphi^\top \Omega \Delta\varphi + \frac{\partial}{\partial \varphi} \left(\frac{\varrho}{\varepsilon + \det(\mathbf{T}(\varphi) \mathbf{W}^{-1} \mathbf{T}^\top(\varphi))} \right) \bigg|_{\varphi_{\text{prev}}} \Delta\varphi \right\} \quad (3.66)$$

subject to:

$$\begin{aligned} & \mathbf{T}(\varphi_{\text{prev}}) \mathbf{F}_{\text{prev}} + \mathbf{T}(\varphi_{\text{prev}}) \Delta\mathbf{F} + \frac{\partial}{\partial \varphi} (\mathbf{T}(\varphi) \mathbf{F}) \bigg|_{\substack{\varphi = \varphi_{\text{prev}} \\ \mathbf{F} = \mathbf{F}_{\text{prev}}}} \Delta\varphi = \boldsymbol{\tau}_{\text{des}} + \mathbf{s} \\ & -\mathbf{F}_{\max} - \mathbf{F}_{\text{prev}} \leq \Delta\mathbf{F} \leq \mathbf{F}_{\max} - \mathbf{F}_{\text{prev}} \\ & -\varphi_{\max} - \varphi_{\text{prev}} \leq \Delta\varphi \leq \varphi_{\max} - \varphi_{\text{prev}} \\ & -\Delta\varphi_{\max} \leq \Delta\varphi \leq \Delta\varphi_{\max} \end{aligned} \quad (3.67)$$

The solutions $\Delta\mathbf{F}$ and $\Delta\varphi$ that minimize J_{lin} , and the slack vector $\mathbf{s}$, are added to the previous states $\mathbf{F}_{\text{prev}}$ and φ_{prev} . The resulting absolute variables are denoted as $\mathbf{F}_{\text{cmd}}$ and φ_{cmd} .

The limitations used in Equation 3.65 and 3.67 and presented below represent approximate physical limitations observed for Celia One:

$$\begin{aligned} \mathbf{F}_{\max} &= \begin{bmatrix} 600 \text{ N} \\ 600 \text{ N} \end{bmatrix} \\ \varphi_{\max} &= \begin{bmatrix} 45^\circ \\ 45^\circ \end{bmatrix} \\ \Delta\varphi_{\max} &= \begin{bmatrix} \Delta t \text{ } 18^\circ/\text{s} \\ \Delta t \text{ } 18^\circ/\text{s} \end{bmatrix} \end{aligned} \quad (3.68)$$

From the geometry in Figure 3.8 and values in Equation 3.68, the input constraints of the MPC from Equation 3.55) is defined as:

$$\boldsymbol{\tau}_{max} = \begin{bmatrix} 2 \times 600 \\ 2 \times 600 \times \sin(45^\circ) \\ \frac{d_m}{2} \times 2 \times 600 \cos(45^\circ) + d_{cog} \times 2 \times 600 \sin(45^\circ) \end{bmatrix} = \begin{bmatrix} 1200 \text{ N} \\ 849 \text{ N} \\ 2860 \text{ Nm} \end{bmatrix} \quad (3.69)$$

where $d_{cog} = 2.37 \text{ m}$ and $d_m = 2 \text{ m}$ is the geometry of the WAMV in VRX.

However, because the physical motors accept control commands rather than direct Newton force values, a force-to-command conversion is required (see Section 3.2.5.2). This mapping executes for each motor by inputting the commanded thrust $F_{i,\text{cmd}}$ and the estimated water flow rate across the propeller $V_{i,\text{prop}}$ into an empirical polynomial:

$$\text{ThrustCMD} = \text{Poly}(F_{i,\text{cmd}}, V_{i,\text{prop}}). \quad (3.70)$$

This polynomial mapping is derived in Section 4.1.2. The fluid velocity at the propeller disk is estimated using

$$V_{i,\text{prop}} \approx u \cos(\varphi_{i,\text{cmd}}) + (v + r d_{cog}) \sin(\varphi_{i,\text{cmd}}) \quad (3.71)$$

where u , v , and r denote the surge, sway, and yaw rate states. Finally, ThrustCMD and φ_{cmd} are sent to the motors.

3.4 Simulation in VRX

VRX provides a default vehicle model named the Wave Adaptive Modular Vessel (WAMV) [7]. The simulation's hydrodynamic coefficients are pre-tuned for this catamaran, which measures 4 m in length, 2.5 m in width, and weighs 180 kg. While importing a custom hull geometry and unique hydrodynamic parameters is supported, implementation is technically complex. Because the tracking controller scales to Celia One through basic parametric re-tuning, deploying a custom model within the simulator is unnecessary. To increase inertia, the simulated WAMV mass is increased to 250 kg. Consequently, the controller is tuned for this 250 kg configuration. All simulated sensors operate within either an East-North-Up (ENU) world frame or a Forward-Left-Up (FLU) body frame.

3.5 Docking Tests in VRX

The docking sequence is evaluated under various environmental configurations to compare performance between the standalone MPC and the integrated MPC-LQR control architectures. Performance is analyzed from two distinct initial vessel poses. The simulated environmental disturbances consist of ambient wind and wave fields. The docking sequence utilizes the waypoints shown in Table 3.1.

where,

$$\boldsymbol{\eta}_{pose} = \begin{bmatrix} E & N & \psi \end{bmatrix}. \quad (3.72)$$

Table 3.1: Differing initial positions for the ferry docking sequence

	Initial Pose	Twist Point	Docking Pose
Sequence 1	[39.20 -38.34 0.99]	[15.16 0.57 0.079]	[0.20 -0.61, 0.079]
Sequence 2	[42.02, 40.8 0.3]	[15.16, 0.57, 0.079]	[0.20, -0.61, 0.079]

The units of the pose are meters for the position and radians for the heading angle. During the docking maneuvers, the vessel encounters varying magnitudes of environmental disturbances. Four distinct test scenarios are evaluated for each controller configuration, as detailed in Table 3.2, where the variance parameters define the bounds of stochastic wind gusts. To eliminate stochastic bias and ensure fair benchmarking, a fixed random seed enforces identical environmental conditions across comparative evaluation runs. This approach isolates performance metrics from environmental variability. Furthermore, the angle of attack for the wind is constant at 240 degrees and the wave period is 2 seconds. Note that all scenarios still have sensor noise, model error, etc.

Table 3.2: Environmental Parameters by Scenario

	Wind mean [m/s]	Wind Variance [m/s]	Waves [m]
Scenario 1	0.0	0.0	0.0
Scenario 2	4.0	1.5	0.5
Scenario 3	7.0	5.0	1.5
Scenario 4	11.0	4.0	0.0

4

Results and Discussion

This chapter presents the controller testing results, along with the parameter estimations for the hydrodynamical and motor models. It also comments on these results and estimations.

4.1 System Identification

This section presents the data from each system identification experiment alongside the estimated parameter values. Section 3.2 details the experimental setup, objectives, and calculation methods used for parameter estimation.

Due to time limitations, physical testing on the actual ferry was not possible; consequently, all experimental data are collected via simulated sensors in VRX. However, all tests and parameter estimation methods are fully compatible with real-world ferry data.

4.1.1 Estimating the Dampening Parameters

Figures 4.1, 4.2, and 4.3 display the data logged from VRX for Experiments 1, 2, and 3. Several data series are overlaid in the figures. The yaw acceleration in Figure 4.3 is derived from the yaw rate, as discussed in Section 3.2.5.1.

Figures 4.4, 4.5, and 4.6 plot this data and the fitted polynomials. For each experiment, the system uses two fits: a linear fit for the controller and a quadratic fit for estimating motor forces. For surge and yaw rate, the weighted fits are preferred because the unweighted ones fail to match data points accurately at high surge or yaw rates. Conversely, for sway, the unweighted fits are selected. This choice is made because sway data becomes less reliable at higher values, and the controller primarily operates at lower sways. Based on these considerations and the graphs in Figure 4.5, the unweighted fits perform better. Table 4.1 presents all estimated parameters alongside the ground truth parameters used by VRX.

4.1.1.1 Comments on the Inaccuracy of the Estimations

As Table 4.1 shows, the estimated quadratic parameters do not perfectly match the VRX parameters. There are a few reasons for this discrepancy. First, the range of

Table 4.1: Estimated parameters for the dampening in the hydrodynamical model along with the ground truth parameters used by VRX

Dampening Parameter	Estimated linear parameter	Estimated quadratic parameter	Quadratic parameter used by VRX
X_u	-124 (with weights)	[-122, -4.34] (with weights)	[-50, -150]
Y_r	-81.8 (without weights)	[-34.8, -104] (without weights)	[-100, -150]
N_r	-736 (with weights)	[-652, -162] (with weights)	[-800, -800]

surges, sways, and yaw rates is quite low. This limitation primarily affects the yaw rate, which peaks just under 0.8 rad/s in the estimation data. At low values, the quadratic term becomes inherently difficult to detect as it becomes small. Second, the data contains significant noise, which is discussed further in the next paragraph. Finally, in noisy data, the curvature produced by the quadratic term is hard to isolate. As a result, the linear term absorbs the added value from the quadratic term, incorrectly inflating the linear estimate.

4.1.1.2 Comments on the Oscillations in the Data

Oscillations plague the data from Experiments 1 and 2 after the motors turn on or off, as Figures 4.1, 4.2, 4.4, and 4.5 illustrate. This issue likely occurs because the water’s dampening force acts below the pitch and roll rotation axes. When the motors deactivate, the ferry’s inertia causes it to lean forward, inducing a pitch or roll oscillation depending on initial direction of travel.

Initially, this effect was most prominent in Experiment 1. To mitigate it, the motors gradually powers down between 0.6s to 0.9s for different runs instead of instantaneously, which significantly reduced the problem. Rerunning Experiment 2 is unnecessary because the dampening parameter is fine-tunable using the method from Section 3.2.5, and the force-sway data in Figure 4.5 remains clear enough within the controller’s operating range.

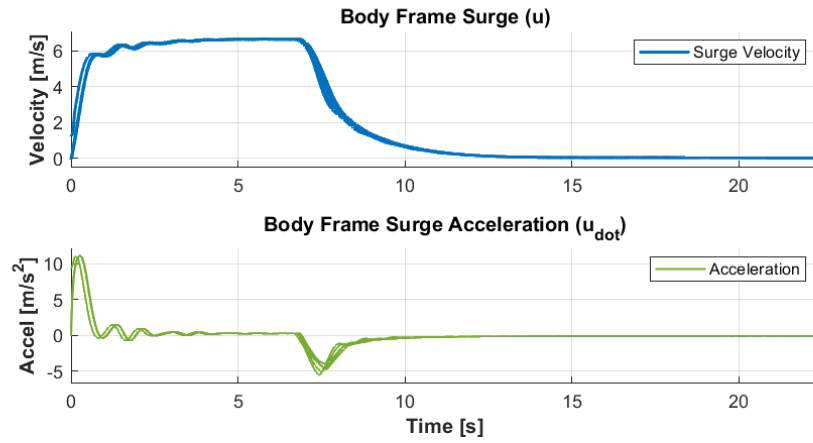


Figure 4.1: Raw surge and surge acceleration data from six runs of Experiment 1.

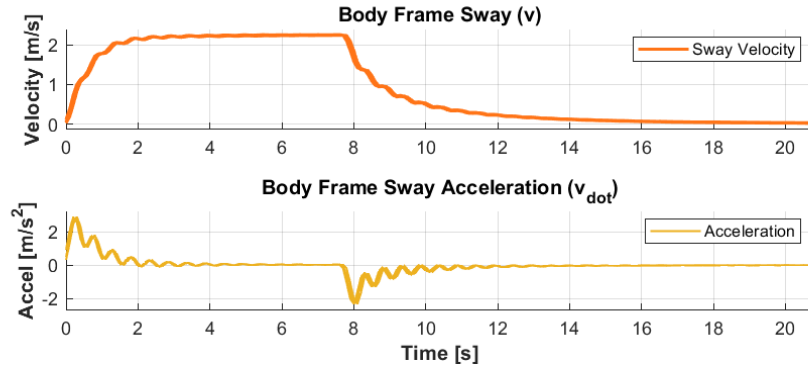


Figure 4.2: Raw sway and sway acceleration data from four runs of Experiment 2.

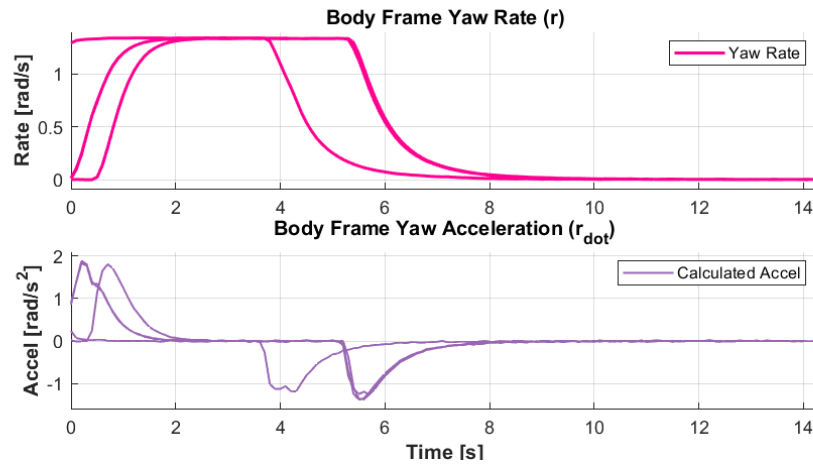


Figure 4.3: Raw yaw rate and calculated yaw acceleration data from four runs of Experiment 3. The data series in this plot has been shifted in time for better visualization.

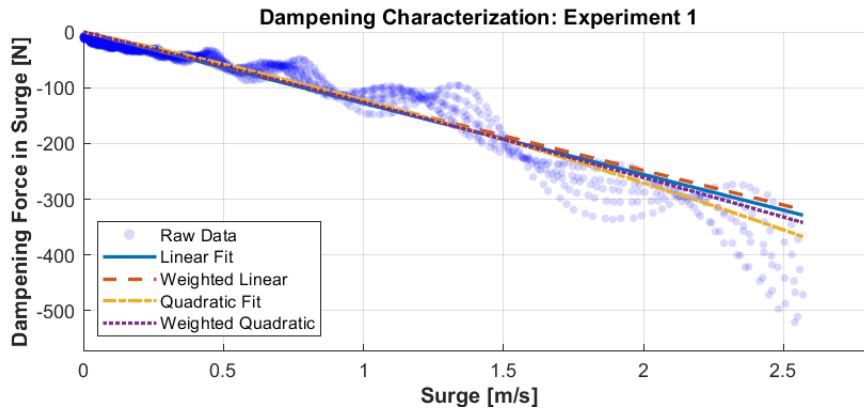


Figure 4.4: Different fits to force-surge data from Experiment 1

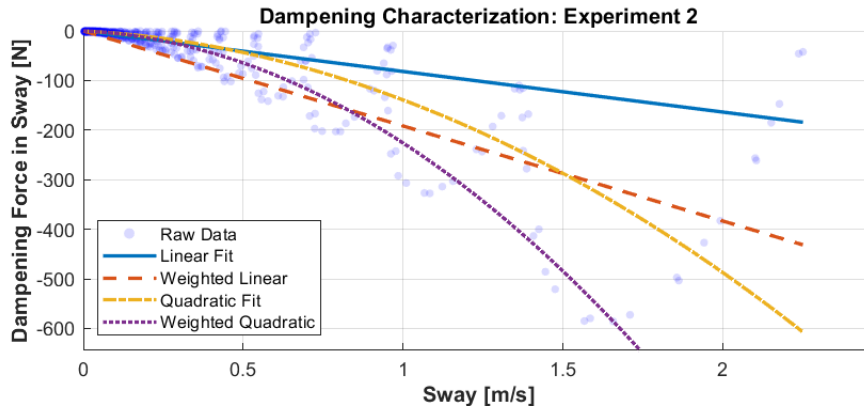


Figure 4.5: Different fits to force-sway data from Experiment 2

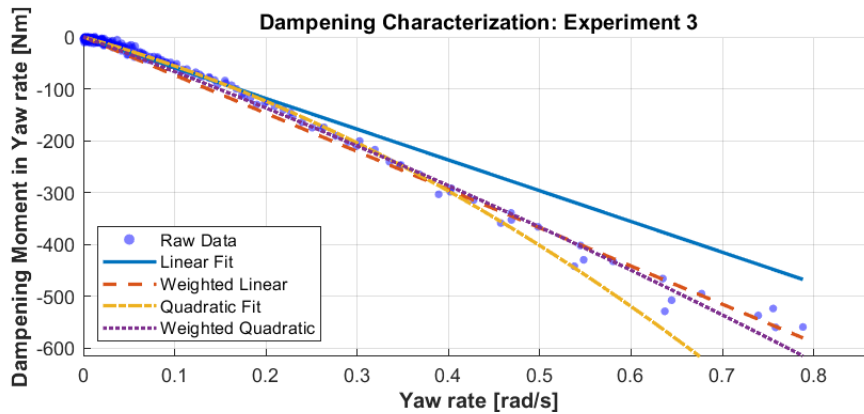


Figure 4.6: Different fits to moment-yaw rate data from Experiment 3

4.1.2 Estimating the Force Produced by the Motors

Figures 4.7, 4.8, and 4.9 display the raw data from three of the four runs in Experiment four. The experimental procedure varies slightly between runs to gather more

data points. The run shown in Figure 4.9 operates in reverse because it was intended to create a separate fit for reverse motor force. However, because it is nearly identical to the forward data, it is included in the main dataset instead. From this data, the algorithm extracts three values at each acceleration peak: acceleration, surge, and thrust command. Section 3.2.5.2 explains how to approximate the motor force at each point using this data. Finally, Figure 4.10 plots a polynomial fitted to these points. The fitted polynomial is

$$f(x, y) = -6.25x + 0.547y + 41.1x^2 + 0.0406xy - 0.0001y^2, \quad (4.1)$$

where x is the surge velocity, y is the requested force, and $f(x, y)$ is the target thrust command.

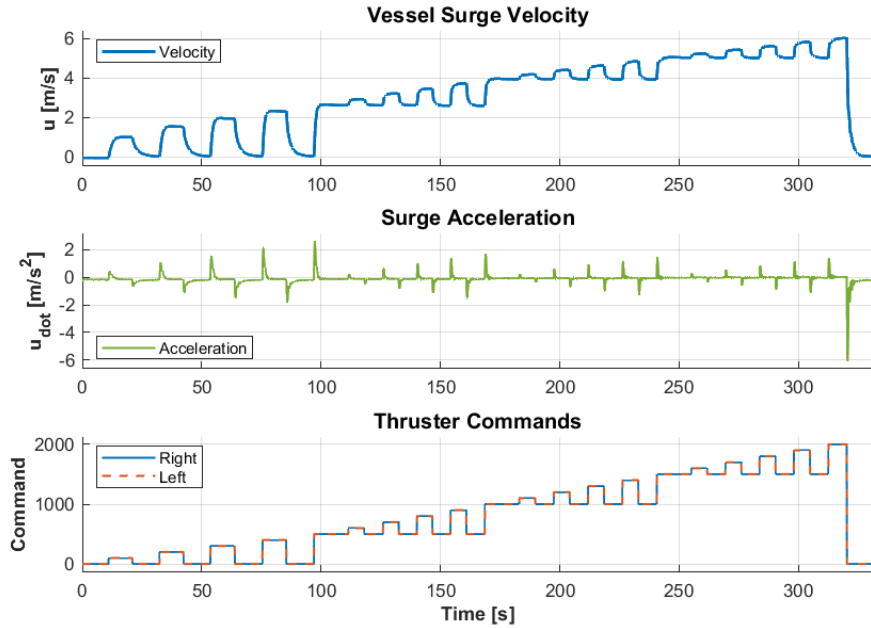


Figure 4.7: Raw data from one of the four runs of Experiment 4.

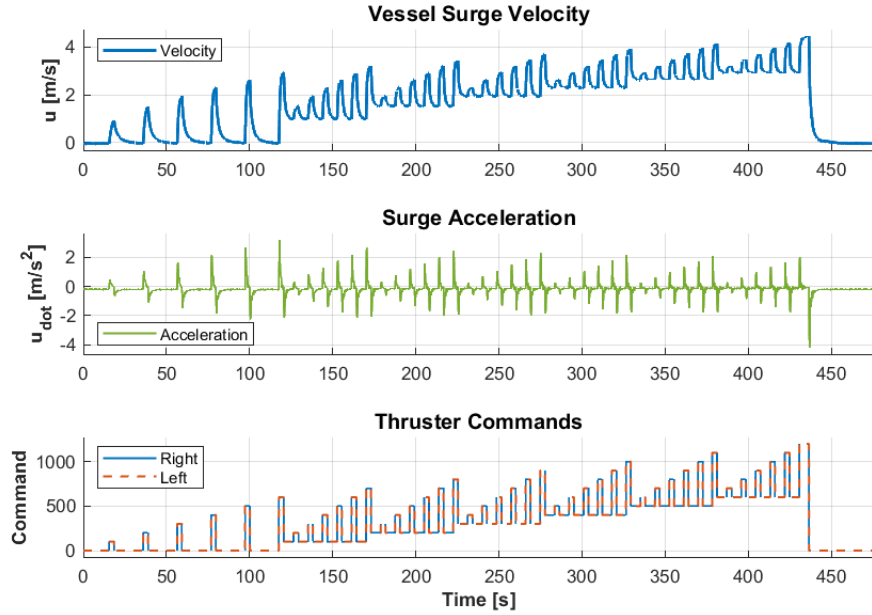


Figure 4.8: Raw data from one of the four runs of Experiment 4, where more steps were taken than was initially planned when the experiment was invented.

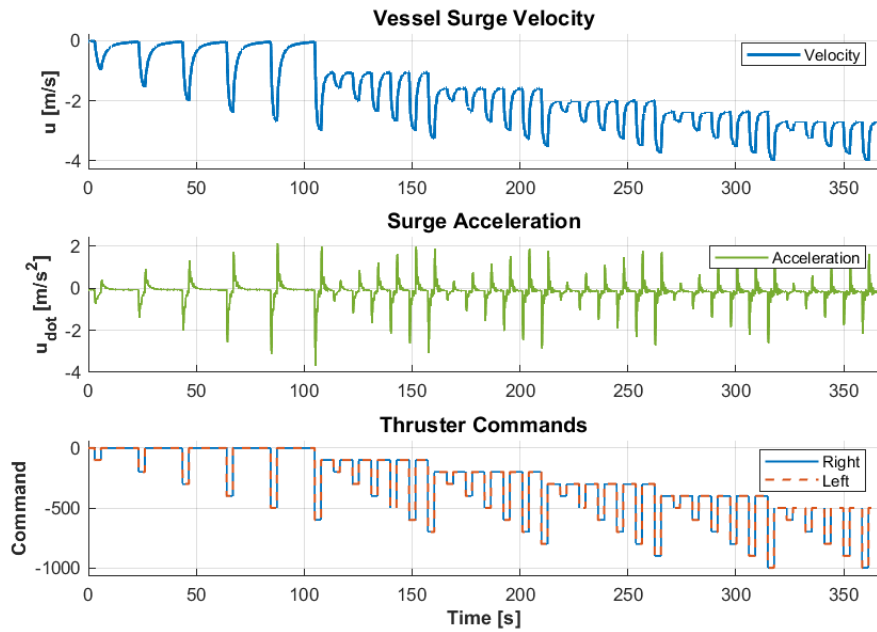


Figure 4.9: Raw data from one of the four runs of Experiment 4, when the ferry was driving backwards instead.

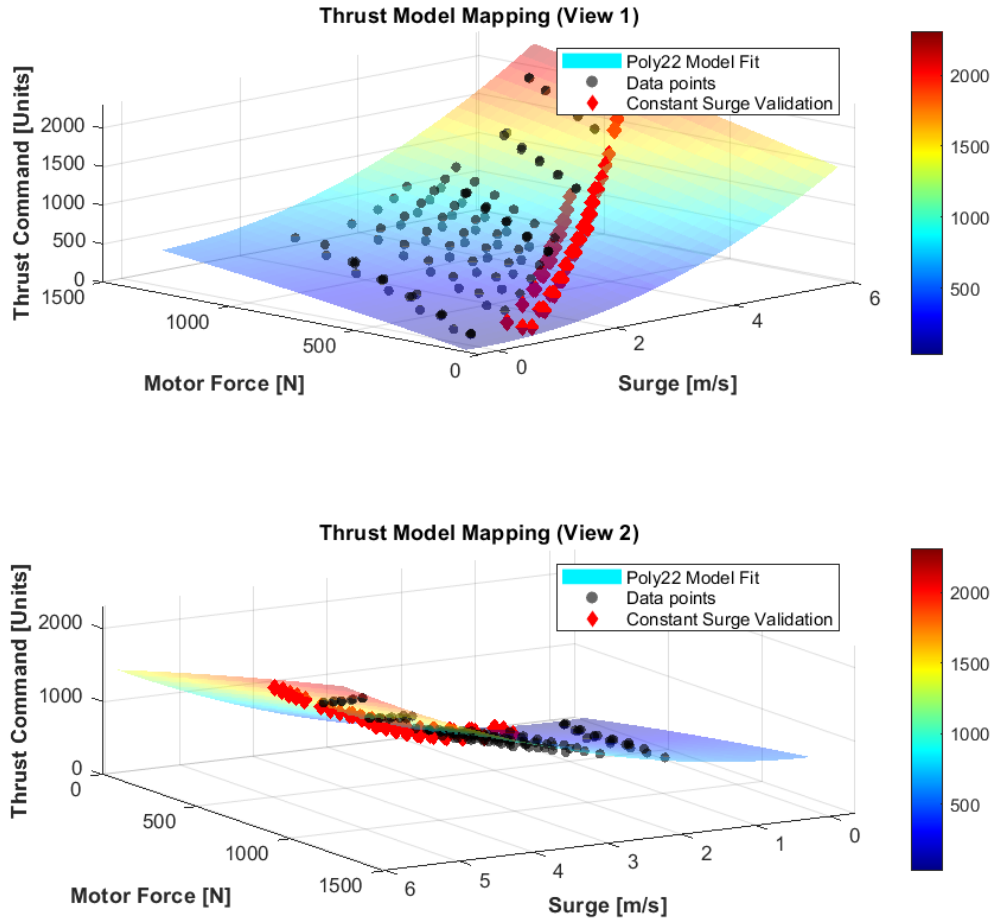


Figure 4.10: The black points are estimated from the Experiment 4 data. The surface is the function estimated from the data. Lastly, the red points are points taken at places in the data with zero acceleration as validation points. View 1 show the graph with the origin in the foreground and the surface sloping upwards as it moves away. View 2 instead, has the origin in the background and the surface is seen decreasing down toward zero as it approaches the origin. View 2 also indicates that the surface is a good estimation of the points since it follows them quite well.

4.2 Subsystems

In this section, the test results of individual subsystems of the controller are presented.

4.2.1 State-Estimator

The EKF performance was evaluated by comparing the estimated states against the VRX ground truth positions and orientations during a docking sequence.

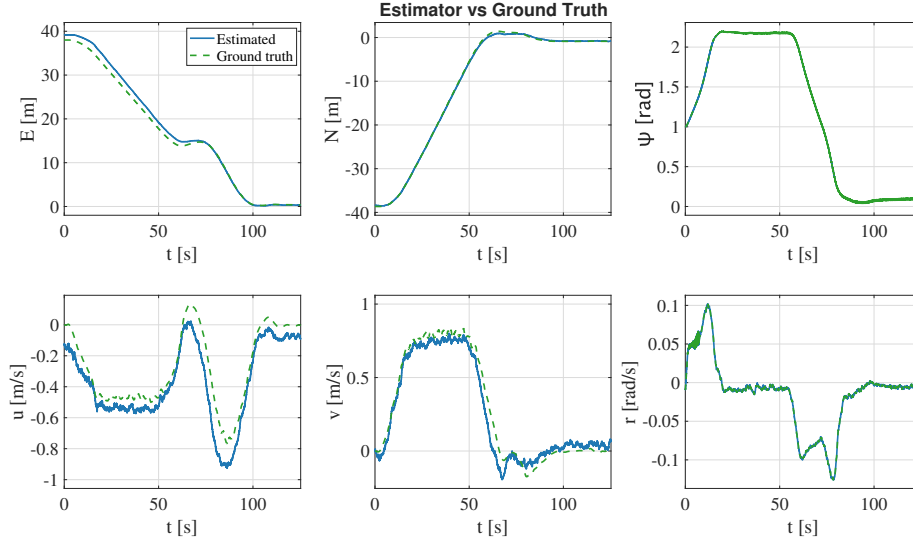


Figure 4.11: The estimated states versus ground truth value during docking Sequence 1, Scenario 1. The initial position pose of the ferry is $\boldsymbol{\eta}_{pose} = [39.2, -38.3, 0.99]$

The ground-truth surge and sway velocities in Figure 4.11 were derived by differentiating ground-truth position measurements and applying a heavy low-pass filter, which introduced some lag. The estimator tracked the ground-truth positions E and N closely. However, a small bias appeared, particularly at the beginning of the trajectory, likely because the filter lacked sufficient time to converge before the sequence began. Furthermore, because the kinematic model in the EKF omitted the ferry's dynamics, its process model predictions were prone to error. Nevertheless, because the measurements remained high quality, these model mismatches did not severely degrade the state estimates.

4.2.2 Path-Planner

In Scenario 1, the Path-Planner produces the path shown in Figures 4.12 and 4.13. The simplicity of this path makes the planner highly robust. The algorithm consistently generates a viable path; however, it does not account for obstacles like other vessels or docks.

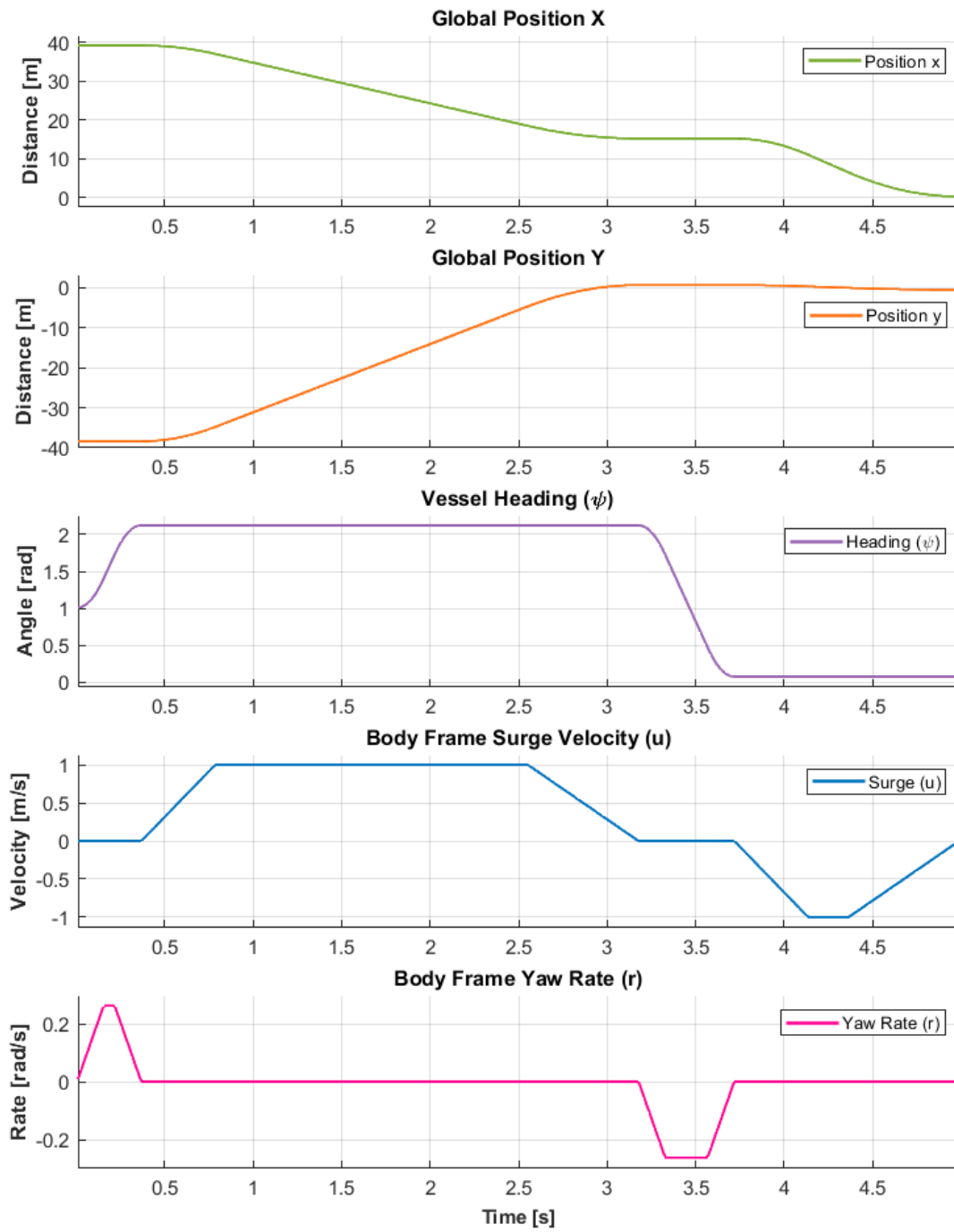


Figure 4.12: The planned path in Scenario 1 with each planned state plotted separately.

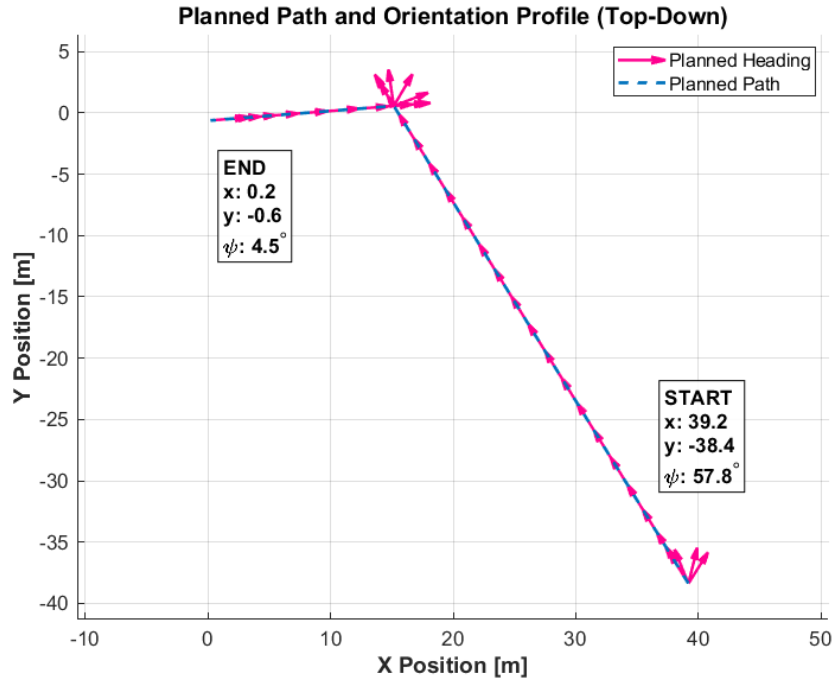
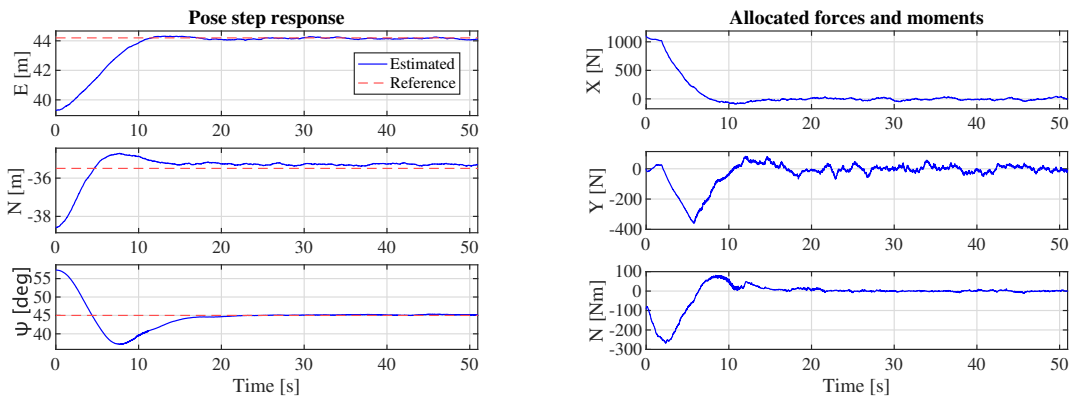


Figure 4.13: The planned path in Scenario 1 from a top view.

4.2.3 LQR

The LQR was evaluated to ensure proper functionality through a step response test, as shown in Figure 4.14.



(a) Step response of the LQR for the crucial docking states η_{pose} .

(b) Allocated forces τ_{cmd} during the performed step response.

Figure 4.14: Figure 4.14a shows a step performance carried out on the ferry in VRX. The EKF estimated states are the blue lines. The dashed red lines denote the reference. Figure 4.14b illustrates the forces that are delivered by the motors via the Thrust-allocator.

Based on Figure 4.14b, the controller responded aggressively to large position errors. The ferry's maximum surge thrust was 1200 N, placing the control output near its saturation limit. Because the LQR was designed to correct rapid disturbances, operational state errors are expected to remain small; therefore, an aggressive response to large errors was an intentional design choice.

Figure 4.14a shows the east position converging smoothly to the reference. The north position overshoot slightly and converged to a steady-state error of approximately 0.4 m. While adding integral action on the pose states could have mitigated this error, the primary objective of the LQR is to stabilize the ferry against rapid disturbances rather than strictly follow the Path-Planner reference. However, incorporating integral action could be a beneficial addition for tracking the MPC prediction, where detailed reference tracking is required.

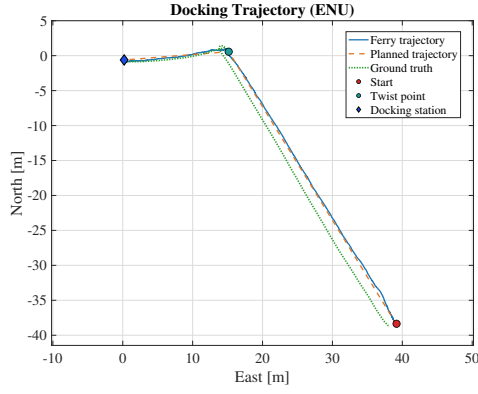
The heading angle ψ converged to the reference without steady-state error, though some overshoot occurred. This overshoot is attributed to dynamic coupling and the ferry's inertia during rotation.

4.3 Docking

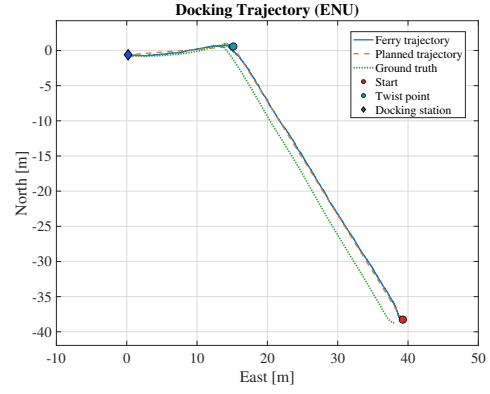
The docking scenarios described in Section 3.5 were performed and evaluated. The evaluation is mainly based on observations in the plots but also the pose error at the docking station. A successful docking pose is defined as a position error of less than 2.0 meters, and a heading error of less than 30 degrees. These relatively relaxed limits are motivated by the fact that in the real situation, Celia One is supposed to dock in-between two Y-booms.

4.3.1 Sequence 1: Scenario 1

The docking was performed without environmental disturbances shown in Figures 4.15 to 4.19.

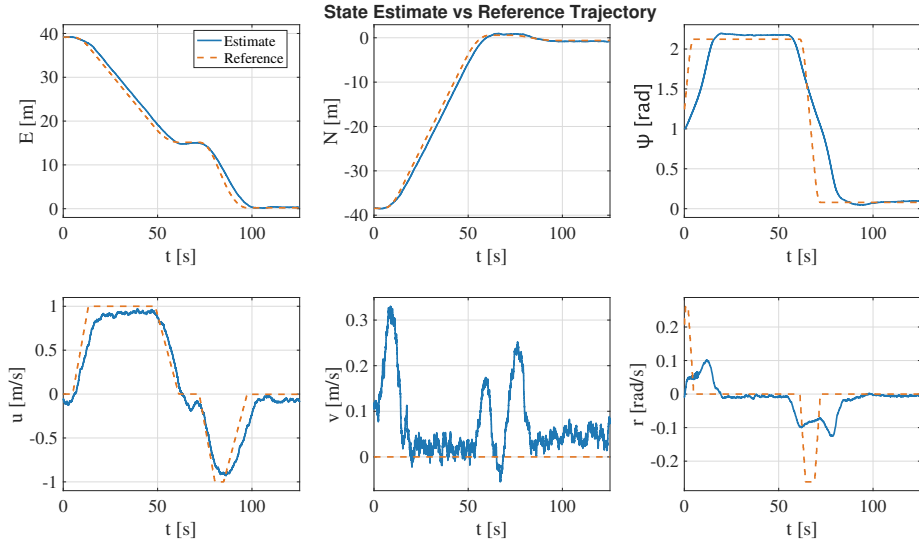


(a) MPC controller only.

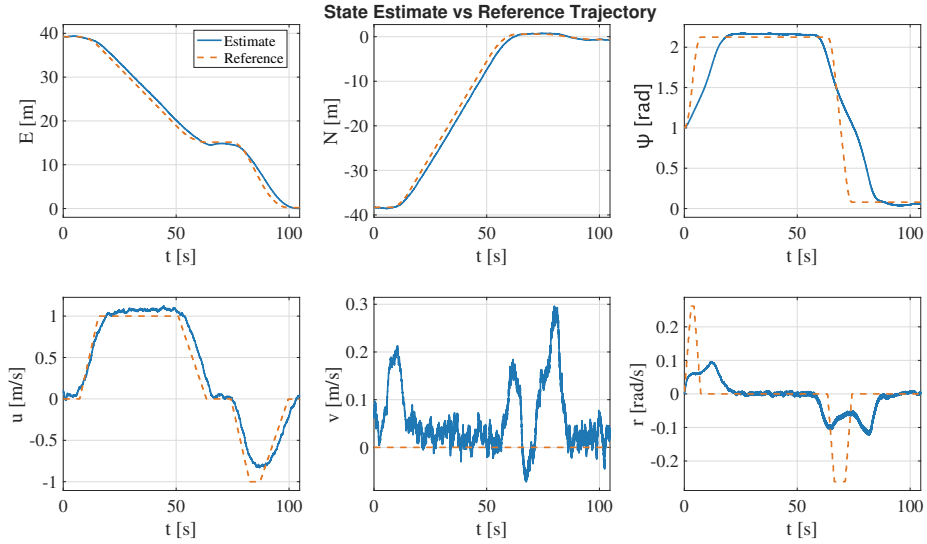


(b) MPC and LQR controller.

Figure 4.15: Bird's-eye view of docking Sequence 1: Scenario 1. The ferry trajectory (blue) is the estimated path traversed by the ferry. The planned trajectory, denoted by the dashed red line, is the Path-Planner reference. The ground truth is the true position according to the VRX environment. The twist point (green) is where the ferry rotates into position to reverse into the dock.

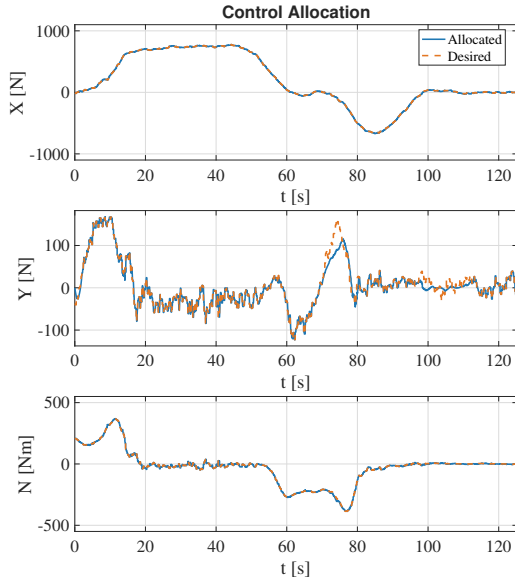


(a) MPC controller only.

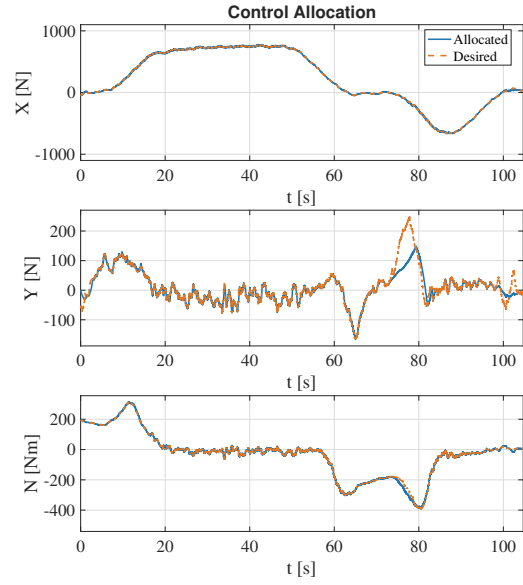


(b) MPC and LQR controller.

Figure 4.16: A comparison of the two different controllers tracking the reference during the docking sequence. The state estimate (blue) is the EKF filtered measurement. The red line is the Path-Planner reference. The horizontal axis denotes time.

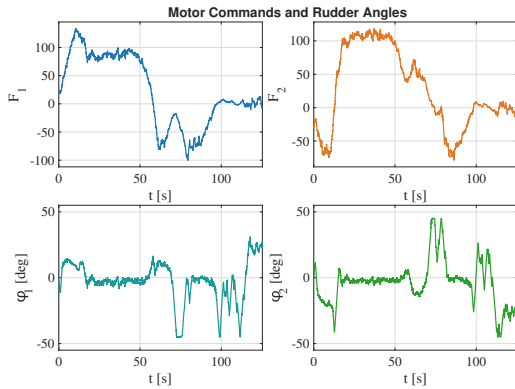


(a) MPC controller only.

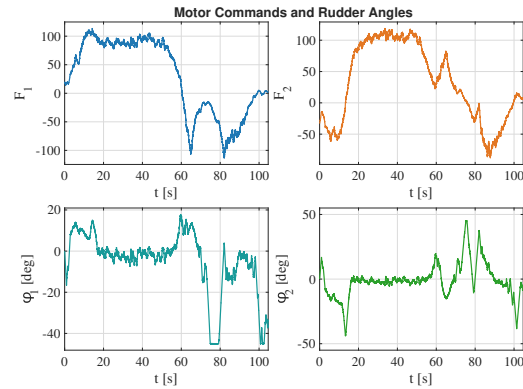


(b) MPC and LQR controller.

Figure 4.17: A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize. The blue lines denote τ_{cmd} and the red ones denote τ_{des} .



(a) MPC controller only.



(b) MPC and LQR controller.

Figure 4.18: Thrust and rudder angle commands based on the allocated forces and moments. The rudder angles and motor forces are computed directly by the Thrust-Allocator from Chapter 3.3.8. The thrust commands applied, F_i , are computed from the motor forces using Equation 3.30.

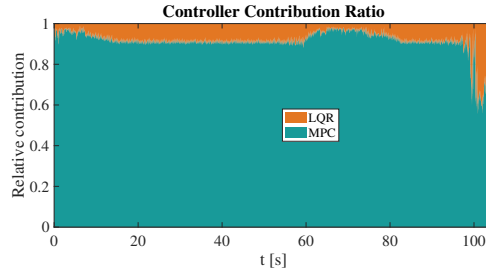


Figure 4.19: Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.

In Figure 4.16, the ψ and r plots illustrate the controllers ability to impose rotation. From the initial state of the trajectory, the ferry rotates to align with the green twist point shown in Figure 4.15. From the surge plot, the ferry performs the first rotation while moving. Once the green twist point is reached, the ferry performs a stationary rotation until the ferry heading locks with the docking station. The ferry then proceeds to reverse to its final docking position.

Figure 4.15 demonstrates that adding the LQR disturbance rejection controller does not visually improve the trajectory. This outcome is expected, as the scenario lacks wind and waves, providing ideal environmental conditions suited for the low-frequency MPC. Additionally, the first three plots in Figures 4.16a and 4.16b display the pose η_{pose} . Both controllers successfully dock the ferry at the desired (E,N) coordinates and heading angle.

The figures confirm accurate position tracking. The heading angle ψ tracks its reference but converges slowly. Figure 4.18a confirm that the commanded yaw moment N is not saturating, thus the bottleneck is not actuator authority. Instead, the issue lies within the cost function tuning. The weight on yaw-rate, r , in $\mathbf{Q}$ and $\mathbf{P}_f$ is low, so the optimizer has little incentive to command the aggressive torque needed to overcome the ferry's rotational inertia I_z and hydrodynamic damping N_r . The weight for ψ is large in $\mathbf{P}_f$. However, the 10-second lookahead (horizon $N_{horizon} = 50$ at 0.2-second steps) implies that the MPC can satisfy the terminal cost of ψ with modestly sustained r . Ultimately, this slow convergence and poor yaw-rate tracking reflect a deliberate design trade-off to prioritize position tracking. The primary objective is safe docking, for that it is more important to track position accurately than heading.

Regarding sway, the Path-Planner generates surge and heading references but sets the sway reference strictly to zero. Despite this, the controller utilizes nonzero sway when it minimizes the overall cost, as the cost function does not heavily penalize sway. This behavior represents an expected optimization outcome, not a tracking failure.

Figures 4.17a and 4.37b demonstrate that the requested forces and moments of the

controller remain feasible for the actuators. The error,

$$\boldsymbol{\tau}_{error} = \boldsymbol{\tau}_{des} - \boldsymbol{\tau}_{cmd} \approx \mathbf{0} \quad (4.2)$$

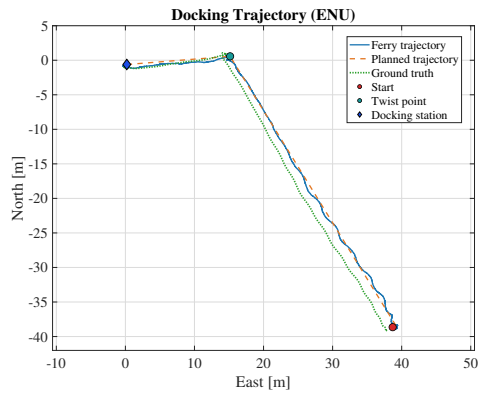
throughout most of the trajectory. This result is crucial because neither the MPC nor the LQR models account for thruster dynamics. The MPC restricts the output so that $\boldsymbol{\tau}_{mpc} \leq \boldsymbol{\tau}_{max}$ and uses the penalty matrix $\boldsymbol{S}$ to suppress large changes in the control signal. However, it is completely unaware of the rudder angular velocity limits. Consequently, the MPC might request an infeasible sway force Y , that the actuators don't have the physical capabilities to address. Generating sway force requires nonzero rudder angles, and the MPC might sometimes request sway force that exceeds the corresponding rudder velocity limit. Both figures highlight this limitation around $t \approx 75$, where $\boldsymbol{\tau}_{error} \neq \mathbf{0}$. If $\boldsymbol{\tau}_{error}$ remains large for extended periods, it suggests that the Thrust-Allocator consistently saturates. Consistent saturation would imply that the process model is suboptimal given the limitation of the Thrust-Allocator or that the desired trajectory is infeasible due to a large amount of disturbances. One solution would be to integrate the Thrust-Allocator directly into the MPC process model with its inherent constraints. This would result in a more consistently feasible MPC solution, as the MPC would optimize for a solution that accounts for the actuator limits. The current MPC accounts for maximum forces and moments, but does not know how fast the rudder angles can physically change. The downside of doing this is that it is not aligned with the controller's architectural philosophy of being modular, as discussed in Section 3.3.

Figures 4.18a and 4.38b illustrate the actual actuator commands. During the initial twist segment ($t \in [0, 20]$), the ferry does not rotate in place. Instead, it applies differential thrust ($F_1 > 0 > F_2$). Because $F_1 \neq -F_2$, the ferry translates and rotates simultaneously. The nonzero rudder angles further induce sway translation. This combination allows the ferry to track position (E, N) and heading concurrently, proving the Thrust-Allocator functions correctly.

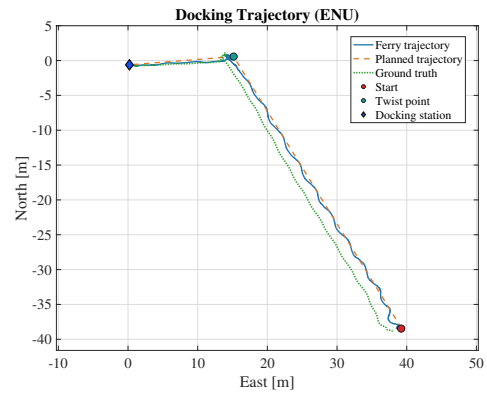
Figure 4.19 demonstrates the percentage of control effort provided by each component of the MPC-LQR controller. As intended, the MPC is the dominant controller. At $t \approx 98$ s the ferry is already stationary at the dock and has completed the trajectory. The LQR ratio increases during this point to maintain position. However, the reference during this time remains constant. The increased ratio of the LQR can be explained by the LQR over-reacting to noise in the measurements.

4.3.2 Sequence 1: Scenario 2

This scenario introduces wind and wave disturbances. The experiment sets the mean wind speed to 4 m/s with a variance of 1.5 m/s, and a wave amplitude of 0.5 meters. This is shown in Figures 4.20 to 4.24.

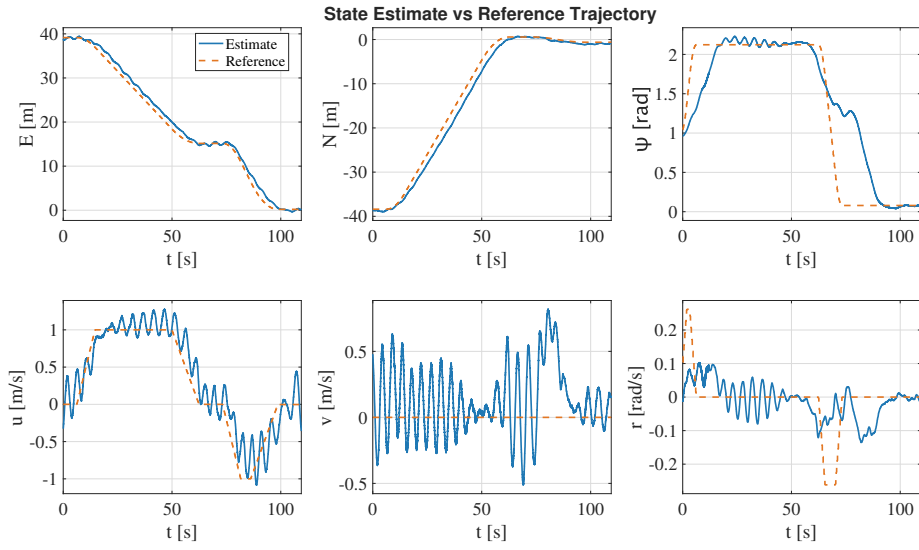


(a) MPC controller only.

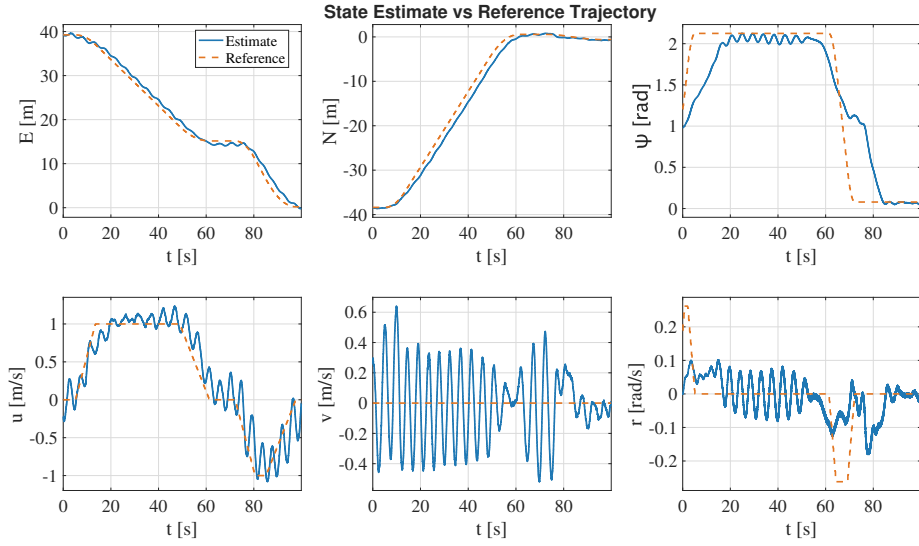


(b) MPC and LQR controller.

Figure 4.20: Bird's-eye view of docking (Sequence 1, Scenario 2).

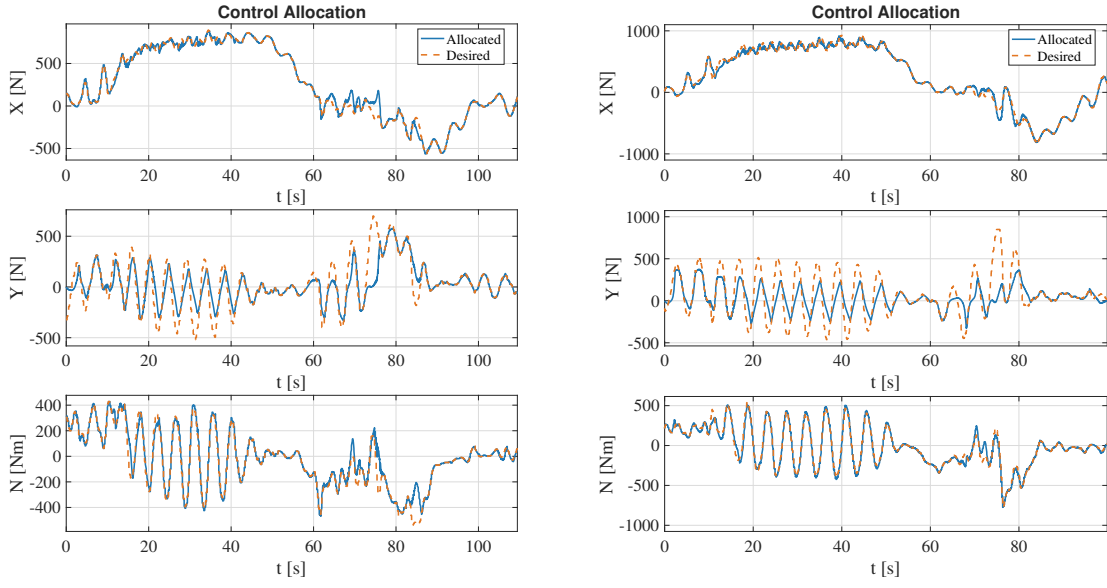


(a) MPC controller only.



(b) MPC and LQR controller.

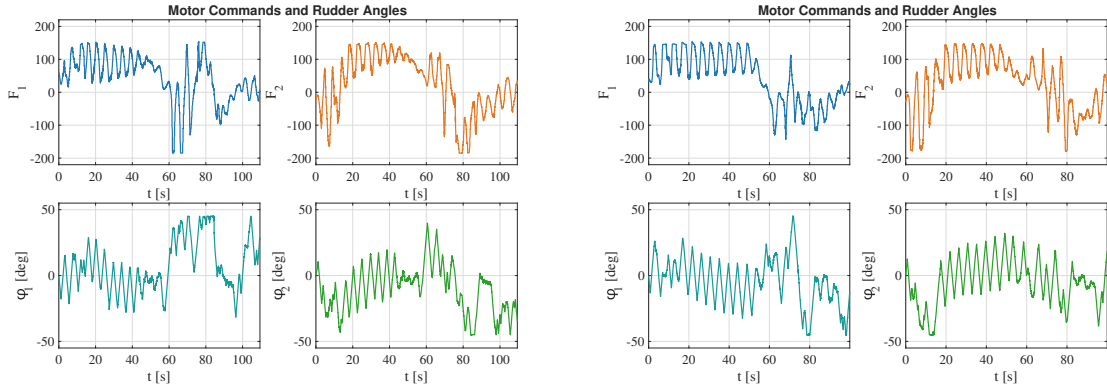
Figure 4.21: A comparison of the two different controllers tracking the reference during the docking sequence.



(a) MPC controller only.

(b) MPC and LQR controller.

Figure 4.22: A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.



(a) MPC controller only.

(b) MPC and LQR controller.

Figure 4.23: Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.

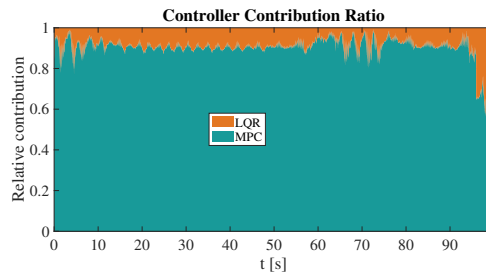


Figure 4.24: Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.

Figures 4.20a and 4.20b demonstrate how wind and wave disturbances affect the trajectory of the ferry. The disturbances are causing the ferry to slightly oscillate around its reference. However, the ferry successfully completes the docking sequence using both control structures.

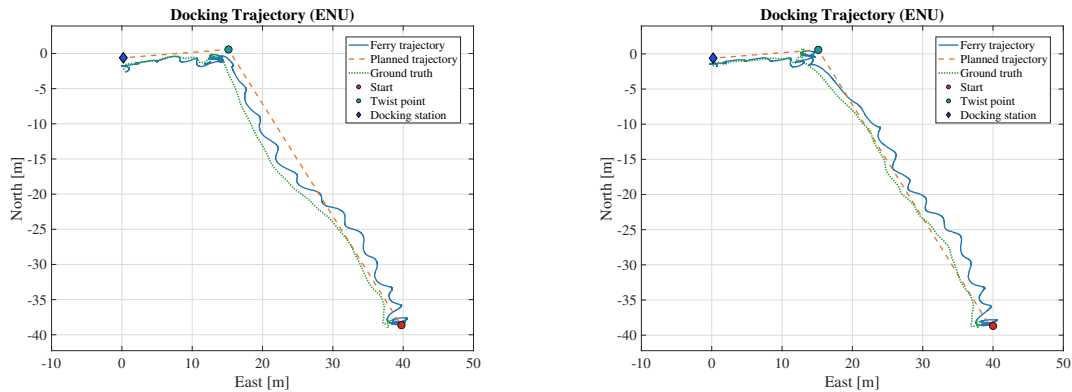
In Figures 4.21a and 4.21b, the surge and sway plots further emphasizes the oscillation caused by disturbances. The oscillation seems to be periodic, suggesting that this effect is mainly driven by the waves.

In Figures 4.22a and 4.22b it becomes evident that the controllers are trying to compensate for the oscillation caused by waves, but it is difficult for the Thrust-Allocator to provide the correct response in Y due to saturation. In Figures 4.23a and 4.23b, the motor thrust F_i and rudder angles are trying to compensate by oscillating back and forth.

Figure 4.24 illustrates a similar pattern to the disturbance free scenario displayed in 4.19. The reason for this is that while the LQR provides a larger magnitude of desired forces and moments, so does the MPC. That is, from Equation 3.61, both τ_{mpc} and τ_{lqr} increases in magnitude, but their proportions stay approximately the same.

4.3.3 Sequence 1: Scenario 3

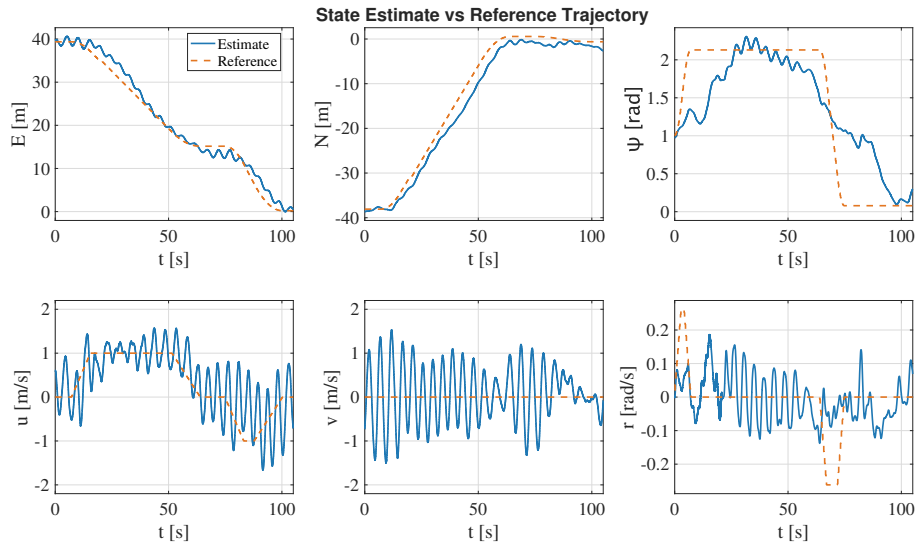
The docking-sequence is performed with a wind velocity mean of 7 meters per second with a variance of 5. The wave amplitude is set to 1.5 meters. The results are shown in Figures 4.25 to 4.29.



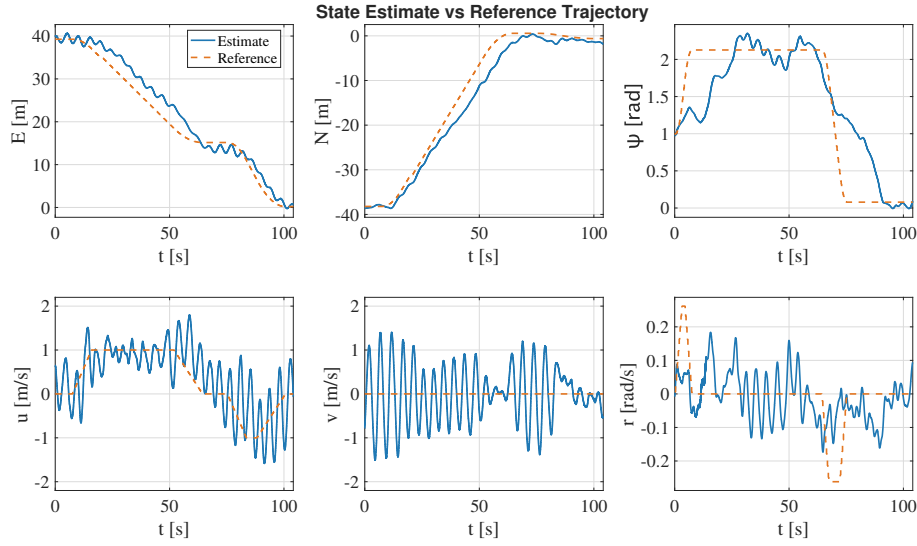
(a) MPC controller only.

(b) MPC and LQR controller.

Figure 4.25: Bird's-eye view of docking (Sequence 1, Scenario 3).

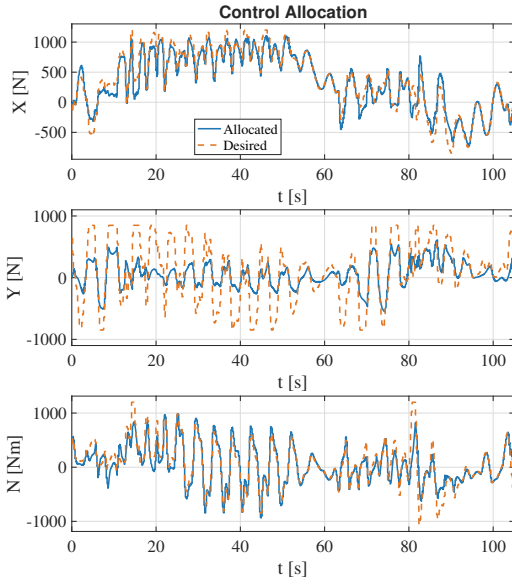


(a) MPC controller only.

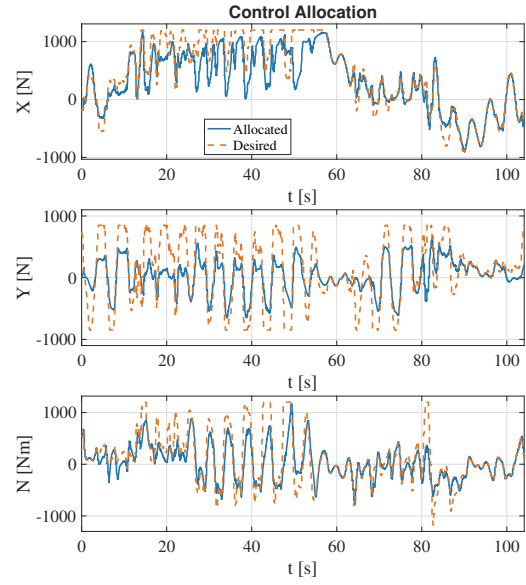


(b) MPC and LQR controller.

Figure 4.26: A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 1, Scenario 3).

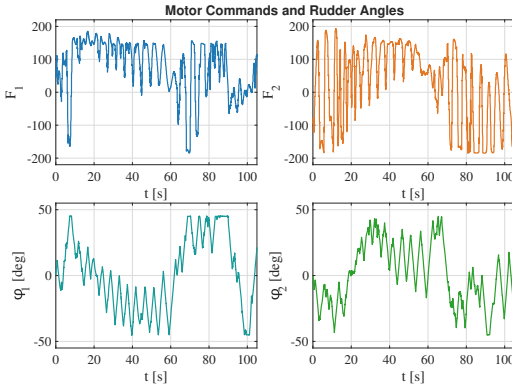


(a) MPC controller only.

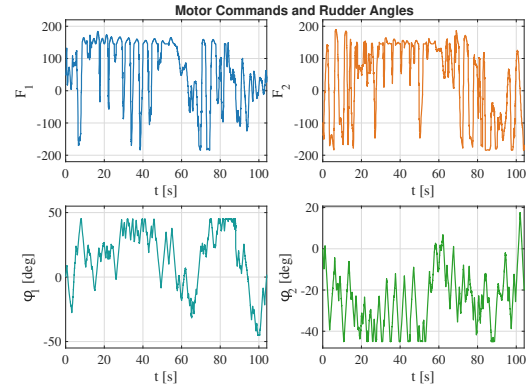


(b) MPC and LQR controller.

Figure 4.27: A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.



(a) MPC controller only.



(b) MPC and LQR controller.

Figure 4.28: Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.

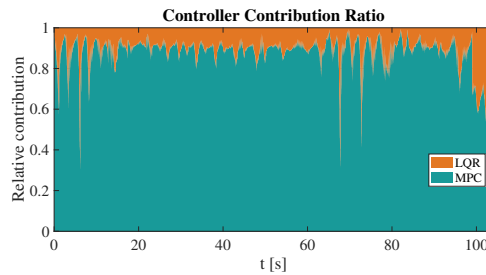


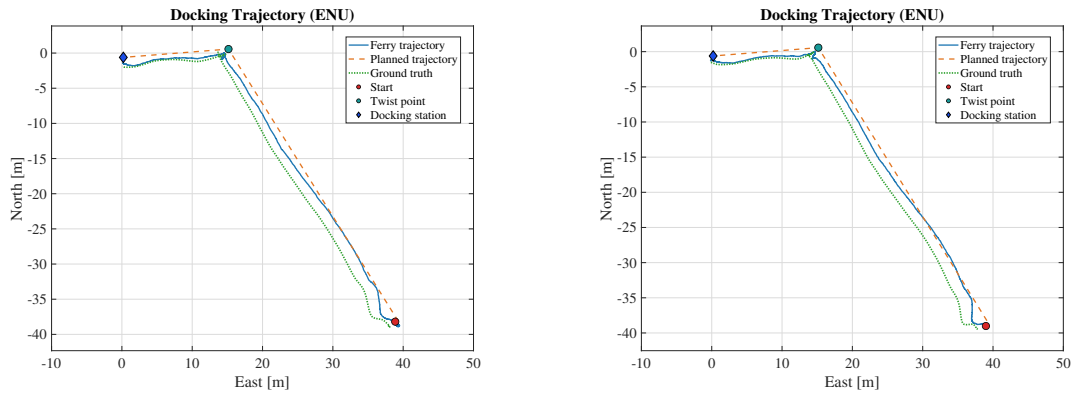
Figure 4.29: Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.

Figure 4.25 demonstrates that intense disturbances severely challenge the desired docking trajectory. Figure 4.27 indicate that the large oscillation caused by the waves are significantly saturating the actuators in both control-structures. The waves and wind cause positional drift, resulting in infeasible requests for sway-force Y . From Figure 4.28 the rudder angles are consistently oscillating, trying to provide the desired sway-force Y , but falters. From Figures 4.26a and 4.26b the ferry still manages the docking with a correct pose. So even though the controller is heavily saturated, the mission is a success for both controllers.

Figure 4.29 shows a similar oscillating behavior. When the waves cause translation motion of the ferry, the ferry also drifts from the MPC prediction. The LQR responds with increasing its control authority. This displays that the LQR is doing exactly what it is intended to do.

4.3.4 Sequence 1: Scenario 4

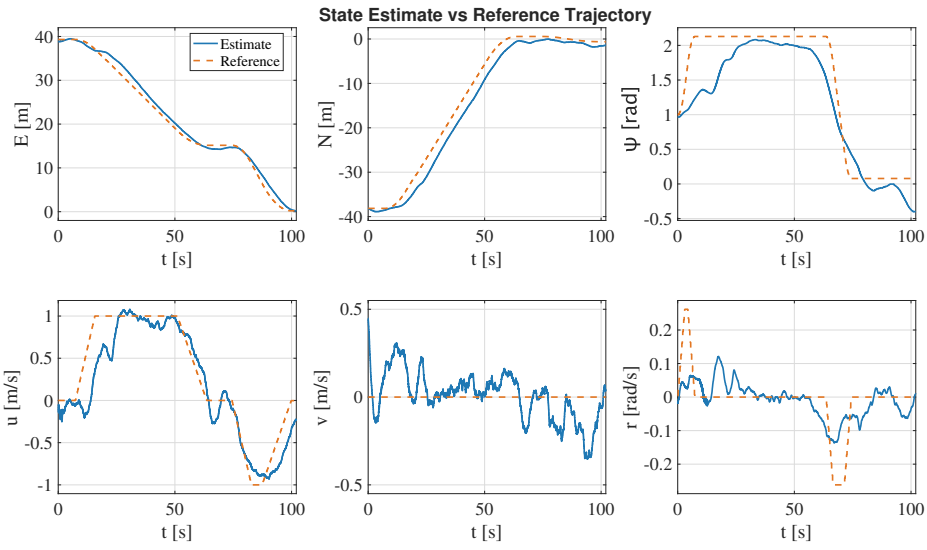
The docking-sequence is performed with a wind velocity mean of 11 meters per second with a variance of 4. The wave amplitude is set to 0 meters. The results are shown in Figures 4.30 to 4.34.



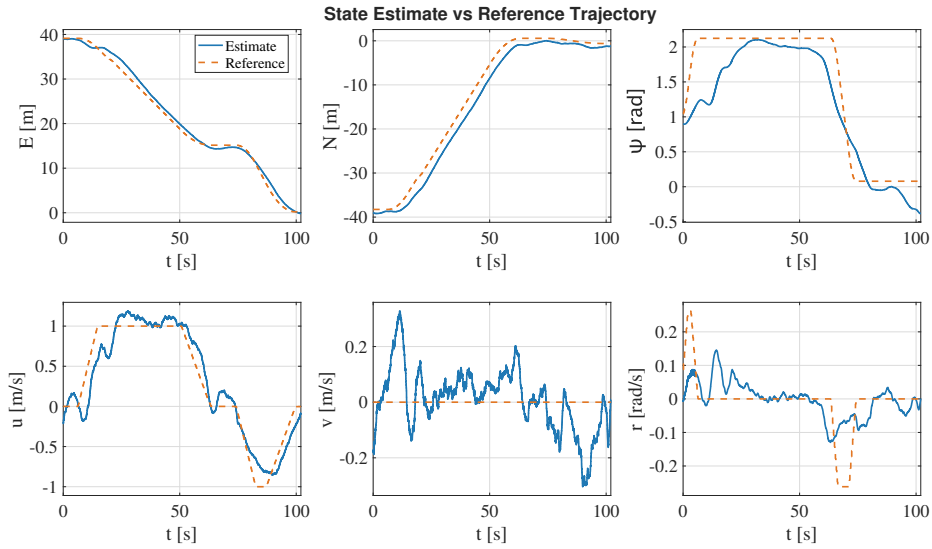
(a) MPC controller only.

(b) MPC and LQR controller.

Figure 4.30: Bird's-eye view of docking (Sequence 1, Scenario 4).

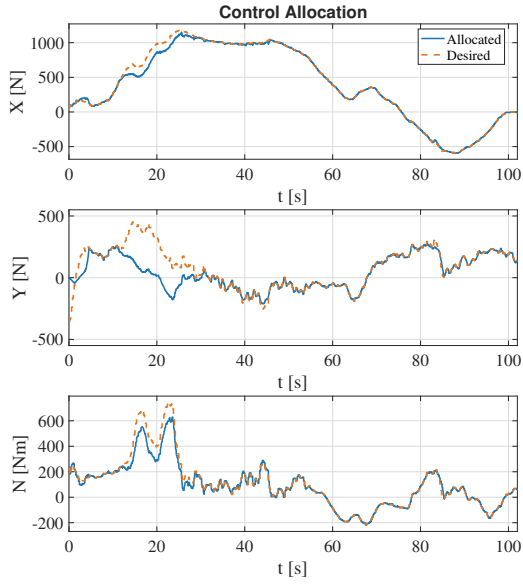


(a) MPC controller only.

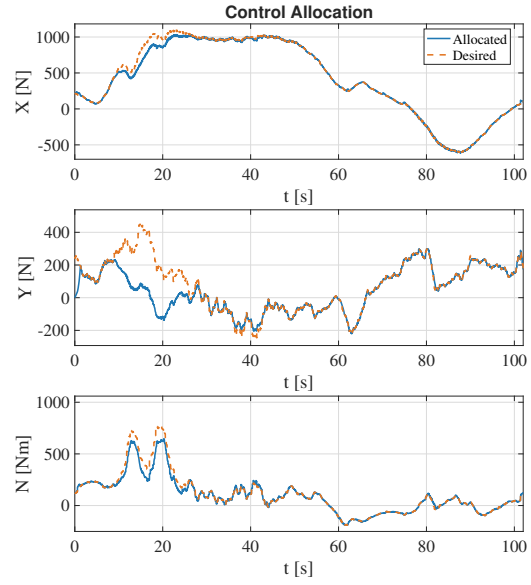


(b) MPC and LQR controller.

Figure 4.31: A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 1, Scenario 4).

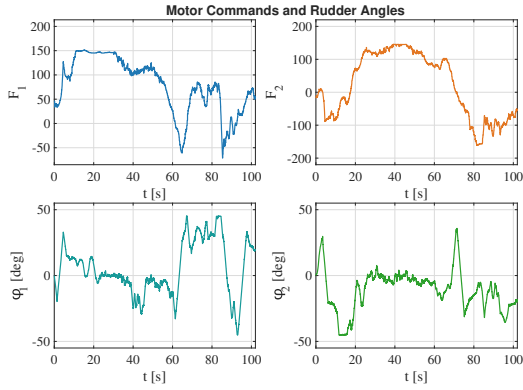


(a) MPC controller only.

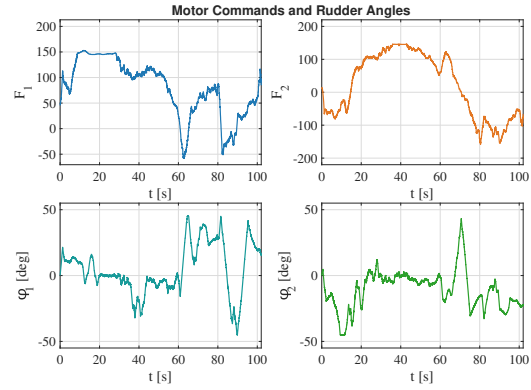


(b) MPC and LQR controller.

Figure 4.32: A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.



(a) MPC controller only.



(b) MPC and LQR controller.

Figure 4.33: Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.

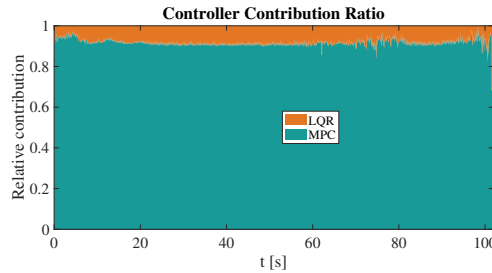


Figure 4.34: Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.

From Figures 4.30 and 4.31 both controllers perform a very similar trajectory. Both controllers manage to dock the ferry at the correct position with a heading error of less than 30 degrees.

From Figure 4.32a the desired forces and moments in Y are no longer oscillating as in the scenarios with waves. This is also visible in Figure 4.32b, where the motors and rudder angles no longer have to oscillate to compensate for the drift caused by waves. However, in the time interval $t \in [13, 27]$ the controller is saturating. In the same interval in the no-disturbance case (see Figure 4.17), the magnitude of τ_{des} is significantly smaller. This suggest that the wind-disturbance is significantly challenging the capabilities of the ferry.

4.3.5 Sequence 2: Scenario 1

The docking was sequence was carried out from the second starting point. The scenario is carried out without disturbances from wind or waves. The results are shown in Figures 4.35 to 4.39.

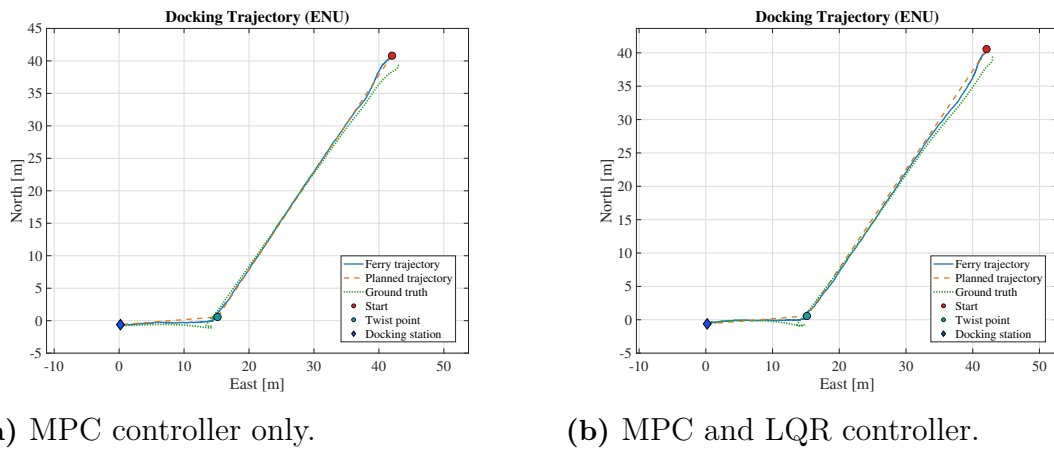
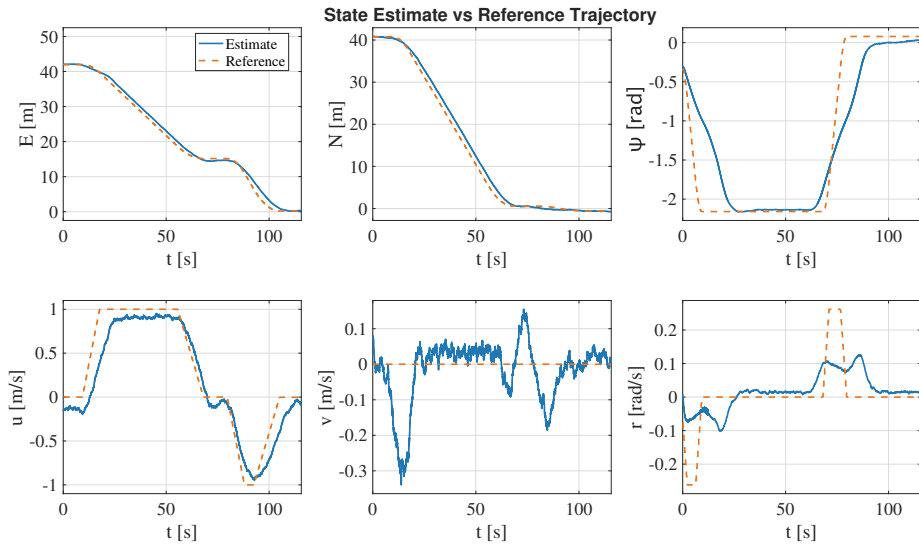
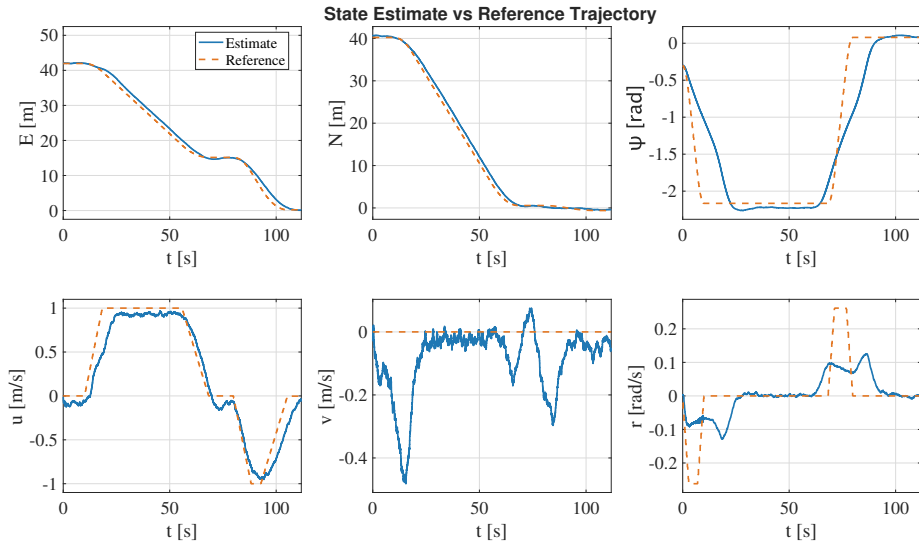


Figure 4.35: Bird's-eye view of docking from the second start position (Sequence 2: Scenario 1).

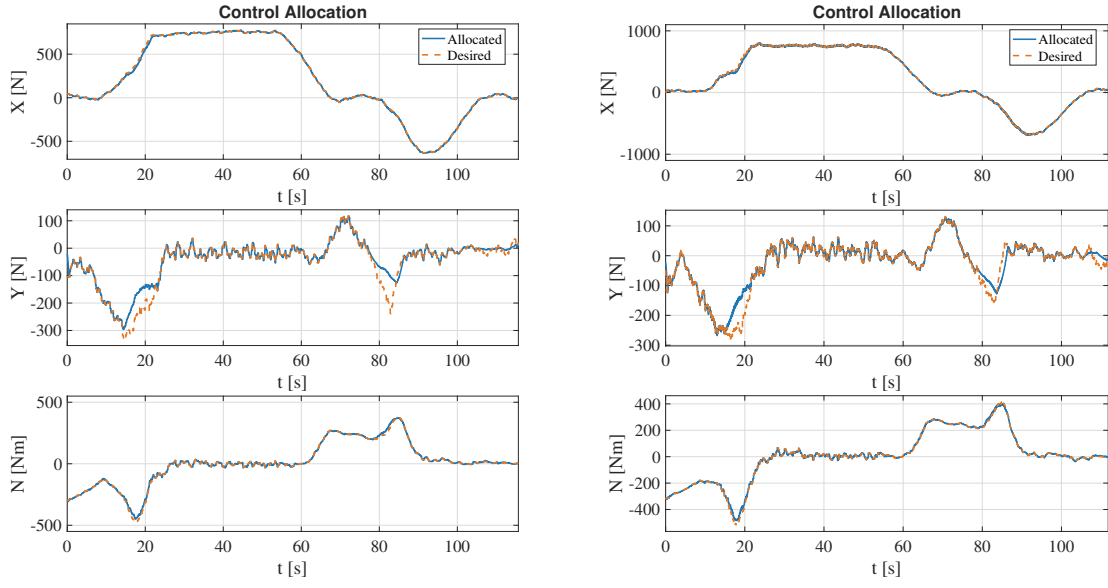


(a) MPC controller only.



(b) MPC and LQR controller.

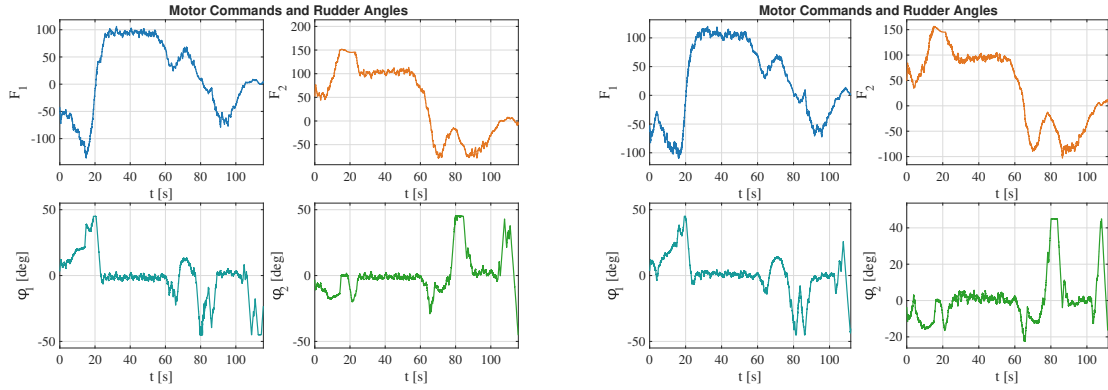
Figure 4.36: A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 2: Scenario 1).



(a) MPC controller only.

(b) MPC and LQR controller.

Figure 4.37: A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.



(a) MPC controller only.

(b) MPC and LQR controller.

Figure 4.38: Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.

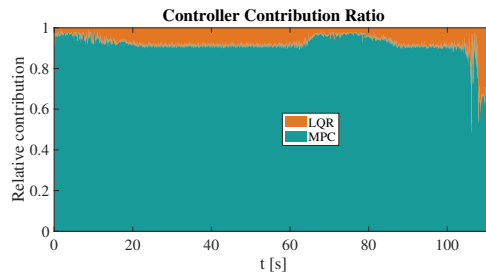
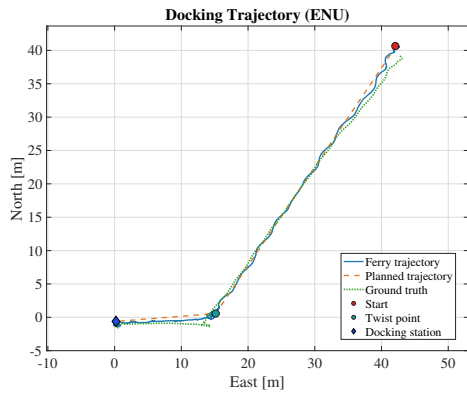


Figure 4.39: Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.

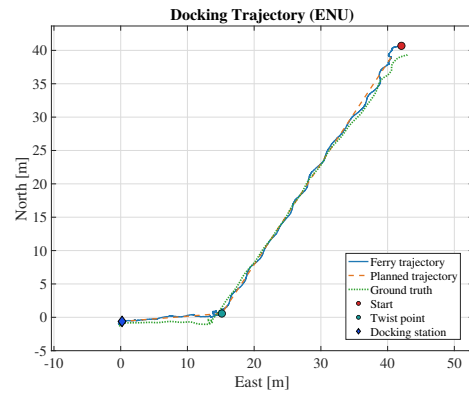
Figures 4.35, 4.36, 4.37, and 4.38 illustrate a successful docking sequence for both controllers. The final heading for the MPC-LQR controller is slightly more accurate, but the difference is only a degree or two. The controller ratio from Figure 4.39 is close to identical to that of Figure 4.19.

4.3.6 Sequence 2: Scenario 2

This scenario introduces wind and wave disturbances. The experiment sets the mean wind speed to 4 m/s with a variance of 1.5 m/s, and a wave amplitude of 0.5 meters. The results are shown in Figures 4.40 to 4.44.

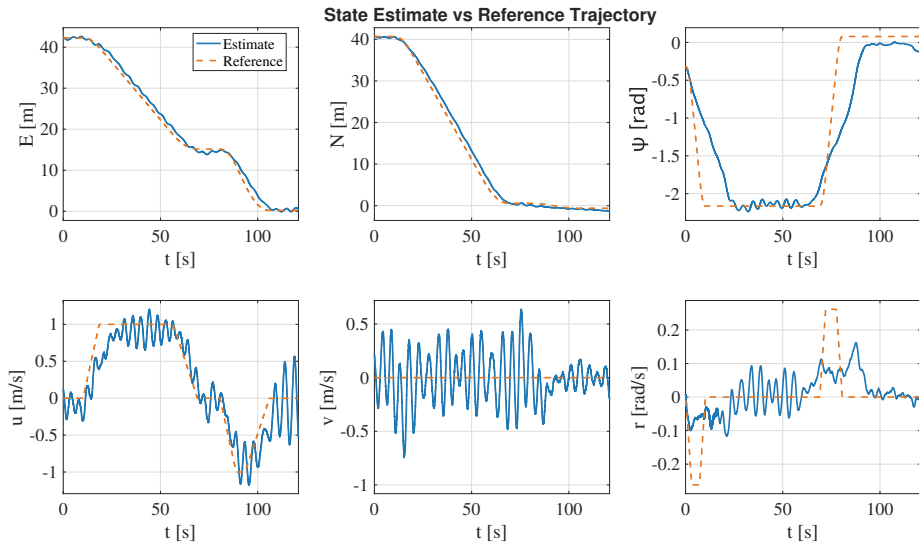


(a) MPC controller only.

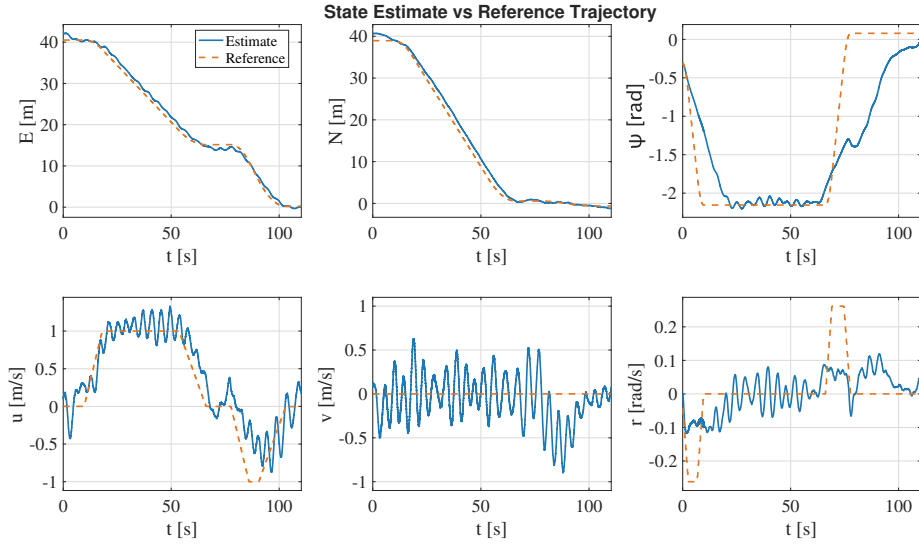


(b) MPC and LQR controller.

Figure 4.40: Bird's-eye view of docking (Sequence 2: Scenario 2).

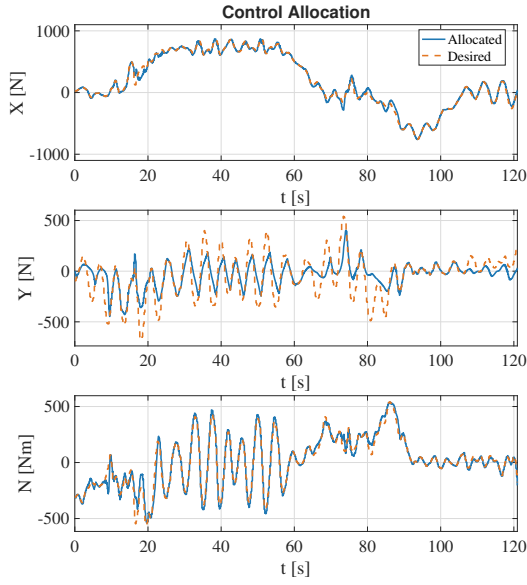


(a) MPC controller only.

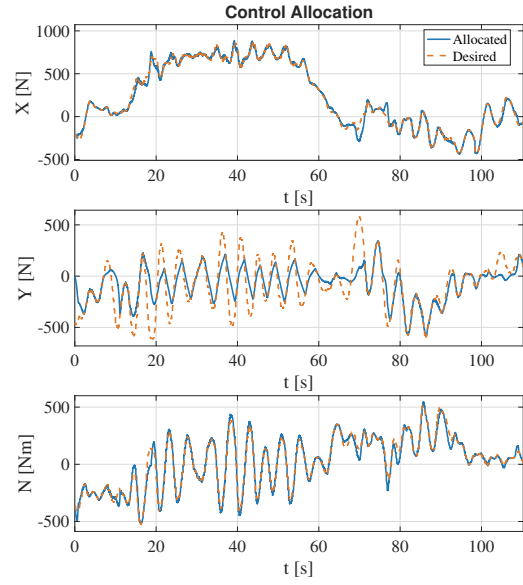


(b) MPC and LQR controller.

Figure 4.41: A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 2: Scenario 2).

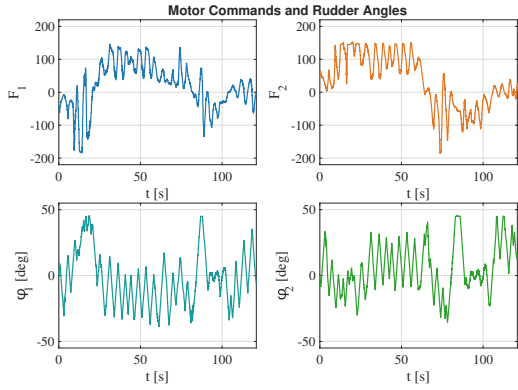


(a) MPC controller only.

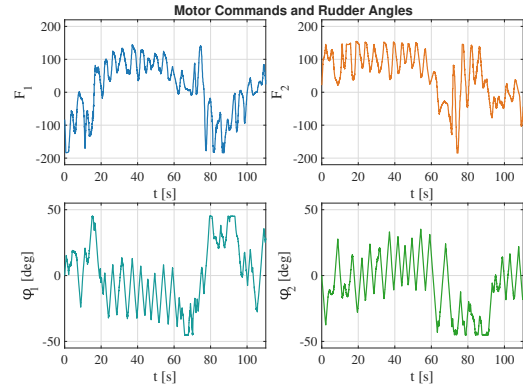


(b) MPC and LQR controller.

Figure 4.42: A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.



(a) MPC controller only.



(b) MPC and LQR controller.

Figure 4.43: Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.

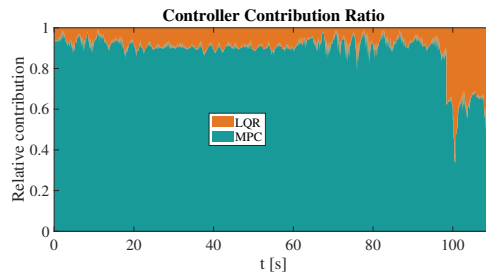
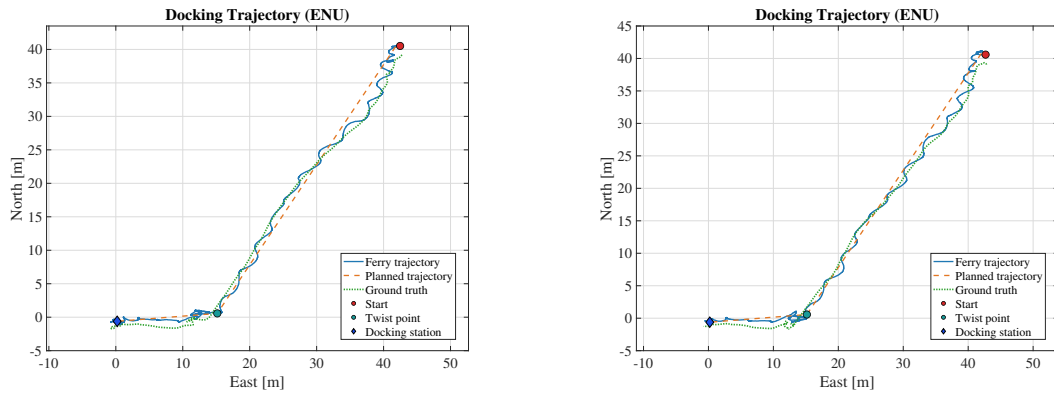


Figure 4.44: Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.

Figure 4.40 shows that both controllers manage to reach the correct docking position. From 4.41 both controllers reach the docking position at approximately $t = 100$. At this point the heading error is smaller for the pure MPC controller. In the interval, $t = [100, 110]$, the MPC-LQR controller makes small adjustments in the heading to compensate for the small initial misalignment. This correction is also seen in Figure 4.44, where the control-ratio of the LQR is increased in this interval. Thus, the MPC-LQR controller is compensating its heading angle at the dock position, while the MPC had a better heading at the time of arrival at the dock. However, heading error at arrival to the dock is less than 15 degrees for both controllers, meaning success in for both controllers. Figures 4.42 and 4.43 tell a similar story to that of their respective counterparts of Sequence 1: Scenario 2.

4.3.7 Sequence 2: Scenario 3

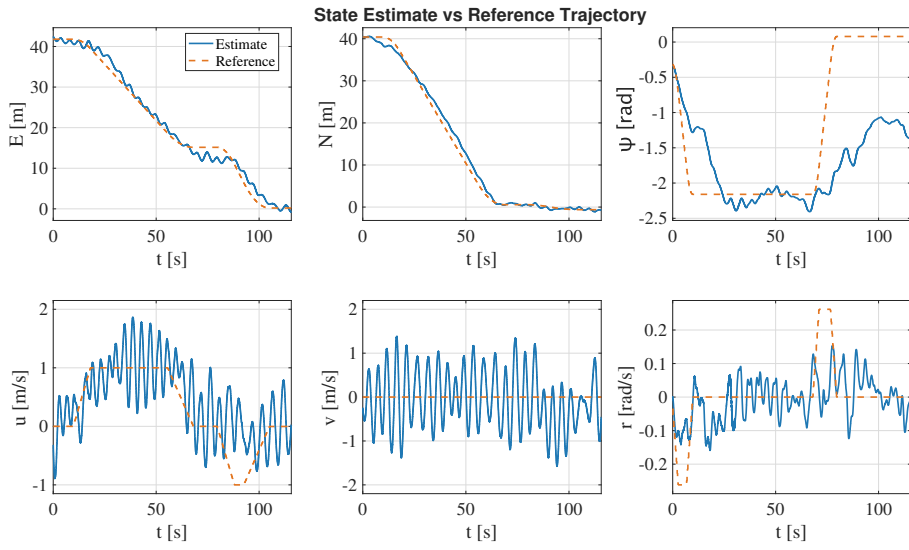
The docking-sequence is performed with a wind velocity mean of 7 meters per second with a variance of 5. The wave amplitude is set to 1.5 meters. The results are shown in Figures 4.45 to 4.49.



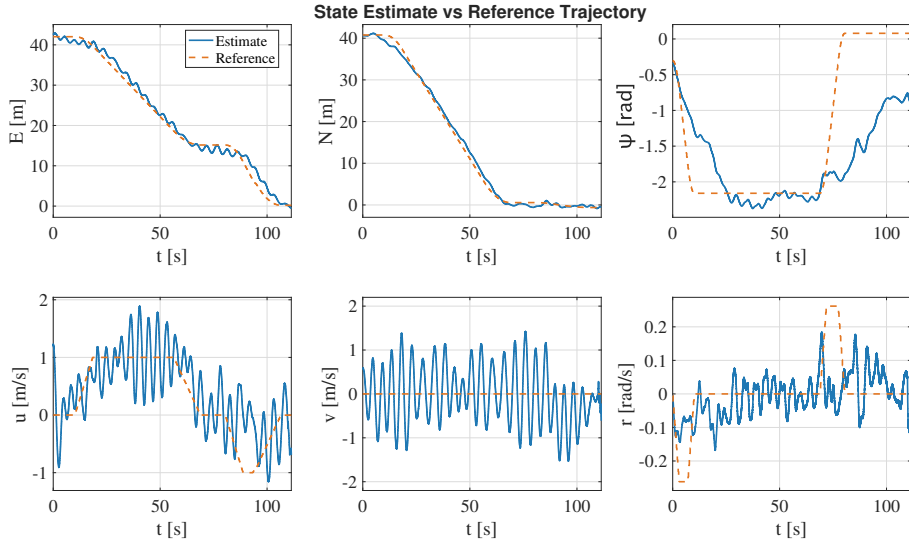
(a) MPC controller only.

(b) MPC and LQR controller.

Figure 4.45: Bird's-eye view of docking (Sequence 2: Scenario 3).

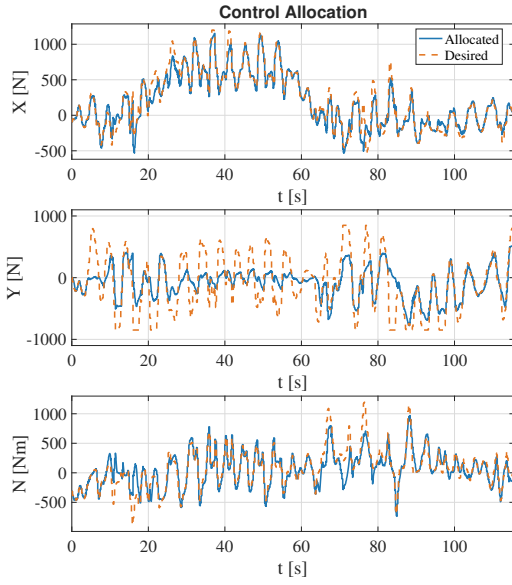


(a) MPC controller only.

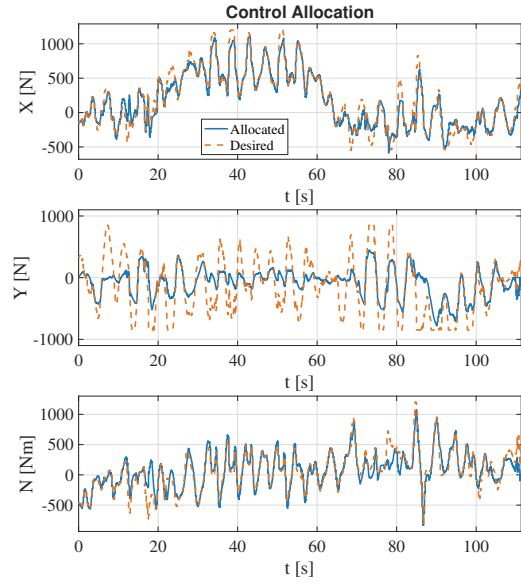


(b) MPC and LQR controller.

Figure 4.46: A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 2: Scenario 3).

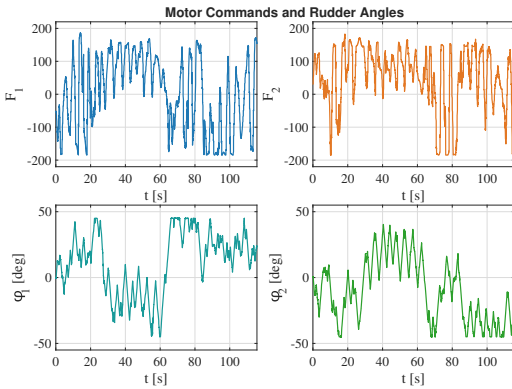


(a) MPC controller only.

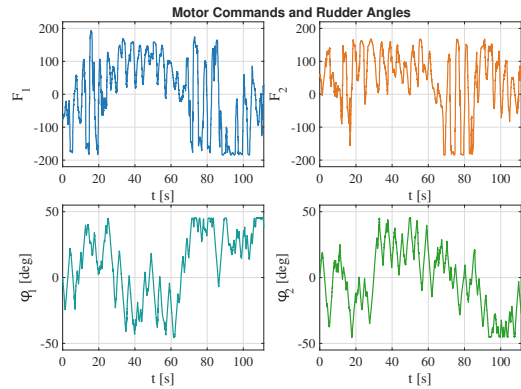


(b) MPC and LQR controller.

Figure 4.47: A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.



(a) MPC controller only.



(b) MPC and LQR controller.

Figure 4.48: Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.

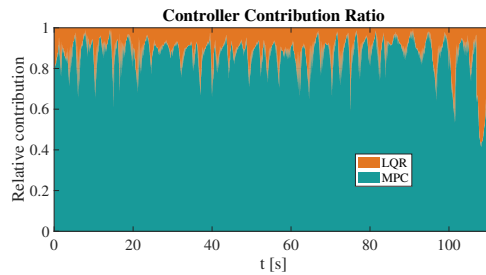


Figure 4.49: Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.

Figures 4.45 and 4.46 demonstrate that both controllers reach the correct docking position. For both control-structures, the ferry arrives at the dock at $t = 105$ s. At the time of arrival, the pure MPC has a heading error of approximately 70 degrees. For the MPC-LQR control structure, the heading error is approximately 45 degrees. These are rather large errors, thus both docking attempts of the ferry are deemed unsuccessful, but the MPC-LQR solution is performing better.

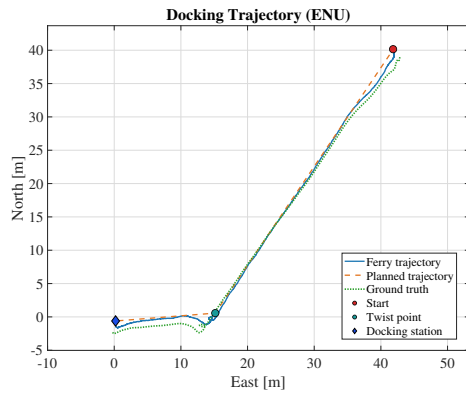
In this scenario, the wave amplitude is constant but the wind variance is large. At low velocities, hydrodynamic damping is minimal, so the ferry is more susceptible to wind-induced drift and rotation. From Figure 4.46, the ferry reaches the twist point at $t = 70$ s, and the yaw-rate plot shows the ferry attempting to rotate against the disturbances. The same disturbance profile is applied in Section 4.3.3, but with a different reference heading. Figure 4.26 demands a counter-clockwise rotation, while Figure 4.46 demands a clockwise rotation. Since the wind direction is fixed, the wind moment opposes the clockwise rotation, requiring larger compensating yaw moments from the controller. This is visible in Figure 4.47a and Figure 4.47b during the rotation interval $t \in [70, 80]$, the desired moment N shows larger peaks than in the counter-clockwise case (Figure 4.27a).

From Figure 4.47 it is evident that both controllers are saturating in the time interval $t \in [70, 80]$. Especially in the sway force Y . However, in contrast to the results in Figure 4.27, the Thrust-Allocator also saturates in yaw moment N during this interval. Neither controller can generate the commanded forces and moments, resulting in an inaccurate heading angle at the dock. Figure 4.48 also shows that the azimuth angle φ_1 saturates at its maximum of 45 degrees during this interval, further limiting the Thrust-Allocator's ability to direct force where it is needed.

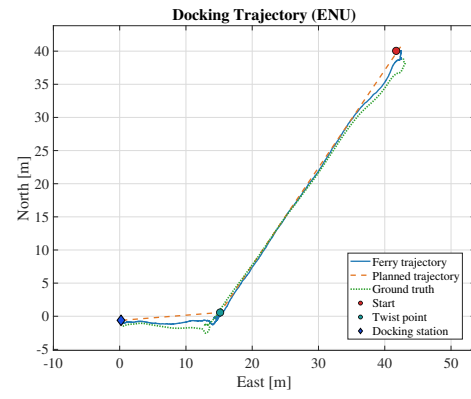
From Figure 4.49 shows how the LQR responds to waves. It looks very similar to Figure 4.29

4.3.8 Sequence 2: Scenario 4

The docking-sequence is performed with a wind velocity mean of 11 meters per second with a variance of 4. There are no waves. The results are shown in Figures 4.50a to 4.54.

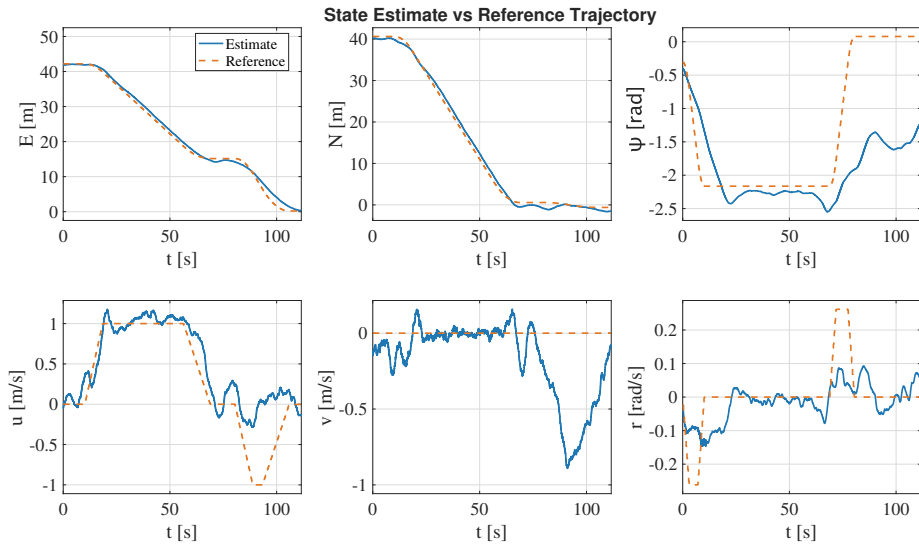


(a) MPC controller only.

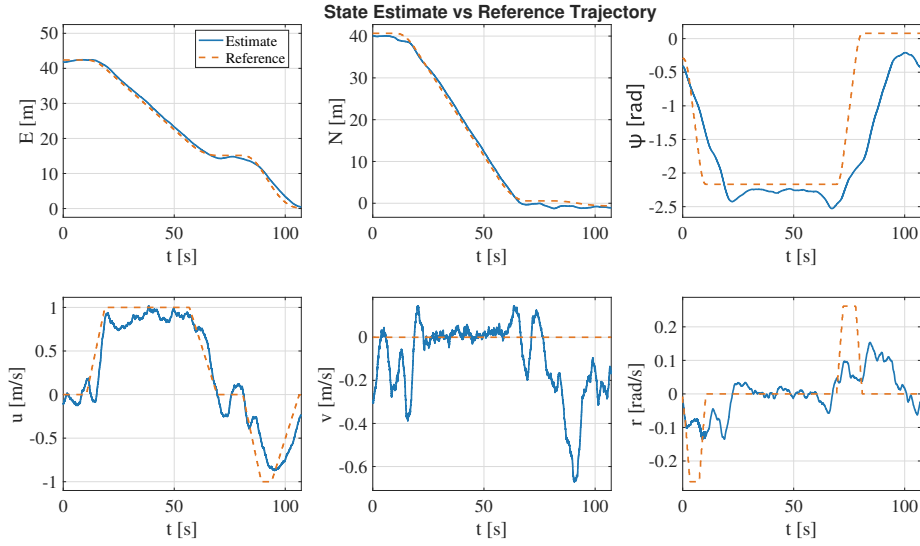


(b) MPC and LQR controller.

Figure 4.50: Bird's-eye view of docking (Sequence 2, Scenario 4).

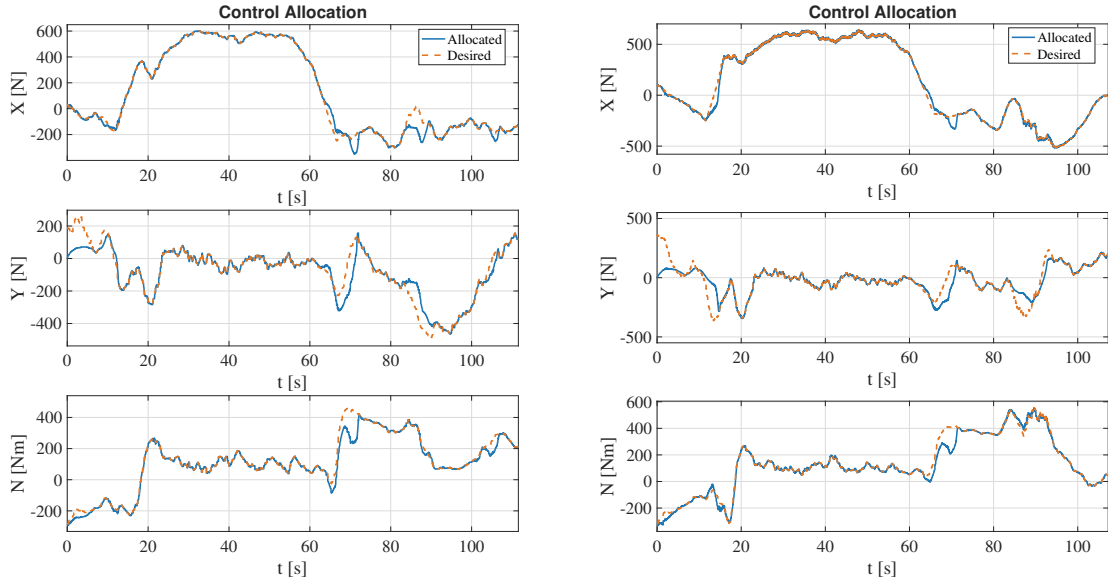


(a) MPC controller only.



(b) MPC and LQR controller.

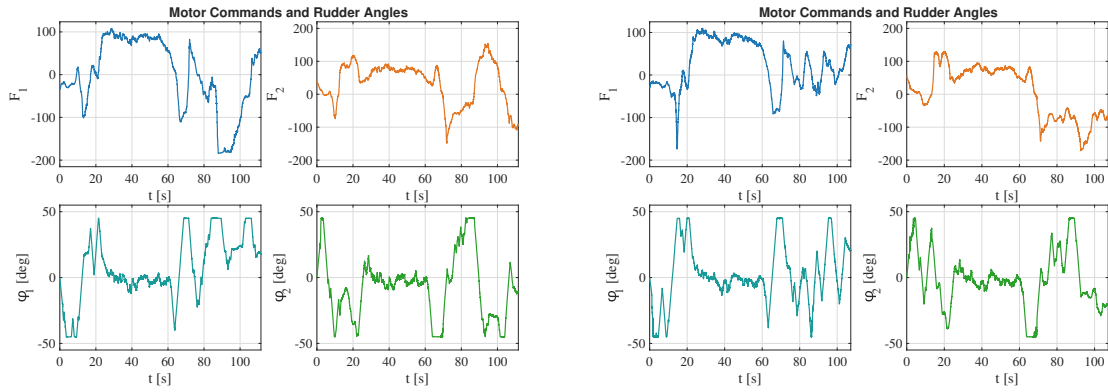
Figure 4.51: A comparison of the two different controllers tracking the reference during the docking sequence (Sequence 2, Scenario 4).



(a) MPC controller only.

(b) MPC and LQR controller.

Figure 4.52: A comparison of the forces and moments that the controller desires versus the forces and moments the Thrust-Allocator is able to realize.



(a) MPC controller only.

(b) MPC and LQR controller.

Figure 4.53: Thrust and rudder angle commands for the MPC and MPC-LQR controller respectively.

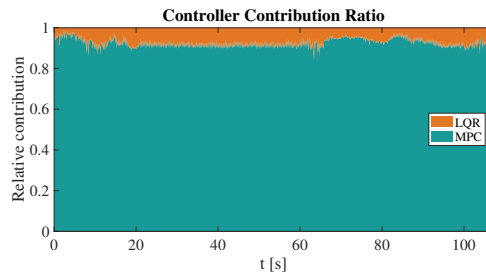


Figure 4.54: Illustration of the ratio of controller contribution in τ_{des} in the case where both controllers are used.

From Figures 4.50a and 4.50b both control structures reaches the docking position within a small error. The MPC-LQR controller is slightly more accurate in its final position. From Figure 4.51a the pure MPC reaches the dock with a error of approximately 1 meters in the north axis. The heading error at arrival for the pure MPC controller is approximately 80 degrees. The ferry should start reversing at $t = 80$ s. However, in the surge plot the speed briefly drops below 0, and then stays around zero during the segment. From the sway plot, it is instead observed that the ferry drifts sideways. This means that during the rotation segment, a wind gust hits the entire side-surface of the ferry, meaning that the wind-force is at its maximum. In turn this interrupts the rotation segments and just carries the ferry sideways. Furthermore, since the ferry is close to zero velocity at the twist point, there is almost no damping at the time when the wind hits. During the drift, the ferry is able to rotate slightly, but is unable to correct the large heading error before reaching the docking position.

From Figure 4.51b the MPC-LQR controller still manages to follow the surge reference with small errors meaning that the ferry successfully pulls of the intended reverse maneuver. The wind still hits at the same time instance and results in sway velocity, but the ferry still accurate tracks the desired position. In terms of final heading angle, the MPC-LQR structure arrives at the dock-station with a heading error of less than 30 degrees. Thus, the MPC-LQR controller successfully docks the ferry, while the pure MPC-controller falters.

The difference of how the controllers handle the reverse segment in the time interval, $t \in [80, 100]$, can also be seen in Figures 4.53a and 4.53b. The motor commands F_1 and F_2 completely saturates for the pure MPC controller, while only F_2 saturates for the MPC-LQR controller. The MPC has feedback, and solves for an optimal control sequence given its internal dynamics. But this control sequence is only optimal for its specific process model. The MPC is unaware of the large wind-disturbance and is linearized at its initial state. The control sequence is based on what the MPC thinks is optimal input at each horizon-step to reach the terminal state. But since the MPC does not account for the wind this control sequence is optimal given the linearized process model, but suboptimal given the true nonlinear plant. The result is bias in the applied control input. In the MPC-LQR controller, the LQR is handling this unmodeled disturbance with its higher frequency feedback. The LQR tracks the MPC prediction, thus it compensates for the model error of the MPC plant model. That is, the MPC generates good predictions given known dynamics, but it is unable to generate an optimal control sequence given the true plant of the system that would contain the force asserted by the wind. The LQR corrects for this at a high frequency, resulting in a more effective trajectory.

4.3.9 Summary of Docking Results

Table 4.2: Docking Performance of the MPC controller for Sequence 1.

Scenario	Pos Error	Heading Error	Pos OK	Heading OK	Docking OK
Scenario 1	0.16m	1.03°	Yes	Yes	Yes
Scenario 2	0.30m	0.68°	Yes	Yes	Yes
Scenario 3	1.98m	12.57°	Yes	Yes	Yes
Scenario 4	0.82m	27.28°	Yes	Yes	Yes

Table 4.3: Docking Performance of the MPC-LQR controller for Sequence 1.

Scenario	Pos Error	Heading Error	Pos OK	Heading OK	Docking OK
Scenario 1	0.19m	1.04°	Yes	Yes	Yes
Scenario 2	0.38m	2.19°	Yes	Yes	Yes
Scenario 3	1.30m	0.86°	Yes	Yes	Yes
Scenario 4	0.70m	26.44°	Yes	Yes	Yes

Table 4.4: Docking Performance of the MPC controller for Sequence 2.

Scenario	Pos Error	Heading Error	Pos OK	Heading OK	Docking OK
Scenario 1	0.13m	2.59°	Yes	Yes	Yes
Scenario 2	0.88m	12.16°	Yes	Yes	Yes
Scenario 3	0.90m	83.23°	Yes	No	No
Scenario 4	0.84m	73.54°	Yes	No	No

Table 4.5: Docking Performance of the MPC-LQR controller for Sequence 2.

Scenario	Pos Error	Heading Error	Pos OK	Heading OK	Docking OK
Scenario 1	0.22m	0.2°	Yes	Yes	Yes
Scenario 2	0.03m	10.26°	Yes	Yes	Yes
Scenario 3	0.84m	56.97°	Yes	No	No
Scenario 4	0.49m	29.13°	Yes	Yes	Yes

5

Conclusions and Future Work

From the docking results, both control structures successfully dock the ferry in low-disturbance scenarios. From Tables 4.3 - 4.5, it is evident that as disturbances levels increase, the MPC-LQR controller consistently outperforms its pure MPC counterpart. The additional feedback provided by the LQR thus successfully corrects the state deviations caused by unmodeled disturbances and linearization errors, resulting in improved docking performance.

In scenarios with both high-wind and wave disturbances, both controllers struggle due to the unmodeled disturbances and actuator saturation. In reality, this would not be a problem, as the operating waters of the Celia One will generally be calm. For Sequence 2: Scenario 4, where the disturbance is strictly wind, the unmodeled wind forces creates a large model mismatch between the linearized plant and the true nonlinear plant. Consequently, the MPC is optimizing for a sequence of control inputs that are suboptimal given the true plant. From Figure 4.52 the Thrust-Allocator has no issues with consistent saturation for neither controller. Thus, the problem is not actuator saturation, but rather that the MPC is requesting suboptimal forces and moments τ_{mpc} given the true dynamics. The fast feedback from the LQR partially solves this mismatch, and results in a successful docking sequence.

5.1 Future Work

This section consists of two parts. The first part describes how to refine and upgrade the aspects within the current project. The second part focuses on other factors beyond the scope of this project that are good or vital for CStrider to dock Celia One and future ferries reliably.

5.1.1 Within This Project

This section is divided into future work for the modeling of the ferry and for the controller.

5.1.1.1 Modeling

Due to time constraints, the hydrodynamical model was never fine-tuned. This should be done by running Experiment 6 and use this data in the procedure explained in Section 3.2.5. This also serves as model validation, that has only been done implicitly through full system docking tests in this project.

Because the controller was implemented in simulation, the angle and angular velocity constraints on Celia One's physical motors were approximated, but never accurately measured. This has to be done by running Experiment 5 before physical implementation.

VRX supports custom boat and hydrodynamical models. This makes it possible to import a 3D model of Celia One alongside parameters that match its hydrodynamical characteristics. Utilizing this feature will greatly simplify the process of tuning the controller for Celia One.

5.1.1.2 Controller

This controller is considered a proof of concept and is currently tuned for the WAMV instead of Celia One. Because of this, accurate and precise tuning was not prioritized. Though hard to quantify, fine tuning the controller should yield performance improvements.

The State-Estimator would improve if the hydrodynamical maneuvering model was implemented as process model of the EKF instead of the simplified kinematic model presented in Section 3.3.2. For Celia One this is not a problem since the installed INU provides an effective internal filter. But in the VRX simulation environment, an improved EKF filter that uses the nonlinear hydrodynamical model as plant would increase the accuracy of the predictions, improving the states estimates.

The control-algorithm does not currently account for any static or dynamic obstacles in the environment. In order to safely traverse urban water-ways and harbors, a map of static obstacles should exist, such that the planned-path never causes immediate danger to Celia One nor its passengers. This would mean that the Path-Planner would have to be reworked in order to account for this map of obstacles.

Dynamic obstacles pose serious challenges in urban water-ways. Examples of dynamic obstacles are moving objects in the water. Celia One has to traverse its path such that there is no collision risk. In order to implement, it is no longer sufficient to only rely on an IMU and a GNSS. The ferry's other sensors, such as its cameras and radars, would have to be used. Depending on the measurements, the planned path of the CStrider ferry would have to change dynamically. This can either be done by adding additional terms to the MPC cost-function that heavily penalizes positions in proximity of the dynamic obstacles or by adding hard constraints.

In order to be able to react to dynamic obstacles, the controller would need a larger headroom for path-planning. This would mean that the horizon length in time would

have to be larger. Long horizons allow the controller to carefully evaluate and plan the path-ahead, while remaining conservative in its control effort in order to avoid the obstacles. Since the uncondensed form MPC is substantially expensive for long horizons, the MPC should be redesigned to reduce complexity and thus allow for long horizons.

The actuator signals should be low-pass filtered before being sent to the motors in order to remove high frequent vibrations to reduce actuator degradation and stress. This would have to be designed such that the signal is smooth, but that the signal does not lag behind.

Currently, the controller is implemented in Python, and its compatibility with Celia One's main computer remains untested. If it proves incompatible, the controller must be rewritten in another language, or a secondary Python-capable computer must be installed and connected to the sensor and actuator network.

5.1.2 Not Within the Project

A significant challenge for Celia One and future CStrider ferries is the varying weight and position of passengers. For a ferry weighing approximately 3,500 kg, 12 passengers increase the total weight by nearly 30%. This weight shift changes the moment of inertia and the yaw rotation axis, especially when passengers stand and move aft during the docking procedure. While feedback mechanisms might compensate for these disturbances, calculating the exact moment of inertia for a real-life ferry is difficult. Therefore, developing a live parameter estimation system would be beneficial. Though, this should not be done before evaluating the effectiveness of feedback in compensating for this.

Another challenge involves sudden environmental disturbances. A ferry operating in a river or canal might dock behind a pier or breakwater that shelters it from wind and currents. However, once it leaves this cover, it encounters these forces simultaneously. To address this, the system can estimate and map disturbance forces at specific coordinates. This enables the ferry to preemptively compensate for disturbances based on its recent disturbance interactions in the same area.

Bibliography

- [1] Gustaver, M. (2020) A Chalmers University of Technology Master’s thesis template for L^AT_EX. Unpublished.
- [2] CStrider, “CStrider - Innovation on Water,” 2026. <https://www.cstrider.com/>, [Accessed: 2026-01-23].
- [3] A. Maternová, M. Materna, A. Dávid, A. Török, and L. Švábová, “Human Error Analysis and Fatality Prediction in Maritime Accidents,” *Journal of Marine Science and Engineering*, vol. 11, no. 12, p. 2287, 2023. doi: 10.3390/jmse11122287.
- [4] E. Eide, M. Breivik, E. Brekke, B.-O. H. Eriksen, E. Wilthil, Ø. Helgesen, E. Thyri, E. Veitch, O. A. Alsos, and T. A. Johansen, “The autonomous urban passenger ferry milliAmpere2: Design and testing,” *Journal of Offshore Mechanics and Arctic Engineering (JOMAE)*, vol. 147, article 031409, 2025. <https://doi.org/10.1115/1.4067370>
- [5] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Santa Barbara, CA: Nob Hill Publishing, 2017 (electronic, 5th printing, Sept. 2024). Available: <https://sites.engineering.ucsb.edu/~jbraw/mpc/MPC-book-2nd-edition-5th-printing.pdf>
- [6] R. Tedrake, *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation*, Course Notes for MIT 6.832, 2023. [Online]. Available: <https://underactuated.csail.mit.edu/lqr.html>
- [7] B. Bingham, C. Agüero, M. McCarrin, J. Klamo, J. Malia, K. Allen, T. Lum, M. Rawson, and R. Waqar, “Toward Maritime Robotic Simulation in Gazebo,” in *Proceedings of MTS/IEEE OCEANS Conference*, Seattle, WA, October 2019.
- [8] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, “Robot Operating System 2: Design, architecture, and uses in the wild,” *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022. doi: 10.1126/scirobotics.abm6074.
- [9] A. Bergholtz and C. Hernvall, “Waypoint-based path following system for a jet ski: Development, simulation and implementation of a path following control

- system for an autonomous driving jet ski that follows a lead boat,” Master’s thesis, Dept. Signals & Systems, Chalmers Univ. Technol., Gothenburg, Sweden, 2015.
- [10] E. Ingemarsson, F. Kerstis, P. Lindesson, M. Löfgren, H. Male, and F. Milqvist, “Autonomous WaveRunner - Follow the Leader: Development of an autonomous path following system for a personal watercraft,” Bachelor’s thesis, Department of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden, 2019.
 - [11] N. Danielsson and V. Lindström, "Development and Verification of an Autonomous Personal Watercraft Testbed," Master’s thesis, Department of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden, 2021.
 - [12] E. Laitala and E. Strömdahl, “Autonomous Docking: Trajectory planning and dynamic route adaptation for autonomously docking a marine vessel using Model Predictive Control,” Master’s thesis, Department of Mechanics and Maritime Sciences, Chalmers University of Technology, Gothenburg, Sweden, 2024.
 - [13] S. Balasubramanian, G. Licht, A. Richter, M. Büker, and A. Trächtler, “Comparison of Dynamic and Kinematic Model Driven Extended Kalman Filters (EKF) for the Localization of Autonomous Underwater Vehicles,” *arXiv preprint arXiv:2105.12309*, 2021.
 - [14] Chalmers University of Technology, “Regulations for the use of AI tools” 2023. [Online]. Available: <https://www.chalmers.se/en/education/your-studies/masters-and-bachelors-thesis/regulations-for-the-use-of-ai-tools/>. [Accessed: 23-Jan-2025].
 - [15] M. Hallman, “Interview regarding Maritime Automation and Urban Ferry Operations,” Personal interview conducted by M. Odenmarck, January 2026.
 - [16] T. I. Fossen, *Guidance and Control of Ocean Vehicles*. Wiley, 1994. ISBN: 0471941131.
 - [17] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, Ltd, 1st edition, 2011. ISBN: 978-1-119-99149-6.
 - [18] Simo Särkkä and Lennart Svensson (2023). Bayesian Filtering and Smoothing. Second Edition. Cambridge University Press.
 - [19] R.A. Adams and C. Essex, *Calculus: a Complete Course*, Pearson Education Australia, 2017.
 - [20] J. L. Jerez, E. C. Kerrigan, and G. A. Constantinides, “A sparse and condensed QP formulation for predictive control of LTI systems,” *Automatica*, vol. 48, pp. 999–1002, 2012.

- [21] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, “OSQP: an operator splitting solver for quadratic programs,” *Mathematical Programming Computation*, vol. 12, pp. 637–672, 2020.
- [22] A. Zhakatayev, B. Rakhim, O. Adiyatov, A. Baimyshev, and H. A. Varol, “Successive linearization based model predictive control of variable stiffness actuated robots,” in *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 2017, pp. 1774–1779.
- [23] D. Papadimitriou, U. Rosolia, and F. Borrelli, “Control of Unknown Nonlinear Systems with Linear Time-Varying MPC,” in *Proc. 59th IEEE Conf. Decision Control (CDC)*, Jeju Island, South Korea, Dec. 2020, pp. 2258–2263, doi: 10.1109/CDC42340.2020.9304441.
- [24] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed., 5th printing. Madison, WI: Nob Hill Publishing, 2017, pp. 223–228.
[Online]. Available: <https://sites.engineering.ucsb.edu/~jbraw/mpc/MPC-book-2nd-edition-5th-printing.pdf>

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY

Gothenburg, Sweden

www.chalmers.se



CHALMERS
UNIVERSITY OF TECHNOLOGY