OUKA
The University of Osaka
Institutional Knowledge Archive

Title	On the Efficient Trajectory Generation and the Online Control to Realize Automatic Docking: Simulation and Experiment
Author(s)	Rachman, Dimas Maulana
Citation	大阪大学, 2023, 博士論文
Version Type	VoR
URL	https://doi.org/10.18910/92971
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

Doctoral Thesis

On the Efficient Trajectory Generation and the Online Control to Realize Automatic Docking: Simulation and Experiment

(自動着桟制御を実現する自動経路生成とオンライン制御に関する研究)

Semionline Trajectory Planner, Trajectory Tracking Controller with Evolution Strategy, and Multipurpose Low-speed Positioning System with VecTwin Rudder

Dimas Maulana Rachman

June, 2023

Department of Naval Architecture and Ocean Engineering
Division of Global Architecture
Graduate School of Engineering
Osaka University

On the Efficient Trajectory Generation and the Online Control to Realize Automatic Docking: Simulation and Experiment
Discussion and demonstration on (1) an OCP-based (optimal control problem) semionline (semi-real time) trajectory planner for automatic docking that takes into account the wind dynamics and geometry of a harbor, (2) a controller for trajectory tracking operation and path following operation that implements an evolution algorithm, and (3) a multipurpose low-speed positioning system with a high-performance VecTwin rudder.

Dimas M. Rachman, 2023.
Supervisor: Professor Atsuo Maki, Ship Intelligentization Subarea, Osaka University

Review Committee:
Professor Atsuo Maki, Ship Intelligentization Subarea, Osaka University (*chair*)
Professor Hiroyoshi Suzuki, Marine Hydro-Science and Engineering Subarea, Osaka University
Professor Munehiko Minoura, Ocean Space Development Subarea, Osaka University

Doctoral Thesis
Department of Naval Architecture and Ocean Engineering
Division of Global Architecture
Graduate School of Engineering
Osaka University
2-1 Yamadaoka, Suita, Osaka 565-0871



Summary

This thesis addresses—from the practical point of view—the two most common problems that arise when discussing the automatic docking of a ship. The first problem is how to efficiently plan a safe docking trajectory (or path) when the instantaneous environment dynamics is considered and the ship's maneuvering dynamics is highly nonlinear. The second problem is how to control the ship to track/follow the given trajectory/path until safely stops at the final docking point.

The first problem arises when there is a need to replan a new trajectory for docking due to some changes in the environmental dynamics, e.g., wind force. The common method to plan a trajectory is by solving a trajectory optimization problem. It is a case of continuous nonlinear optimal control problem (OCP) that is solved directly by transcribing it into a finite-dimensional nonlinear optimization problem (nonlinear programming/NLP) via Hermite-Simpson collocation method. Spatial constraints related to the geometry of the harbor (convex/nonconvex) are included in the NLP such that a safe (allision-free) trajectory can be generated. These allision constraints are generated via point-in-polygon method where the ship is represented with multiple points instead of just one point.

The constructed NLP is then solved by the proven gradient-based solver known as sequential quadratic programming (SQP). However, the wind and the highly nonlinear dynamics of the ship increase the difficulties of the NLP, consequently leading to a longer computational time. These issues can be attenuated to some extent by warm-starting the NLP, i.e., using a near-globally optimal solution as the initial guess (the location where the gradient descent starts) for the solver. This near-globally optimal solution is generated by solving an identical NLP but with a better global solver, e.g., evolution strategy with covariance matrix adaptation (CMA-ES). It was demonstrated in numerical simulations that the balance between computational efficiency and feasibility of the solution can be achieved by warm-starting the NLP; quantified by significant computational time speedups. This fast trajectory planner can be integrated into an automatic docking support system from which a decision on whether a docking operation can be executed safely can be concluded.

There are two topics in the second problem: trajectory tracking and path following, where the former is generally more difficult than the latter. There are many methods that can be implemented when designing a controller for each of the topics. The most common one is via nonlinear model predictive control (NMPC) that implements a gradient-based solver. This solver may fail to converge to the solution when the search space is nonsmooth with discontinuities due to the nonlinearities. To overcome this, despite being a computationally expensive solver, it is demonstrated that CMA-ES can also be used as an algorithm to solve the NLP in the controller, both for trajectory tracking and path following of an underactuated ship. This was tested in simulations where constant wind disturbance exists.

Moreover, when addressing the second problem, one of the most important matters to consider is the redundancy and the ability of the actuators of the ship. It is arguably always true that an underactuated ship will always have difficulties executing an automatic docking operation without the help of external devices. A fully actuated ship, e.g., a DP ship (dynamic positioning) usually has multiple thrusters that can accelerate the ship

in the surge and the sway directions. However, this kind of arrangement is expensive to install and maintain. This thesis introduces an arrangement featuring a fixed-pitch propeller (as the main thruster), a bow thruster, and a high-performance VecTwin rudder system.

A VecTwin rudder system comprises two fishtail rudders with reaction fins attached to elevate their performance. Together with at least one bow thruster, it enables the ship to execute low-speed maneuvers—also called dynamic positioning—like hover, crabbing/sway, rotation, and reversal; all at a constant propeller revolution number in forward mode. With this arrangement, a novel low-speed positioning system for a calm harbor area is introduced. The system is developed on the assumption that the forces due to the interaction between the propeller, the VecTwin rudders, and the hull, are linear/proportional with the rudder angles within a certain range around the hover rudder angle. When the resources and time to conduct bollard-pull experiments are limited, such a linear relationship can be obtained via multiple linear regression of the results from several computational fluid dynamics (CFD) simulations. This linear relationship allows a simple and robust PID controller to be designed. The PID controller regulates the required forces to achieve the control objective. These forces are then mapped to the appropriate control commands.

The novel low-speed positioning system was tested in several scale model experiments where environmental disturbances (e.g., wind and gusts) exist. The performance of the system shows a promising practical implementation. Moreover, the overall approach when designing the positioning system and the system itself can be an addition to the existing design of a dynamic positioning system.

Acknowledgements

All praises to the almighty God, Allah *subhanahu wa ta'ala*, for His uncountable blessings and mercy that the author thrived and held strong during his study despite the many ups and downs. The author will never be able to complete this thesis without endless prayers, support, and wisdom from his parents, especially the author's mother, ibu Hj. Nurmiati Syamsuddin.

Thank you very much to the people of Japan, the Government of Japan, and the Ministry of Education, Culture, Sports, Science and Technology (MEXT) for giving the author the opportunity and support to pursue higher studies in Japan.

The author would like to convey his sincerest respect to his supervisor: Professor Atsuo Maki who never stops encouraging his students to come up with many adventurous ideas. His energetic attitude and enthusiasm, along with his strong urge to produce a lot of papers, have helped the author to improve his own academic aptness. Without his support and patience, the author would have never reached the current state of academic competence and achievement.

The author also would like to extend his thanks and sincerest appreciation to his seniors and colleagues. To Dr. Yoshiki Miyauchi, for his quick thinking, reasoning, and his deep understanding of CFD, the author learned a lot from him. To D3 Sreenath Subramaniam who has motivated the author and listened to many of the author's unnecessary complaints (remark: the author is not really sorry for this, thank you). To Dr. Myo Zin Aung, who has formed the foundation of the author's current competence and who has guided the author since the start of his time in Japan. To D2 Hiroaki Koike who has tirelessly helped the author in many emergency situations. To Nozomi Amano, who has helped the author in conducting experiments and brought fun to the laboratory. To Yuta Fueki, who was never absent in the author's experiments and who always brings the fun out in everything; more importantly, Yuta Fueki is the author's Japanese teacher and consultant when finding a job. To Yusuke Aoki, who has helped the author in the CFD simulations and brought laughter together with Amano at the laboratory. This sincerest appreciation also goes to the author's batch mates and families. Finally, the author is truly grateful for his fiancée Yoshinta Puspitasari who brings joy and happiness to the author with her antics.

"Don't compare! Don't expect too fast! Don't worry! Be kind to yourself!"

– Tomo Fujita, guitar professor at Berklee College of Music

May 2023, Osaka
Dimas

Contents

List of Figures xi

List of Tables xiii

List of Greek Symbols xv

List of Alphabet Symbols xvii

List of Acronyms xxvii

I General Introduction 1
- 1.1 Some Definitions and Motivation . . . 1
  - 1.1.1 Planning a Trajectory: Trajectory Optimization . . . 2
  - 1.1.2 Maneuvering the Ship: Motion Control . . . 5
    - 1.1.2.1 Real-time Application of CMA-ES for Solving a Kind of NMPC . . . 6
    - 1.1.2.2 Low-speed Positioning System with VecTwin Rudder . . . 6
- 1.2 Contributions and Publications . . . 7
  - 1.2.1 Peer-reviewed Publications . . . 9
  - 1.2.2 Non Peer-reviewed Conferences . . . 9
- 1.3 Thesis Outlines . . . 9

II Dynamics of Maneuvering at a Low Speed 11
- 2.1 Kinematics . . . 11
- 2.2 Kinetics . . . 12
- 2.3 Description of the Forces . . . 13
  - 2.3.1 Hull Forces at a Low Speed . . . 13
  - 2.3.2 Propeller Forces . . . 14
  - 2.3.3 Rudder Forces . . . 16
  - 2.3.4 Wind Forces . . . 17
- 2.4 Dynamics . . . 17

Appendices 19
- II.A Principal particulars of the subject ships: the EOShip and the VTShip . . . 19

III Efficient Planning of a Safe Docking Trajectory 21
- 3.1 Introduction . . . 21
  - 3.1.1 Some Key Points . . . 22
  - 3.1.2 Proposed Scheme . . . 23
- 3.2 Offline Trajectory Planner: Off-TP . . . 24
- 3.3 Semionline Trajectory Planner: SO-TP . . . 25

3.3.1 Hermite-Simpson Collocation . . . 26
3.3.1.1 Hermite Cubic Interpolation . . . 26
3.3.1.2 Simpson Quadrature . . . 28
3.3.1.3 Collocation Method: Transcription . . . 29
3.3.2 Point-in-Polygon: Allision Constraints . . . 31
3.3.3 Transcribed OCP: the NLP in the SO-TP . . . 33
3.3.4 Conditions for Simulations . . . 34
3.3.5 Defining the Harbor's Wall and the Ship's Points . . . 35
3.3.6 Discretization and Interpolation . . . 35
3.4 Result of the Simulations and Evaluation . . . 37
3.4.1 Case M1: Same Initial States with no Wind . . . 38
3.4.2 Case M2: Farther Initial States with Wind Blows Away from the Berth . . . 38
3.4.3 Case M3: Closer Initial States with Wind Blows to the Berth . . . 41
3.4.4 Case M4: Starts Parallel with Wind Blows Away from Berth . . . 41
3.4.5 Additional Cases . . . 44
3.4.6 The Significance of the Off-TP in the Scheme . . . 48
3.5 Concluding Remark . . . 48

Appendices **51**
III.A Hermite Interpolation Polynomial: Coefficients . . . 51
III.B Interpolation at the Mid-segment Point . . . 53
III.C Simpson Quadrature Constraints: Implicit Integration . . . 54

IV Motion Control with Evolution Strategy **55**
4.1 Introduction . . . 55
4.2 Scopes . . . 56
4.2.1 Trajectory Tracking: Complete Docking Operation . . . 56
4.2.2 Path Following: Harbor-entrance Operation . . . 57
4.3 Formulation of the OCP for Motion Control . . . 57
4.4 Formulation of the NLP for Motion Control . . . 58
4.5 Simulation: Complete Docking Operation . . . 59
4.5.1 Simulation Condition . . . 61
4.5.2 Simulation Result . . . 63
4.6 Simulation: Harbor-entrance Operation . . . 64
4.7 Concluding Remark . . . 67

V Low-speed Positioning System with VecTwin Rudder **69**
5.1 Introduction . . . 69
5.1.1 On the Design of the VTPS . . . 70
5.1.2 Scopes . . . 71
5.2 Commands-Forces Relationship . . . 71
5.2.1 Propeller-VecTwin-Hull Interactions . . . 72
5.2.1.1 CFD Simulations . . . 72
5.2.1.2 On the Results of Simulations . . . 73
5.2.1.3 Regression: Linearity Around the Hover Rudder Angle . . . 75
5.2.2 Bow Thruster . . . 78

5.2.3 Complete Commands-Forces Relationship 78
5.3 Control Forces Allocation 79
5.4 PID Controller to Regulate the Required Forces 80
5.5 Guidance: Selecting the Desired Pose 83
5.6 Implementation of the VTPS: Experiment Setup 84
5.7 Combined Automatic Docking and Position/Station keeping Experiments . 86
5.7.1 Automatic Docking Experiments: Positioning Operation 86
5.7.2 Position/Station keeping Experiments 88
5.7.3 Results of the Experiments 88
5.7.3.1 Experiment A 88
5.7.3.2 Experiment B 91
5.7.3.3 Experiment C 94
5.8 Concluding Remark 96

Appendices **99**
V.A On the linearity assumption for a wider range of rudder angle 99
V.B User Interface of the VTPS in MATLAB Simulink 100

VI General Conclusion **103**
6.1 Summary 103
6.2 Future Work 105

Bibliography **107**

大阪大学
OSAKA UNIVERSITY

List of Figures

1-1 Continuity and smoothness of a path, wp: waypoints. 3
1-2 VecTwin rudder and the possible maneuvers. 7
1-3 The subject ships. 8

2-1 Coordinate systems. 12
2-2 The definition of quadrants. 14

3-1 Overviews of the proposed scheme to efficiently plan a trajectory. 23
3-2 Hermite-Simpson collocation method: match the slopes. 30
3-3 A point $\mathbf{p}_\mathrm{s}$ is inside the polygon if the sum of all $\theta_{\mathrm{s},i}$ (for $i = 1, 2, \ldots, n_\mathrm{v}$) is equal to 2π. 32
3-4 The geometry of the Inukai experiment pond and the ship when generating allision constraints. Scales are different. 36
3-5 Visualization and optimal control for case M1 with computational time of 0.478 s. 39
3-6 Summary of optimal state trajectory for case M1 with computational time of 0.478 s. 39
3-7 Visualization and optimal control for case M2 with computational time of 0.645 s. 40
3-8 Summary of optimal state trajectory for case M2 with computational time of 0.645 s. 40
3-9 Visualization and optimal control for case M3 with computational time of 8.999 s. 42
3-10 Summary of optimal state trajectory for case M3 with computational time of 8.999 s. 42
3-11 Visualization and optimal control for case M4 with computational time of 11.664 s. 43
3-12 Summary of optimal state trajectory for case M4 with computational time of 11.664 s. 43
3-13 Visualizations of the optimal trajectory for case A1, A2, and A3. See tables 3-3 and 3-4 for a more detailed comparison. 44
3-14 Visualizations of the optimal trajectory for case A4, A5, and A6. 44
3-15 Visualizations of the optimal trajectory for case A7, A8, A9, and A10. See tables 3-3 and 3-4 for a more detailed comparison. 45
3-16 Comparison of the computational time between the SO-TP with a warm start and without a warm start. 45

4-1 Objective of the OCP for motion control. 58
4-2 The reference trajectory in the complete docking operation. 60
4-3 Block diagram of the trajectory tracking simulation. 62

4-4 Simulation of trajectory tracking for a complete docking operation with CMA-ES: visualization and control commands histories. 63
4-5 Comparison between the actual states and the reference states in the simulation. B is the EOShip's draft (see table 2-1). 64
4-6 Block diagram of the path following simulation. 65
4-7 Simulation of path following for the harbor-entrance operation with CMA-ES: visualization and control commands histories. 66
4-8 Comparison between the actual states and the reference states in the simulation. B is the EOShip's draft (see table 2-1). 66

5-1 Computational domain of the CFD simulations. 73
5-2 Modeling of the propeller, the rudder, the hull, and the skegs. 73
5-3 Ship-fixed coordinate system and rudder angle definition. 75
5-4 Linearity of X_{CT} around the hover rudder angle. 77
5-5 Linearity of Y_{CT} around the hover rudder angle. 77
5-6 Visualization of the desired pose in both coordinate systems. 81
5-7 Block diagram of the VTPS. Blue lines denote wireless communication. . . 85
5-8 Visualization and time histories of the control commands in experiment A. 89
5-9 Time histories of the states of the VTShip in experiment A. 89
5-10 Desired pose in the ship-fixed coordinate system: the transformed error $\mathbf{e}$, and the required forces $\mathbf{f}_{\mathrm{req}}$ in experiment A. 90
5-11 Deviation of the pose of $\boldsymbol{\eta}$ from the desired pose $\boldsymbol{\eta}_{\mathrm{d},0}$ during position/station keeping experiment in experiment A. 90
5-12 Deviation of the position (x_0, y_0) from the desired position $(x_{\mathrm{d},0}, y_{\mathrm{d},0})$ during position/station keeping experiment in experiment A. 90
5-13 Visualization and time histories of the control commands in experiment B. 92
5-14 Time histories of the states of the VTShip in experiment B. 92
5-15 Desired pose in the ship-fixed coordinate system: the transformed error $\mathbf{e}$, and the required forces $\mathbf{f}_{\mathrm{req}}$ in experiment B. 93
5-16 Deviation of the pose of $\boldsymbol{\eta}$ from the desired pose $\boldsymbol{\eta}_{\mathrm{d},0}$ during position/station keeping experiment in experiment B. 93
5-17 Deviation of the position (x_0, y_0) from the desired position $(x_{\mathrm{d},0}, y_{\mathrm{d},0})$ during position/station keeping experiment in experiment B. 93
5-18 Visualization and time histories of the control commands in experiment C. 94
5-19 Time histories of the states of the VTShip in experiment C. 95
5-20 Desired pose in the ship-fixed coordinate system: the transformed error $\mathbf{e}$, and the required forces $\mathbf{f}_{\mathrm{req}}$ in experiment C. 95
5-21 Interpolated surface from the results of additional CFD simulations and the regression plane (first row of $\tilde{\mathbf{V}}$ for rudder angle between 60 deg to 105 deg). 99
5-22 Interpolated surface from the results of additional CFD simulations and the regression plane (second row of $\tilde{\mathbf{V}}$ for rudder angle between 60 deg to 105 deg). 100
5-23 Dashboard of the VTPS in MATLAB Simulink: the user interface. 101
5-24 Real-time monitoring of the pose during the stages of the experiments. . . 102

List of Tables

1-1 Differences between path planning, path smoothing, and trajectory planning. 3
1-2 Differences between the common motion control objectives. 5

2-1 Particulars of the EOShip: a 1/108 scale model. 19
2-2 Particulars of the VTShip: a 1:50 scale model. 19

3-1 Limiting/saturation values for the control command vector when generating the trajectory. 26
3-2 Cases in evaluating the performance of the SO-TP. 35
3-3 Comparison of SO-TP: with a warm start from the Off-TP and without a warm start (16 GB RAM, 8-core processor laptop). 46
3-4 Comparison of the SO-TP: warm-started by the Off-TP and warm-started by the cold-started SO-TP (with no warm start/linear guess) for case M1. 47

4-1 Parameters used in the simulation of the trajectory tracking with CMA-ES. 62

5-1 Summary of the total forces from the CFD simulations at $n = 10\,\text{rps}$. . . . 75
5-2 State measurements on the VTShip. 85
5-3 Ranges of the actuators in the transition stage: LOS guidance + PID rudder controller. 86
5-4 Ranges of the actuators for the VTPS in the docking stage and the position/station keeping experiment. 87
5-5 PID gains in the VTPS (docking stage and position/station keeping experiment). 87
5-6 Saturation values for the required forces $\mathbf{f}_{\text{req}}$ from the PID controller in the VTPS (docking stage and position/station keeping experiment). 87

大阪大学 OSAKA UNIVERSITY

List of Greek Symbols

Symbol	Unit	Description	Page
α_R	rad or deg	effective inflow angle; the angle between U_R and the centerline of the rudder	16
β	rad or deg	drift angle, the angle between the advance speed U and the surge velocity u, $\beta = \mathrm{atan2}\left(-v_\mathrm{m}/u\right)$	11
χ	rad or deg	angle between the true wind speed U_T from the x_0-axis	25, 62
δ	rad or deg	rudder angle	16
Δk		step range to determine the next waypoint (desired pose) from the array $\mathbf{P}$ in the VTPS	84, 88
δ_max	rad or deg	maximum/saturation value of the rudder angle δ	26, 62
δ_p	rad or deg	angle of the port rudder of the VecTwin rudder	71, 86, 87
δ_ph	rad or deg	angle of the port rudder at a hovering condition	76
δ_s	rad or deg	angle of the starboard rudder of the VecTwin rudder	71, 86, 87
δ_sh	rad or deg	angle of the starboard rudder at a hovering condition	76
$\Delta\bar{t}$	s	prediction horizon in the NMPC such that $\bar{t}_j \leq t \leq \bar{t}_j + \Delta\bar{t}$	57, 62
ε		ratio between the wake fraction at the rudder position and that at the propeller position	16
$\boldsymbol{\eta}$		pose of the ship in the earth-fixed coordinate system	11, 80
$\boldsymbol{\eta}_\mathrm{d,0}$		desired pose in the earth-fixed coordinate system	80
$\tilde{\boldsymbol{\eta}}_\mathrm{d,0}$		desired pose in the earth-fixed coordinate system inside the array $\mathbf{P}$	83
η_R		ratio between the diameter of the propeller D_P and the height of the rudder H_R	16
γ_A	rad or deg	angle of attack of the apparent wind speed U_A relative to the surge velocity u	17
γ_N and γ_P		flow straightening coefficients	16
λ		rudder aspect ratio	16

Symbol	Unit	Description	Page
ψ	rad or deg	yaw angle, the angle between ship-fixed and earth-fixed coordinate system	11
ψ_{d}	rad or deg	desired orientation (yaw angle) in the ship-fixed coordinate system	82
$\psi_{\mathrm{d},0}$	rad or deg	desired yaw angle in the earth-fixed coordinate system	80
ρ	$\mathrm{kg/m^3}$	density of the water	13
ρ_{a}	$\mathrm{kg/m^3}$	density of the air	17
$\theta_{\mathrm{s},i}$	rad or deg	angle between two adjacent lines that are formed by connecting an arbitrary point to each of two adjacent polygon vertices	31, 32

List of Alphabet Symbols

Symbol	Unit	Description	Page
A_1 to A_5		linear regression coefficients to estimate the sway force induced by the propeller Y_P in the 3rd and 4th quadrant	15
A_6, A_7, and A_8		polynomial coefficients to estimate the sway force induced by the propeller Y_P in the 2nd quadrant	15
a_H		factor for additional sway force due to the steering	17
A_L	m^2	lateral windage area	17
A_R	m^2	effective area of the rudder	16
A_T	m^2	transversal windage area	17
B_1 to B_5		linear regression coefficients to estimate the yaw moment induced by the propeller Y_P in the 3rd and 4th quadrant	15
B_6, B_7, and B_8		polynomial coefficients to estimate the yaw moment induced by the propeller N_P in the 2nd quadrant	15
C_3, C_6, and C_7		linear regression coefficients to estimate the surge force induced by the propeller X_P in the 3rd and 4th quadrant	15
C_{10}		upper limit of J_s when approximating X_P the 3rd and 4th quadrant	15
C_B		constant for the sway force from the bow thruster, $C_\mathrm{B} = \rho D_\mathrm{B}^4 K_\mathrm{B}$	78
C_D		cross-flow drag resistance coefficient	13
C_N		coefficient to calculate the yaw moment due to the wind	17
C'_P and τ_P		empirical coefficients to estimate w_P	15
C_Pr		correction factor for estimating u_R in the 3rd quadrant	17
C_{rN}		correction factor for the yaw moment	13
C_{rY}		correction factor for the sway force	13
C_X		coefficient to calculate the force in the surge direction due to the wind	17
C_Y		coefficient to calculate the force in the sway direction due to the wind	17

Symbol	Unit	Description	Page
d_s	m	draft of the ship	13, 19
D_B	m	diameter of the blade of the bow thruster	78
K_B		thrust coefficient for the bow thruster	78
$\mathbf{e}$		desired pose that is transformed to the ship-fixed coordinate system, $\mathbf{e} = \begin{bmatrix} x_\mathrm{d} & y_\mathrm{d} & \psi_\mathrm{d} \end{bmatrix}^\top$	82
$\mathbf{e}_0$		deviation of the pose from the desired pose in the earth-fixed coordinate system, $\mathbf{e}_0 = \boldsymbol{\eta}_{\mathrm{d},0} - \boldsymbol{\eta}$	81
$\mathbf{f}$		dynamics, equivalent by definition to the first time-derivative of the states $\mathbf{x}$, i.e., $\mathbf{f} \triangleq \dot{\mathbf{x}}$; a system of ordinary differential equations	18, 25
f_α		rudder lift gradient coefficient	16
$\mathbf{f}_\mathrm{H}$		vector of continuous functions in the collocation method as an approximation to the true dynamics $\mathbf{f}$; equivalent to $\dot{\mathbf{x}}_\mathrm{H}$, i.e., $\mathbf{f}_\mathrm{H} \triangleq \dot{\mathbf{x}}_\mathrm{H}$	27
$\mathbf{f}_{2s}$		dynamics at the mid-segment point when discretizing the OCP, $s = 1, 2, \ldots, N_\mathrm{s}$	30
$\mathbf{f}_\mathrm{c}$	N	vector of the total forces from the actuators, $\mathbf{f}_\mathrm{c} = \begin{bmatrix} X_\mathrm{CT} & Y_\mathrm{CT} & Y_\mathrm{B} \end{bmatrix}^\top$	78
$\mathbf{f}_\mathrm{CT}$	N	vector of the total forces due to the interaction between the propeller, the rudder, and the hull, i.e., $\mathbf{f}_\mathrm{CT} = \begin{bmatrix} X_\mathrm{CT} & Y_\mathrm{CT} \end{bmatrix}^\top$	75, 77
$\mathbf{f}_\mathrm{itcp}$	N	intercept of the regression model; the total forces when $\mathbf{u}_\delta = 0$	75
$\mathbf{f}_\mathrm{req}$		vector of the required forces, $\mathbf{f}_\mathrm{req} = \begin{bmatrix} X_\mathrm{req} & Y_\mathrm{req} & N_\mathrm{req} \end{bmatrix}^\top$	79, 80, 83
F_N	N	rudder normal force	16
h_p	s	length between two consecutive discretization points p when discretizing the OCP, $h_\mathrm{p} = 0.5 h_\mathrm{s}$	29
H_R	m	height of the rudder	16
h_s	s	length of a segment s (in time) when discretizing the OCP, $h_\mathrm{s} = t_\mathrm{f}/N_\mathrm{s}$	26
$I_{z\mathrm{G}}$	$\mathrm{kg} \cdot \mathrm{m}^2$	moment of inertia about the center of gravity x_G	12

Symbol	Unit	Description	Page
J_{mc}		performance/objective function in the NMPC for motion control	57–59, 64
J_{on}		performance/objective function in the SO-TP	25
J_{P}		advance ratio of the propeller, $J_{\mathrm{P}} = (1 - w_{\mathrm{P}})\, u / (nD_{\mathrm{P}})$	15
J_{s}		advance ratio of the propeller when $u < 0$, or $J_{\mathrm{P}} = u / (nD_{\mathrm{P}})$	15
J_z	$\mathrm{kg} \cdot \mathrm{m}^2$	added moment of inertia for the yaw motion	12
k_0, k_1, and k_2		polynomial coefficients to approximate the thrust coefficient K_{T}	15
$\mathbf{K}_{\mathrm{D}}$		positive diagonal gain matrix for the derivative term in the VecTwin positioning system, $\mathbf{K}_{\mathrm{D}} = \mathrm{diag}\,(\, K_{\mathrm{DX}},\, K_{\mathrm{DY}},\, K_{\mathrm{DN}}\,)$	83, 87
$\mathbf{K}_{\mathrm{I}}$		positive diagonal gain matrix for the integral term in the VecTwin positioning system, $\mathbf{K}_{\mathrm{I}} = \mathrm{diag}\,(\, K_{\mathrm{IX}},\, K_{\mathrm{IY}},\, K_{\mathrm{IN}}\,)$	83, 87
$\mathbf{K}_{\mathrm{P}}$		positive diagonal gain matrix for the proportional term in the VecTwin positioning system, $\mathbf{K}_{\mathrm{P}} = \mathrm{diag}\,(\, K_{\mathrm{PX}},\, K_{\mathrm{PY}},\, K_{\mathrm{PN}}\,)$	83, 87
K_{T}		thrust coefficient for the propeller	15
k_x		empirical coefficient to estimate u_{R} in the 1^{st} and 2^{nd} quadrant when $n \geq 0$	16
$k_{x\mathrm{r}}$		empirical coefficient to estimate u_{R} in the 3^{rd} quadrant	17
L	m	length between perpendiculars of the ship	13, 19
L_2		Euclidean norm to quantify how different the initial states in the SO-TP from those of the offline solution	34
$\mathbf{d}$		vector of multipliers used in the SO-TP to modify the initial states	34, 35
l_{R}	m	effective longitudinal position of the rudder from the midship	16
m	kg	mass of the ship	12, 19
m_x	kg	added mass of the ship in the surge direction	12
m_y	kg	added mass of the ship in the sway direction	12

Symbol	Unit	Description	Page
n	rps, 1/s	propeller revolution number	13, 86, 87
n_{B}	rps, 1/s	revolution number of the bow thruster	71, 78, 86, 87
$\tilde{n}_{\mathrm{B}}$	rps^2, $1/\mathrm{s}^2$	modified revolution number of the bow thruster	78
N_{p}		number of discretization points p when solving the OCP	29, 35
N_{CT}	$\mathrm{N}\cdot\mathrm{m}$	total yaw moment due to the interaction between the propeller, the rudder, and the hull, from the CFD simulations of bollard pull test; neglected	74
N_{H}	$\mathrm{N}\cdot\mathrm{m}$	hull yaw moment acting about the midship	13
N_{m}	$\mathrm{N}\cdot\mathrm{m}$	total yaw moment about the midship	13
$n_{\max}$	rps, 1/s	maximum/saturation value of the n	26, 62
N_{P}	$\mathrm{N}\cdot\mathrm{m}$	yaw moment acting about the midship induced by the propeller	13–15
N_{R}	$\mathrm{N}\cdot\mathrm{m}$	yaw moment acting about the midship induced by the rudder	13, 17
N'_r		linear hydrodynamic derivative for moment with respect to yaw rate r	14
N_{req}	$\mathrm{N}\cdot\mathrm{m}$	required yaw moment about the x_{G}	79
N_{s}		number of segments when discretizing the OCP	26, 35
n_{s}		number of vertices that form the ship	32, 35
N'_v		linear hydrodynamic derivative for yaw moment with respect to sway velocity v_{m}	14
n_{v}		number of vertices that form the polygon that represents the harbor's boundary	31, 35
N_{W}	$\mathrm{N}\cdot\mathrm{m}$	yaw moment acting about the midship due to the wind	13
P		pitch of the propeller	15
p		p-th discretization point when discretizing the OCP, $p = 1, 2, \ldots, N_{\mathrm{p}}$	29
$\mathbf{P}_{\mathrm{b}}$		an array whose columns are the position of the vertices of the harbor's boundary in the earth-fixed coordinate system, see $\mathbf{p}_{\mathrm{b}}$	31, 35, 36
$\mathbf{p}_{\mathrm{b}}$		position of a vertex of harbor's boundary, i.e., $\mathbf{p}_{\mathrm{b}} = \begin{bmatrix} x_{\mathrm{b}} & y_{\mathrm{b}} \end{bmatrix}^\top$	31
$\mathbf{P}$		array whose rows are the time-ordered reference pose $\tilde{\boldsymbol{\eta}}_{\mathrm{d},0}$ expressed in the earth-fixed coordinate system	83

Symbol	Unit	Description	Page
$\mathbf{P}_\mathrm{s}$		an array whose columns are the position of the points of the ship in the earth-fixed coordinate system, see $\mathbf{p}_\mathrm{s}$	31, 36
$\mathbf{p}_\mathrm{s}$		position of a point of the ship, i.e., $\mathbf{p}_\mathrm{s} = \begin{bmatrix} x_\mathrm{s} & y_\mathrm{s} \end{bmatrix}^\top$	31
r	rad/s, deg/s	yaw rate, $r = \dot{\psi} = \mathrm{d}\psi/\mathrm{d}t$	11
r'		nondimensionalized yaw rate, $r' = rL/U$	13
$\mathbf{R}\,(\psi)$		rotation matrix, an orthogonal transformation $\mathbf{R}^{-1} = \mathbf{R}^\top$	12, 82
s		s-th segment when discretizing the OCP, $s = 1, 2, \ldots, N_\mathrm{s}$	29
t	s	time; the normalized prediction time inside the SO-TP or the OCP in general	12, 25
t_{2s}	s	time at the mid-segment point when discretizing the OCP, $s = 1, 2, \ldots, N_\mathrm{s}$	29
$\bar{t}$	s	actual time outside the SO-TP or the OCP in general	25, 57
T_cs	s	computational time of the SO-TP without a warm start: cold start (a linear guess)	37
t_f	s	final time; the time to reach the final docking point $\mathbf{x}_\mathrm{fin}$	25, 33
$t_\mathrm{f,off}$	s	final time; the time to reach the final docking point $\mathbf{x}_\mathrm{fin}$ in the Off-TP	24
t_k	s	time at the k-th knot point when discretizing the OCP	26
$\mathbf{T}$		invertible configuration matrix in the VecTwin positioning system	80
t_P		thrust deduction factor	15
t_p	s	time at the p-th discretization point when discretizing the OCP	29
t_R		steering resistance deduction factor and propeller thrust increase induced by steering	17
T_w1	s	computational time of the SO-TP with a warm start obtained from the Off-TP	37
U	m/s	advance velocity, the resultant of u and v_m, that is, $U = \sqrt{u^2 + {v_\mathrm{m}}^2}$	11

Symbol	Unit	Description	Page
$\mathbf{u}$		control command vector or just the command or the control; denote the predicted control in the SO-TP; modified control command vector in the VecTwin positioning system	18, 78
u	m/s	surge velocity of the ship with respect to the earth-fixed coordinate system (either at midship or x_{G})	11
$\mathbf{u}^*$		optimal control command vector, $\mathbf{u}^*(t)$ denotes the optimal control	25, 57, 59
$\mathbf{u}^*_{\mathrm{off}}$		optimal control command vector in the Off-TP, $\mathbf{u}^*_{\mathrm{off}}(t)$ denotes the optimal control	24
$\mathbf{u}^*_p$		optimal control command vector at the p-th discretization point	33
U_{A}	m/s	apparent wind speed	17
$\dot{u}_{\mathrm{acc}}$	m/s^2	threshold value for the surge acceleration in the acceleration period to ensure that the propeller is in the forward mode	60, 62
$\mathbf{u}_{\mathrm{c}}$		control command vector in the VecTwin positioning system	71
$\mathbf{u}_\delta$		pair of rudder angles in the VecTwin positioning system, $\mathbf{u}_\delta = \begin{bmatrix} \delta_{\mathrm{p}} & \delta_{\mathrm{s}} \end{bmatrix}^\top$	71
$\mathbf{u}_{\mathrm{H}}$		approximation function of the optimal control within a segment	28, 35, 36
$\mathbf{u}^*_{\mathrm{H}}$		approximation function of the optimal control between two consecutive discretization points after solving the NLP	36
$\mathbf{u}_{2s}$		control at the mid-segment point when discretizing the OCP, $s = 1, 2, \ldots, N_{\mathrm{s}}$	30
$\mathbf{u}_{\delta\mathrm{h}}$	rad or deg	vector of the hover rudder angles, $\mathbf{u}_{\delta\mathrm{h}} = \begin{bmatrix} \delta_{\mathrm{ph}} & \delta_{\mathrm{sh}} \end{bmatrix}^\top$	76
$\tilde{\mathbf{u}}_{\delta\mathrm{h}}$	rad or deg	vector of the rudder angles around the hover rudder angles, $\tilde{\mathbf{u}}_{\delta\mathrm{h}} = \begin{bmatrix} \tilde{\delta}_{\mathrm{p}} & \tilde{\delta}_{\mathrm{s}} \end{bmatrix}^\top \equiv \mathbf{u}_\delta - \mathbf{u}_{\delta\mathrm{h}}$	76
$\dot{u}_{\mathrm{dec}}$	m/s^2	threshold value for the surge acceleration in the deceleration period to ensure that the propeller is in the reverse mode	60–62
$\dot{u}_{\mathrm{lim}}$	m/s^2	threshold value for the surge acceleration to attenuate sudden change of the propeller revolution number	61, 62
u_{P}	m/s	effective inflow velocity from the propeller at the rudder position, $u_{\mathrm{P}} = (1 - w_{\mathrm{P}})\, u$	16

Symbol	Unit	Description	Page
U_{R}	m/s	inflow speed or the resultant of the inflow velocities at the rudder, $U_{\mathrm{R}} = \sqrt{{u_{\mathrm{R}}}^2 + {v_{\mathrm{R}}}^2}$	16
u_{R}	m/s	inflow velocity at the rudder in the surge direction	16, 17
u_{rest}	m/s	threshold value for the surge velocity in the stationing period to ensure that the propeller is off	61, 62
u_{RPr1}	m/s	for estimating u_{R} in the 3$^{\mathrm{rd}}$ quadrant	16
u_{RPr2}	m/s	for estimating u_{R} in the 3$^{\mathrm{rd}}$ quadrant	16
u_{Rsq}	m^2/s^2	for estimating u_{R} in the 3$^{\mathrm{rd}}$ quadrant	16
U_{T}	m/s	true wind speed	17, 25, 62
$\mathbf{V}$		invertible force coefficient matrix in the VecTwin positioning system	79
v	m/s	sway velocity at x_{G} with respect to the earth-fixed coordinate system	81
$\mathbf{v}$		vector of the ship's velocities relative to the earth-fixed coordinate system, either at the midship or at the x_{G}	12, 81
v_{m}	m/s	sway velocity at the midship with respect to the earth-fixed coordinate system	11
v'_{m}		nondimensionalized sway velocity at the midship, $v'_{\mathrm{m}} = v_{\mathrm{m}}/U$	15
v_{R}	m/s	inflow velocity at the rudder in the sway direction	16
$\tilde{\mathbf{V}}$		matrix whose rows are the regression obtained from the multiple linear regression of the forces from the CFD simulations	75
w_{P}		wake fraction coefficient at the position of the propeller x_{P}	15
$w_{\mathrm{P},0}$		wake fraction coefficient at the position of the propeller when $v_{\mathrm{m}} = 0$ and $r = 0$	15
X	N	total forces acting at the midship in the surge direction	13
$\mathbf{x}$		state vector or just the states; denote the predicted states in the SO-TP	18
x	m	longitudinal axis of the ship-fixed coordinate system, the surge direction	11
$\mathbf{x}^*$		optimal state vector, $\mathbf{x}^*(t)$ denotes the optimal state trajectory	25

Symbol	Unit	Description	Page
$\mathbf{x}^*_{\text{off}}$		optimal state vector in the Off-TP, $\mathbf{x}^*_{\text{off}}(t)$ denotes the optimal state trajectory	24, 34
$\mathbf{x}^*_p$		optimal state vector at the p-th discretization point	33
x_0	m	longitudinal position of the ship in the earth-fixed coordinate system (either midship or x_{G})	11, 80
$X'_{0(\text{A})}$		resistance coefficient when moving astern	13
$X'_{0(\text{F})}$		resistance coefficient when moving ahead	13
x_{B}	m	longitudinal position of the bow thruster from the center of gravity of the ship x_{G}	19, 79
x_{b}	m	longitudinal position of a vertex of the harbor's boundary in the earth-fixed coordinate system	31
$\mathbf{x}_{\text{H}}$		vector of Hermite interpolation polynomials in the collocation method as an approximation to the true optimal state trajectory $\mathbf{x}(t)$; also called the interpolant	27
$\dot{\mathbf{x}}_{\text{H}}$		first derivative of the interpolant $\mathbf{x}_{\text{H}}(t)$ in the collocation method, i.e., $\dot{\mathbf{x}}_{\text{H}}(t) = \text{d}\mathbf{x}_{\text{H}}/\text{d}t$; equivalent to the approximating dynamics $\mathbf{f}_{\text{H}}$	27
$\mathbf{x}^*_{\text{H}}$		interpolation polynomial to interpolate the optimal states in a segment after solving the NLP	37
$\mathbf{x}_{2s}$		states at the mid-segment point when discretizing the OCP, $s = 1, 2, \ldots, N_{\text{s}}$	30
X_{CT}	N	total force in the surge direction due to the interaction between the propeller, the rudder, and the hull, from the CFD simulations of bollard pull test	73
x_{d}	m	desired longitudinal position in the ship-fixed coordinate system	82
$x_{\text{d},0}$	m	desired longitudinal position in the earth-fixed coordinate system	80
x_{dec}	m	longitudinal position in the earth-fixed coordinate system that marks the location where the ship is supposed to decelerate	60–62
$\mathbf{x}_{\text{fin}}$		final docking states	24, 34
x_{fR}	m	longitudinal position of the acting point of the propeller-rudder-hull forces from the center of gravity of the ship x_{G}	19, 74
x_{G}	m	longitudinal position of the center of gravity of the ship from the midship	12, 19

Symbol	Unit	Description	Page
X_{H}	N	hull force acting at the midship in the surge direction	13
x_{Hr}	m	longitudinal position of the acting point of the additional sway force due to the steering	17
X_{P}	N	thrust from the propeller, force acting at the midship in the surge direction induced by the propeller	13–15
x_{P}	m	longitudinal position of the propeller from the midship	15
x'_{P}		nondimensionalized position of the propeller $x'_{\mathrm{P}} = x_{\mathrm{P}}/L$	15
X_{R}	N	force acting at the midship in the surge direction induced by the rudder	13, 17
x_{R}	m	longitudinal position of the rudder from the midship	16, 17
$\mathbf{x}_{\mathrm{ref}}$		reference state vector used in motion control, $\mathbf{x}_{\mathrm{ref}}\left(\bar{t}\right)$ denotes the reference state trajectory	57
$\bar{\mathbf{x}}$		actual state vector of the ship outside the SO-TP or the OCP in general	25, 57
X_{req}	N	required force in the surge direction	79
x_{s}	m	longitudinal position of a point of the ship in the earth-fixed coordinate system	31
X'_{vr}		hydrodynamic derivative with respect to sway velocity v_{m} and yaw rate r	14
X_{W}	N	force acting at the midship in the surge direction due to the wind	13
Y	N	total forces acting at the midship in the sway direction	13
y	m	lateral axis of the ship-fixed coordinate system, the sway direction	11
y_0	m	lateral position of the ship in the earth-fixed coordinate system (either midship or x_{G})	11, 80
Y_{B}	N	sway force from the bow thruster	78
y_{b}	m	lateral position of a vertex of the harbor's boundary in the earth-fixed coordinate system	31
Y_{CT}	N	total force in the sway direction due to the interaction between the propeller, the rudder, and the hull, from the CFD simulations of bollard pull test	73

Symbol	Unit	Description	Page
y_{d}	m	desired lateral position in the ship-fixed coordinate system	82
$y_{\mathrm{d},0}$	m	desired lateral position in the earth-fixed coordinate system	80
Y_{H}	N	hull force acting at the midship in the sway direction	13
Y_{P}	N	force acting at the midship in the sway direction induced by the propeller	13–15
Y_{R}	N	force acting at the midship in the sway direction induced by the rudder	13, 17
Y'_r		linear hydrodynamic derivative for sway force with respect to yaw rate r	14
Y_{req}	N	required force in the sway direction	79
y_{s}	m	lateral position of a point of the ship in the earth-fixed coordinate system	31
Y'_v		linear hydrodynamic derivative for sway force with respect to sway velocity v_{m}	14
Y_{W}	N	force acting at the midship in the sway direction due to the wind	13
$\mathbf{Z}$		inverse of the multiplication between $\mathbf{T}$ and $\mathbf{V}$, $\mathbf{Z} = (\mathbf{TV})^{-1}$	80

List of Acronyms

Symbol	Description	Page
AS	autonomous ship	1
BVP	boundary value problem	25
CFD	computational fluid dynamics	8, 70
CMA-ES	covariance matrix adaptation evolution strategy	8, 24, 56
DP	dynamic positioning	6, 69
DPS	dynamic positioning system	6, 7, 69
GNSS	global navigation satellite system	84
IMO	International Maritime Organization	1
IPM	interior point method	4
IVP	initial value problem	29
LOS	line-of-sight	86
MASS	maritime autonomous surface ship	1, 49
MMG	Maneuvering Modeling Group	12
NLP	nonlinear programming	4, 33, 59
NMPC	nonlinear model predictive control	5, 6, 55
OCP	optimal control problem	4, 6, 8, 21, 25, 57
ODE	ordinary differential equation	11, 18
Off-TP	offline trajectory planner	23
PID	proportional, integral, and derivative	5–7
ROS	robotic operating system	85
SO-TP	semionline trajectory planner	22, 33
SQP	sequential quadratic programming	4, 24
VTPS	VecTwin positioning system	69

I

General Introduction

This chapter describes the introduction of this thesis. It begins with an explanation of the general idea of an autonomous ship, including the two practical problems on the topic of ship's automatic docking that motivate the study. Then, it describes the list of contributions with the corresponding academic publications that have been published. Finally, the chapter ends with an overview of the organization of the thesis.

1.1 Some Definitions and Motivation

In robotic or machinery, the terms *automation* and *autonomization* are two different things that often create misconceptions. The purpose of automation is to reduce human involvement in a process and its product is an *automatic* process. Autonomization, on the other hand, seeks to eliminate the need for human involvement up to the system level. The product of autonomization is an *autonomous* system that is capable of automating its process and making its own decision. An automatic process does not imply an autonomous system, but an autonomous system always implies that it has at least one automatic process. Thus, the term autonomous refers to a broader concept of automation. Moreover, one can say that an autonomous system is both *automatic* and *intelligent.*

In the maritime field, the generally accepted idea of a maritime autonomous surface ship, abbreviated as MASS or just *autonomous ship* (AS), by the IMO (International Maritime Organization) [1] is as follows: a MASS is *a ship which, to a varying degree, can operate independent of human interaction.* This idea led to the categorization of autonomous ships based on the degree of autonomy (the level of human interventions) by the IMO [1] as follows:

AS degree 1: Seafarers are on board to operate and control shipboard systems and functions. Some operations may be automated and at times be unsupervised but with seafarers on board ready to take control.

AS degree 2: Remotely controlled ship *with* seafarers on board. The ship is controlled and operated from another location. Seafarers are available on board to take control and operate the shipboard systems and functions.

AS degree 3: Remotely controlled ship *without* seafarers on board. The ship is controlled and operated from another location. There are no seafarers on board.

AS degree 4: Fully autonomous ship. The operating system of the ship is able to make decisions and determine actions by itself.

Each degree has its own characteristics and limitations that determine the autonomy of the ship. A higher degree of autonomy must satisfy the necessary criteria of all the preceding degrees. For example, an **AS degree 4** must be remotely controllable; which is a criterion of both **AS degree 2** and **AS degree 3** for emergency measures. Note that there are also other categorizations of autonomous ships (level of autonomy) proposed by experts in the art such as [2, 3].

Regardless of the degrees of autonomy, one of the aspects of realizing an autonomous ship is automating a docking/berthing[1] operation. A docking operation must be executed at a very low speed with careful assessment of the situation of the environment (e.g., the presence of obstacles or sudden wind change). It is a very intricate operation that strictly depends on the type of the ship and its maneuverability at a very low speed. This maneuverability is closely related to the *actuation* of the ship.

A ship is said to be *fully actuated* if it has at least one *actuator* that directly accelerates the ship in a certain direction. For example, a ship that is equipped with a main propeller and several side thrusters is fully actuated. An *underactuated* ship, on the other hand, has no side thruster that can accelerate the ship in the sway direction (laterally); it is equipped with just a main propeller and a rudder. For an underactuated ship, the low-speed criterion of a docking operation requires the thrust from the propeller to be as minimal as possible. This significantly decreases the steering forces from the rudder and thus the maneuverability of the ship, which in turn makes the automation of a docking operation more challenging. Nevertheless, the topic of automatic docking operation for underactuated ships has seen significant development in the past few decades as reported in [4–14]. The concern of underactuated can be eliminated by making the ship fully or even overly actuated in exchange for high installation and maintenance costs. A fully/overly actuated ship gives more ease and flexibility to perform automatic docking as reported in [15–21]. We shall discuss these developments in a dedicated chapter later.

Regardless of the actuation of the ship, a manual docking operation led by a ship-master typically evolves around two problems: (1) planning a *trajectory* or *path* of the docking operation and (2) maneuvering/moving the ship slowly and carefully according to the plan until finally stops at the designated docking point. Automating a docking operation is also similar; generally, it evolves around the same two problems that must be executed in an efficient and systematic way. The following subsections will describe each problem in a general manner.

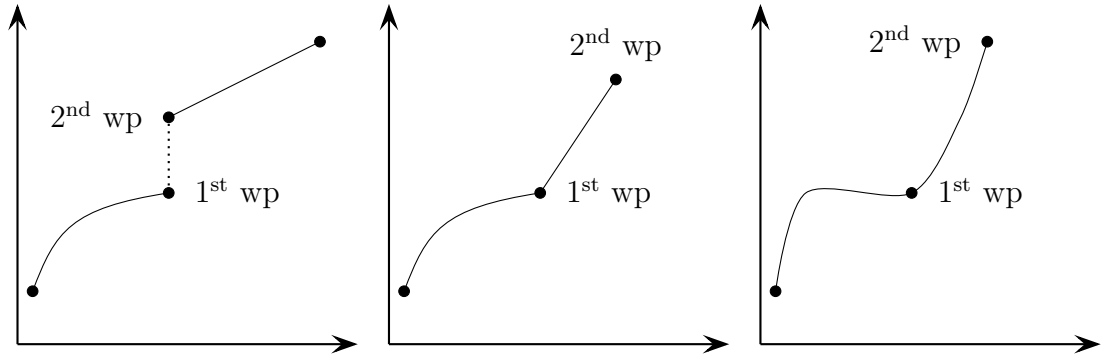
1.1.1 Planning a Trajectory: Trajectory Optimization

Let us begin by explaining the definition of a trajectory and a path in robotics. The simplest definition of a path is a *continuous* geometrical object that is formed by a sequence of connected *waypoints* [22–24]. These waypoints describe the locations ***where*** the ship should be as the operation progresses. A path can be generated solely based on the geometric information of the surrounding through a procedure called *path planning*. If the *kinematics*[2] of the system is also considered in the procedure, the path is said to

[1] A berth is a designated place within a dock. Docking means maneuvering the ship until it is fixed safely on a structure that connects the ship to the land. Thus, docking refers to a more general idea.

[2] Kinematics describes how a body moves regardless of the reason why it moves (forces). Kinematics refers to a set of equations that relate a body's position to its velocity without considering its mass.

be *smoothed*, and the procedure is called *path smoothing* [25, 26]. The visualization of continuity and smoothness is shown in figure 1-1.



(a) Discontinuous, not a path (b) Continuous and nonsmooth (c) Continuous and smooth

Figure 1-1. Continuity and smoothness of a path, wp: waypoints.

Moreover, if the waypoints that made up the smoothed path are scheduled, that is, parameterized by time, then they form a trajectory. Hence, a trajectory describes ***where***, ***how***, and ***when*** the ship should be situated as time progresses. A trajectory can also be interpreted as the ***time history of something***. Generally, generating a trajectory is a more sophisticated procedure because the geometric information, the kinematics, and the *kinetics*[3] of the system are taken into account [23, 24]. The procedure of generating a trajectory is known as *trajectory planning*. The differences between the three procedures are summarized in table 1-1.

Table 1-1. Differences between path planning, path smoothing, and trajectory planning.

Procedure	Consideration			Result
	Geometric	Kinematics	Kinetics	
Path planning	✓	✗	✗	path: sequence of waypoints
Path smoothing	✓	✓	✗	smoothed path
Trajectory planning	✓	✓	✓	time-parameterized smooth path

When preparing for a docking operation, adept and experienced shipmasters must understand the capability of the ship that they are piloting (maneuverability, actuation, etc.). They also need to know sufficient information about the harbor where the ship must dock, e.g., the harbor geometry and the existence of other ships. Moreover, the shipmaster must be able to make decisive actions against sudden changes in environmental conditions. Equipped with these skills, the shipmaster knows where, how, and when the ship should be situated during the entire docking operation. From these explanations, obviously planning a trajectory for a docking operation is preferred to planning just a path. Since a trajectory is just a path generated with more considerations, it can also act as a smooth path when it is needed.

[3]Kinetics describes the characteristics of how a mass accelerates due to the forces. The kinematics and the kinetics form the *dynamics* of a body. Modern physicists use both the terms kinetics and dynamics interchangeably to refer to the same thing. However, in this thesis, we distinguish each term.

In an autonomous ship, instead of the shipmaster, the trajectory planning of a docking operation is handled by a component called a *trajectory planner*. This trajectory planner must be capable of acting like a shipmaster whose task is to plan the trajectory by considering the following points:

- maneuverability and actuation of the ship (determined by the dynamics of the ship),
- geometric information of the surrounding,
- environmental forces (the dynamics of the environment),
- computationally fast enough to react to the change in environmental condition.

The ability of an autonomous ship to plan a trajectory by considering the above points is one of many aspects that makes it *intelligent*.

The question now is how to mathematically incorporate all of the above points in the trajectory planner. When the mathematical model of the dynamics of a system is known, a trajectory can be generated by solving a case of open-loop continuous OCP (optimal control problem) known as *trajectory optimization* problem [27–29]. The very general goal of an OCP is to *determine the control signals that cause a system to satisfy the given constraints such that a performance criterion is minimized or maximized* [30]. Examples of the constraints here are the dynamics of the system and geometric constraints. In the case of a trajectory optimization problem, the goal is to find the continuous time histories of control commands that result in the trajectories (time histories) of state variables such that a performance criterion is optimized, without violating the given constraints.

A *state variable* refers to a physical quantity of the position (the *where*), the pose (yaw, roll, pitch), and the velocities (the *how*) of the system. A *state vector* (or just *states*) is a set of state variables. Moreover, a control command history that satisfies all the control constraints is termed an *admissible/feasible control*. Similarly, a state trajectory that satisfies all the state variables constraints is termed an *admissible/feasible state trajectory*. If a feasible control does optimize the performance criterion, it is termed *optimal control*, and the corresponding feasible state trajectory is termed *optimal state trajectory*[4]. These two are the solution that we seek upon solving a trajectory optimization problem.

Practically, a continuous trajectory optimization problem is solved directly by transcribing it to a finite-dimensional NLP (nonlinear programming) [28,29]. The NLP is then commonly solved by a gradient-based (Newton's method: a local optimization) solver like SQP (sequential quadratic programming) or IPM (interior point method) [31]. A gradient-based method is really sensitive to where the gradient descent starts, i.e., the initial guess of the solution. A bad initial guess can result in the solver being computationally expensive or even unable to converge to the solution (an *ill-conditioned* process). These issues are amplified when the mathematical model of the dynamics is highly nonlinear and intricate, e.g., dynamics of an underactuated ship at a very low speed [10, 14, 32], which may result in a nonsmooth search space with many discontinuities. This thesis discusses a possible way to circumvent these issues that arise due to the nonlinearity such that one can achieve the balance between (1) the feasibility of the solution with respect to the dynamics and the geometric constraints and (2) the computational cost.

[4]It is worth noting that, even in its singular form, all the terms here generally can refer to both single and multiple variables, e.g., control command describes both a single control command or a set of control commands, an optimal control describes the time histories of one or more control commands, and a state trajectory describes the trajectories (time histories) of one or more state variables.

大阪大学 OSAKA UNIVERSITY

1.1.2 Maneuvering the Ship: Motion Control

An autonomous ship must be able to maneuver and control itself based on the planned trajectory/path. In the field of robotics, this is a case of *motion control* problem. Motion control discusses the design of a *controller* (or *control algorithm*) that determines the necessary control command to achieve the given objective [33] under external disturbances. There are five most common objectives of motion control [33, 34] in autonomous ships namely, *setpoint stabilization*, *position keeping*, *path following*, *positioning*, and *trajectory tracking*.

A setpoint stabilization is when a ship needs to maintain itself at a constant desired state value (a *setpoint*: a reference value). Examples of setpoint stabilization are course keeping [35–37] and speed control [37]. If there are multiple constant setpoints, e.g., a stationary position and pose, then it is position/station keeping [38,39]. When a ship has to maintain itself along a geometric path, then it is a path following [40–42]. Moreover, a path following that is executed at a low speed is known simply as *positioning* [33]. On the other hand, when the ship has to be at some certain states and at a scheduled time such that it *tracks* a reference trajectory, then it is a trajectory tracking [42–44], which is the most intricate of the five objectives. The differences between the five objectives are summarized in table 1-2. In addition, the main differences between path following and trajectory tracking are how the setpoint/reference value is determined and how it is constrained by time.

Table 1-2. Differences between the common motion control objectives.

Objective	Consideration			Note
	States	Time-constrained	Constant	
Setpoint stabilization	single	no	yes	the smallest objective that constructs the others
Position keeping	multiple	no	yes	e.g., stationary position and constant pose
Path following	multiple	no	no	can be a sequence of setpoint stabilization
Positioning	multiple	no	no	path following that is conducted at a low speed
Trajectory tracking	multiple	yes	no	the ship states must be at the scheduled time

In an automatic docking operation, because of the low-speed requirement, the motion control objective (or simply, *control objective*) can be either trajectory tracking or positioning. The complicated dynamics of the ship at a low speed, coupled with underactuated, limit the available choice of mathematical tools that can be utilized when designing the controller. Several methods—regardless of the actuation–have been implemented such as NMPC (nonlinear model predictive control) [8,11,16], backstepping [5,45], ANN (artificial neural network) [6,7,17], and PID (proportional, integral, and derivative) controller [9,13,21]. This thesis, however, limits the discussion within the scope of NMPC for trajectory optimization and PID controller for positioning; both implemented in automatic docking operation.

1.1.2.1 Real-time Application of CMA-ES for Solving a Kind of NMPC

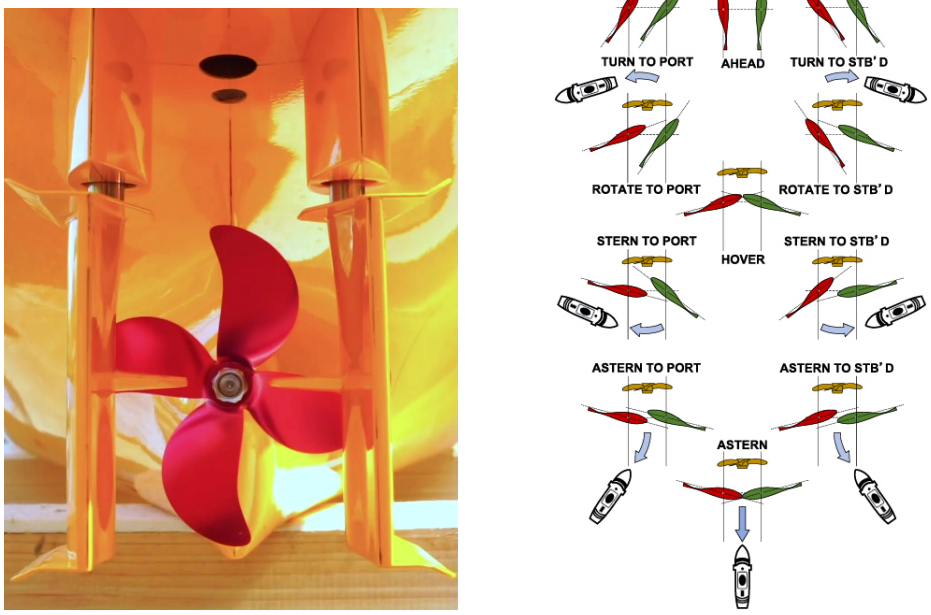
In principle, NMPC is a bridge between open-loop OCP (e.g., trajectory optimization) and closed-loop feedback control. It performs trajectory optimization regularly at a certain period, where at each instant, the present states of the ship are considered (through measurement) and the desired final states are not changed. Then, only the first control command of the optimal control is sent to the ship which drives the ship to new states. The new states are then fed back to the trajectory optimization (NMPC). This whole process is repeated as a closed loop where the input is the measured states and the output is the first control command of the optimal control.

As mentioned at the end of section 1.1.1, a highly nonlinear trajectory optimization problem may result in a search space that is nonsmooth with multiple discontinuities and many local optima (rugged). When a good initial guess of the solution is not correctly provided (ill-conditioned), the condition of the search space may result in the gradient-based solver being *stuck* in a local optimum. This is because the solver relies on the information of the gradient, which sometimes has to be approximated numerically. To overcome this issue, this thesis investigates the possibility to implement a gradient-free and global nonlinear optimization solver known as CMA-ES (covariance matrix adaptation evolution strategy) [46], as the algorithm in the NMPC for trajectory tracking and path following of automatic docking operation. This solver does not require an initial guess and the information of the gradient, which is suitable for an ill-conditioned and highly nonlinear problem like an automatic docking of an underactuated ship.

1.1.2.2 Low-speed Positioning System with VecTwin Rudder

The ability of a ship to achieve the control objectives in automatic docking operation depends on its maneuverability at a low speed. As mentioned at the beginning of this chapter, maneuverability is closely related to the actuation of the ship. A fully actuated ship equipped with multiple thrusters is highly maneuverable at a low speed. This kind of ship is usually known as a DP (dynamic positioning) ship. The term dynamic positioning refers to the low-speed maneuvers that are done by means of the ship's own propulsion system (thrusters) [33, 39, 47]. A DP ship has a system known as DPS (dynamic positioning system) that is capable of automating the desired low-speed maneuvers. The DPS allocates the forces (hence the term *dynamic* in the name) from all available thrusters to fulfill the forces required to execute the desired maneuvers. This allocation process is commonly known as *control/thrust allocation* where the required forces can be regulated by traditional decoupled PID controller [33, 39].

As an alternative to the usual DPS, this thesis proposes an approach to design a novel low-speed positioning system that utilizes a high-performance rudder known as VecTwin rudder (designed by Japan Hamworthy & Co., Ltd. [48]). VecTwin rudder comprises twin fishtail rudders with reaction fins attached to increase the overall performance (see figure 1-2a). Each rudder can move within a wide range independently; each complements the other. A ship equipped with a VecTwin rudder is capable of executing low-speed maneuvers like hover, crabbing, rotation, and reversal (see figure 1-2b). These maneuvers can be executed without auxiliary side thrusters, and at a constant propeller revolution



(a) VecTwin rudder

(b) Possible maneuvers with VecTwin rudder

Figure 1-2. VecTwin rudder and the possible maneuvers.

number in forward mode. With at least one auxiliary side thruster, e.g., bow thruster, a pure sway maneuver can also be executed. Note that the proposed arrangement featuring a fixed-pitch propeller, VecTwin rudder, and one bow thruster does not make the ship fully actuated, because there is no actuator that can directly accelerate the ship laterally.

The proposed positioning system still implements the simple decoupled PID controller that regulates the forces required to execute the maneuvers. Moreover, the constant propeller revolution effectively renders the propeller unavailable as an actuator. Hence, in contrast to the usual DPS, the positioning system with such an arrangement relies mainly on the VecTwin rudder to allocate the constant thrust from the propeller to meet the required forces, supported by the bow thruster. The ability of this positioning system to execute low-speed maneuvers makes it suitable to be implemented in an automatic docking operation. Due to its simplicity, it can also be an addition to the usual DPS.

1.2 Contributions and Publications

There are many problems in the discussion of ship's automatic docking. This thesis discusses two of the main problems: trajectory planning and motion control for automatic docking. Through this thesis, the author proposes workarounds for the problems. These include an efficient scheme to plan a safe optimal state trajectory, a controller for trajectory tracking, and a low-speed positioning system; all applicable to underactuated ships. These contributions with the corresponding academic publications are summarized below.



(a) Single-rudder single-propeller underactuated scale model ship: EOShip



(b) Scale model ship equipped with VecTwin rudder, a bow thruster, and a propeller: VTShip

Figure 1-3. The subject ships.

Contrb. I: A fast OCP-based trajectory planner that can take into account the geometric information and wind dynamics in the harbor. The computational time is reduced by using a near-globally optimal solution as a warm start for the solver. The resulting feasible trajectory is guaranteed to be safe from allision and feasible with respect to the dynamics of an underactuated ship and the wind. The trajectory planner was tested in several simulations with a mathematical model of an underactuated ship named EOShip (see figure 1-3a). The simulations were done under different conditions but with the same warm start. This is a huge improvement to the previous studies by the author's colleagues who developed planners that require heavy computational cost. This contribution was presented at a conference as **publication [D]** and made public in **publication [B]**.

Contrb. II: Motion control for automatic docking operations that implements CMA-ES: a gradient-free and global nonlinear optimization solver that is suitable for an ill-conditioned problem with nonsmooth and rugged search space. It was tested in several simulations with the mathematical model of EOShip. To the limit of the author's knowledge, this is the first attempt to implement an evolution strategy in motion control of a ship. This contribution was presented at a conference as **publication [A]**.

Contrb. III: A simple and practical low-speed positioning system that utilizes a high-performance VecTwin rudder. The system also utilizes a bow thruster whereas the propeller is at a constant revolution number in forward mode. Because of the limited resources to conduct experiments of bollard pull tests, the relationship between the control command and the resulting forces was obtained via linear regression of several CFD (computational fluid dynamics) simulations. The system was tested in multiple experiments with a scale model ship named VTShip (see figure 1-3b) in a pond where environmental disturbances exist. This contribution was presented at a conference as **publication [E]** and made public in **publication [C]**.

1.2.1 Peer-reviewed Publications

Conference Publication

[**A**] D. M. Rachman, Y. Miyauchi, N. Umeda, and A. Maki, "Feasibility study on the use of evolution strategy: CMA-ES for ship automatic docking problem," in *Proc. 1st International Conference on the Stability and Safety of Ships and Ocean Vehicles*, 2021, STAB&S 2021 (presented online).

Journal Publication

[**B**] D. M. Rachman, A. Maki, Y. Miyauchi, and N. Umeda, "Warm-started semionline trajectory planner for ship's automatic docking (berthing)," *Ocean Engineering*, vol.252, p. 111127, 2022. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0029801822005388`

[**C**] D. M. Rachman, Y. Aoki, Y. Miyauchi, N. Umeda, and A. Maki, "Experimental low-speed positioning system with VecTwin rudder for automatic docking (berthing)," *Journal of Marine Science and Technology*, Jul 2023. [Online]. Available: `https://doi.org/10.1007/s00773-023-00950-9`

1.2.2 Non Peer-reviewed Conferences

[**D**] D. M. Rachman, N. Umeda, and A. Maki, "Warm-started online trajectory planner for ship's docking (berthing) under wind disturbance", in 日本船舶海洋工学会講演会論文集 *Conference proceedings, the Japan Society of Naval Architects and Ocean Engineers*, vol. 33. 日本船舶海洋工学会, Nov. 2021, pp. 39-48, (presented online).

[**E**] D. M. Rachman, Y. Aoki, Y. Miyauchi, and A. Maki, "Automatic docking (berthing) by dynamic positioning system with VecTwin rudder", in 日本船舶海洋工学会講演会論文集 *Conference proceedings, the Japan Society of Naval Architects and Ocean Engineers*, vol. 34. 日本船舶海洋工学会, May 2022, pp. 35-39, (Hiroshima).

1.3 Thesis Outlines

Chapter I gives a very fundamental introduction to the motivation of the studies and to the definition of some general terms used in this thesis. Chapter II describes the mathematical model for maneuvering at a low speed of one of the subject ships: the EOShip. Chapter III explains the methodology and examples of a safe trajectory fast planner for automatic docking (simulated with the mathematical model of EOShip) that is part of **contribution I**. Chapter IV investigates the possibility of implementing CMA-ES in real time to solve different OCPs related to automatic docking (simulated with the mathematical model of EOShip); this is part of **contribution II**. Chapter V introduces a novel multipurpose low-speed positioning system that is suitable for automatic docking

(tested on several scale model experiments with VTShip); this is part of **contribution III**. Finally, the studies in this thesis are summarized in chapter VI along with some possible future work.

II

Dynamics of Maneuvering at a Low Speed

As mentioned in the footnote 3 of chapter I, the dynamics of a body is formed by its kinematics and kinetics. Kinematics is a set of equations that relate the body's position with its velocity without consideration of the forces that cause the body to move (see footnote 2). Meanwhile, kinetics is a set of equations that relate a mass's acceleration with the forces (see footnote 3). The dynamics is usually expressed as a set of first-order ODE (ordinary differential equation) that *evolves in time*, which is why it can also be referred to as a *dynamical system*. It is also referred to as a *mathematical model*. In this thesis, the terms *dynamics*, *dynamical model*, *mathematical model*, and *maneuvering model* are used interchangeably.

This chapter explains the dynamics of a conventional underactuated ship: EOShip. Here the term *conventional ship* refers to an underactuated ship that is equipped only with a rudder and a propeller. Since a docking operation must be conducted at a low speed, the dynamics at all possible combinations of surge velocity and the propeller's mode (forward or reverse) must be described. This chapter starts with the descriptions of the kinematics, followed by the kinetics along with the explanation of how to estimate the forces.

2.1 Kinematics

We shall introduce two coordinate systems: a ship-fixed coordinate system o$-xy$ with the midship as its origin and an earth-fixed coordinate system $\mathrm{o}_0{-}x_0y_0$ (see figure 2-1). The angle between the two coordinate systems is the yaw angle, denoted by ψ, and r denotes the yaw angular velocity (yaw rate). The surge and the sway velocities (relative to the earth-fixed coordinate system) at the midship are denoted by u and v_m, respectively. The resultant of these velocities, $U = \sqrt{u^2 + {v_\mathrm{m}}^2}$, is the advance speed. The angle between U and the surge velocity u is the drift angle, denoted by β, that can be computed as $\beta = \operatorname{atan2}(-v_\mathrm{m}/u)$.

Now, the kinematics at the midship, i.e., the velocities with respect to the earth-fixed coordinate system, can be transformed into the ship-fixed coordinate system by the following transformation:

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\eta} = \mathbf{R}(\psi)\,\mathbf{v}, \tag{2.1–1}$$

where $\boldsymbol{\eta}$ is the *pose* of the ship in the earth-fixed coordinate system defined as follows:

$$\boldsymbol{\eta} = \begin{bmatrix} x_0 & y_0 & \psi \end{bmatrix}^\top, \tag{2.1–2}$$

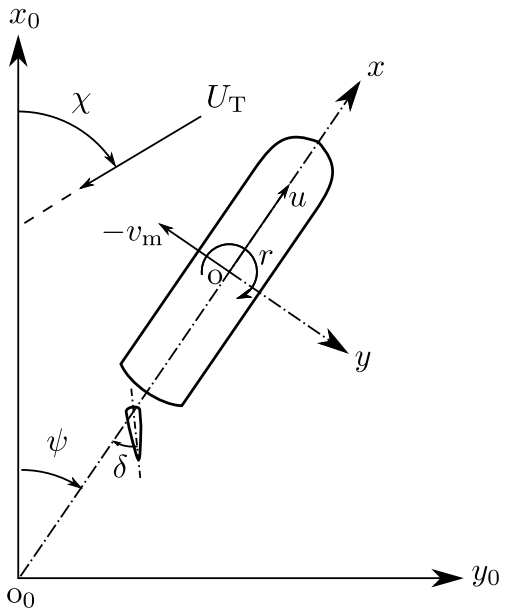


Figure 2-1. Coordinate systems.

and $\mathbf{v}$ is the vector of the ship's velocities at the midship relative to the earth-fixed coordinate system defined as follows:

$$\mathbf{v} = \begin{bmatrix} u & v_{\mathrm{m}} & r \end{bmatrix}^{\top}. \tag{2.1–3}$$

The matrix $\mathbf{R}(\psi)$ represents an orthogonal transformation $\left(\mathbf{R}^{-1} = \mathbf{R}^{\top}\right)$ known as the rotation matrix (rotation in the counterclockwise direction) as follows:

$$\mathbf{R}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.1–4}$$

2.2 Kinetics

The kinetics follows the standard equations of motion presented by the MMG [49]. These equations of motion are described in the ship-fixed coordinate system. Suppose a ship has a mass m and its center of gravity is located at x_{G} from the midship. It also has a moment of inertia about x_{G}, denoted by $I_{z\mathrm{G}}$. Then, the equations of motions about the midship can be written as follows:

$$(m + m_x)\,\dot{u} - (m + m_y)\,v_{\mathrm{m}} r - x_{\mathrm{G}} m r^2 = X \tag{2.2–1a}$$

$$(m + m_y)\,\dot{v}_{\mathrm{m}} + (m + m_x)\,ur + x_{\mathrm{G}} m \dot{r} = Y \tag{2.2–1b}$$

$$\left(I_{z\mathrm{G}} + {x_{\mathrm{G}}}^2 m + J_z\right)\dot{r} + x_{\mathrm{G}} m\,(\dot{v}_{\mathrm{m}} + ur) = N_{\mathrm{m}}, \tag{2.2–1c}$$

where m_x, m_y, and J_z denote the added masses and added moment of inertia in the respective axes. Note that the $(\dot{\ })$ denotes the derivative with respect to time t or $\mathrm{d}(\ )/\mathrm{d}t$.

The right-hand side of equation (2.2–1), i.e., X, Y, and N_{m} denote the total forces in the respective axes. These total forces can be expanded as follows:

$$X = X_{\mathrm{H}} + X_{\mathrm{P}} + X_{\mathrm{R}} + X_{\mathrm{W}} \tag{2.2–2a}$$

$$Y = Y_{\mathrm{H}} + Y_{\mathrm{P}} + Y_{\mathrm{R}} + Y_{\mathrm{W}} \tag{2.2–2b}$$

$$N_{\mathrm{m}} = N_{\mathrm{H}} + N_{\mathrm{P}} + N_{\mathrm{R}} + N_{\mathrm{W}}, \tag{2.2–2c}$$

where subscripts H, P, R, and W denote the forces and yaw moments due to the hull, the propeller, the rudder, and the wind, respectively.

Sometimes it is convenient to nondimensionalize some of the physical parameters. The nondimensionalized parameters are denoted by prime $'$ symbol. The lengths are nondimensionalized by the ship's length L, the surge and the sway velocities by the ship's advance speed U, and the yaw rate by L/U. The forces and the moments are nondimensionalized by $(1/2)\,\rho L d_{\mathrm{s}} U^2$ and $(1/2)\,\rho L^2 d_{\mathrm{s}} U^2$, respectively. Meanwhile, the mass and the moment of inertia are nondimensionalized by $(1/2)\,\rho L^2 d_{\mathrm{s}}$ and $(1/2)\,\rho L^4 d_{\mathrm{s}}$, respectively. Here ρ is the density of the water and d_{s} is the ship's draft.

2.3 Description of the Forces

This section describes the way to estimate each of the force components in equation (2.2–2) as follows: the estimation of the hull forces, the thrust from the propeller, and the steering forces from the rudder at all four *quadrants*, i.e., the possible combinations of surge velocity u and propeller revolution number n, and the forces due to the wind.

2.3.1 Hull Forces at a Low Speed

At a low speed, the surge velocity u and the sway velocity v can be in the same order of magnitude. Small u results in a drift angle $\beta \approx \pm 90\,\mathrm{deg}$. When $u \approx 0$, the nondimensionalized yaw rate r' blows up to $\pm\infty$. This makes the standard descriptions to estimate the hull forces shown in [49] not suitable for low-speed maneuvering like a docking operation. Instead, these hull forces are estimated following the descriptions shown in the unified model for ocean and harbor maneuvering [32] as follows:

$$X_{\mathrm{H}} = \frac{1}{2}\rho L d_{\mathrm{s}} \left\{ \left[X'_{0(\mathrm{F})} + \left(X'_{0(\mathrm{A})} - X'_{0(\mathrm{F})} \right) (|\beta|/\pi) \right] uU + X'_{vr} L v_{\mathrm{m}} r \right\} \tag{2.3–1a}$$

$$\begin{aligned} Y_{\mathrm{H}} = \frac{1}{2}\rho L d_{\mathrm{s}} \Bigg[& Y'_v v_{\mathrm{m}} |u| + Y'_r L r u \\ & - \frac{C_{\mathrm{D}}}{L} \int_{-L/2}^{L/2} |v_{\mathrm{m}} + C_{rY} r x| \left(v_{\mathrm{m}} + C_{rY} r x \right) \mathrm{d}x \Bigg] \end{aligned} \tag{2.3–1b}$$

$$\begin{aligned} N_{\mathrm{H}} = \frac{1}{2}\rho L^2 d_{\mathrm{s}} \Bigg[& N'_v v_{\mathrm{m}} u + N'_r L r |u| \\ & - \frac{C_{\mathrm{D}}}{L^2} \int_{-L/2}^{L/2} |v_{\mathrm{m}} + C_{rN} r x| \left(v_{\mathrm{m}} + C_{rN} r x \right) x \, \mathrm{d}x \Bigg], \end{aligned} \tag{2.3–1c}$$

where $X'_{0(\mathrm{F})}$, $X'_{0(\mathrm{A})}$, C_{D}, C_{rY}, and C_{rN} denote the ahead resistance coefficient, the astern resistance coefficient, the cross-flow drag coefficient, the correction factor for the sway

force, and the correction factor for the yaw moment, respectively. X'_{vr} is the hydrodynamic derivative with respect to the sway velocity and yaw rate. Y'_v and N'_v are the linear hydrodynamic derivatives with respect to the sway velocity, while Y'_r and N'_r are the linear hydrodynamic derivatives with respect to the yaw rate.

2.3.2 Propeller Forces

The forces and moment due to the propeller are estimated based on the surge velocity u and the sign of the propeller revolution number n (the mode), i.e., based on the *quadrant* (see figure 2-2).

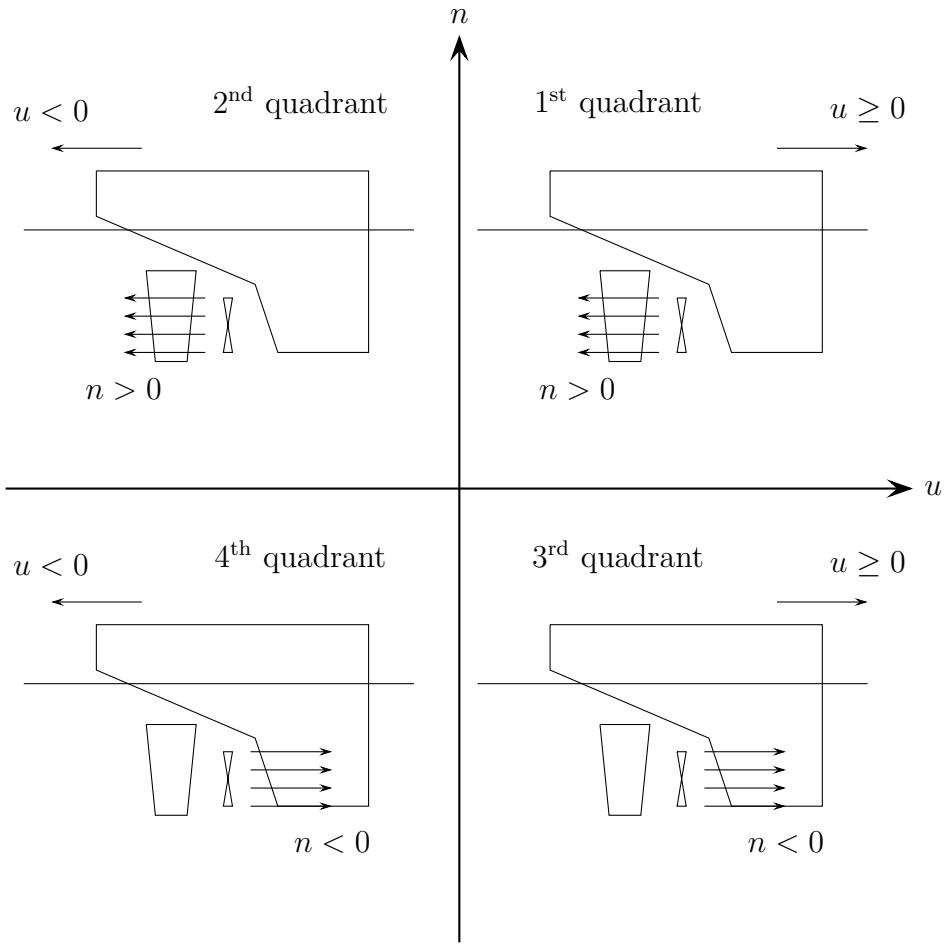


Figure 2-2. The definition of quadrants.

The 1st quadrant ($u \geq 0, n > 0$) describes the ship in the usual forward maneuvers. In this condition, the forces and moments induced by the propeller, i.e., X_{P}, Y_{P}, and N_{P} can be computed as follows [49]:

$$X_{\mathrm{P}} = (1 - t_{\mathrm{P}})\, \rho n^2 D_{\mathrm{P}}^4 K_{\mathrm{T}} \tag{2.3–2a}$$

$$Y_{\mathrm{P}} = 0 \quad \text{at } u \geq 0 \tag{2.3–2b}$$

$$N_{\mathrm{P}} = 0 \quad \text{at } u \geq 0, \tag{2.3–2c}$$

where D_P is the diameter of the propeller and t_P is the thrust deduction factor. The thrust coefficient K_T can be approximated with a polynomial function as follows:

$$K_\mathrm{T} = k_0 + k_1 J_\mathrm{P} + k_2 J_\mathrm{P}^2, \tag{2.3–3}$$

where k_0, k_1, and k_2 are the polynomial coefficients. The advance ratio of the propeller J_P is expressed as follows:

$$J_\mathrm{P} = (1 - w_\mathrm{P}) \frac{u}{nD_\mathrm{P}}, \tag{2.3–4}$$

where w_P is the wake fraction coefficient at the position of the propeller x_P that can be estimated as follows [50]:

$$1 - w_\mathrm{P} = 1 - w_{\mathrm{P},0} + \tau_\mathrm{P} |v'_\mathrm{m} + x'_\mathrm{P} r'| + C'_\mathrm{P} \left(v'_\mathrm{m} + x'_\mathrm{P} r'\right)^2. \tag{2.3–5}$$

In the above equation, C'_P and τ_P are coefficients that were experimentally determined. Meanwhile, $w_{\mathrm{P},0}$ is the corresponding w_P when $v_\mathrm{m} = 0$ and $r = 0$ (straight-ahead).

The 2$^\mathrm{nd}$ quadrant ($u < 0, n > 0$) describes the ship in a crash-forward situation. In this condition, the forces induced by the propeller in the surge direction can still be estimated with equation (2.3–2a). At $u < 0$, it is assumed that there is no decrease in the velocity within the wake field of the propeller, i.e., $w_\mathrm{P} = 0$. With this assumption, the force in the sway direction Y_P and the yaw moment N_P induced by the propeller can be estimated following the expressions presented in [50] as follows:

$$Y_\mathrm{P} = \frac{1}{2} \rho L d_\mathrm{s} (nP)^2 \left(A_6 J_\mathrm{s}^2 + A_7 J_\mathrm{s} + A_8\right) \tag{2.3–6}$$

$$N_\mathrm{P} = \frac{1}{2} \rho L^2 d_\mathrm{s} (nP)^2 \left(B_6 J_\mathrm{s}^2 + B_7 J_\mathrm{s} + B_8\right), \tag{2.3–7}$$

where J_s is the advance coefficient of the propeller when $w_\mathrm{P} = 0$ and P is the pitch of the propeller. The polynomial coefficients, A_6, A_7, and A_8 are used to estimate the force in the sway direction induced by the propeller, while B_6, B_7, and B_8 are used to estimate the yaw moment induced by the propeller.

In the 3$^\mathrm{rd}$ and the 4$^\mathrm{th}$ quadrant, when the propeller is in reverse mode ($n < 0$), the forces and moment can be estimated with the expressions shown in [50] as follows:

$$X_\mathrm{P} = \rho n^2 D_\mathrm{P}^4 \begin{cases} C_6 + C_7 J_\mathrm{s} & \text{if } J_\mathrm{s} \geq C_{10}, \\ C_3 & \text{if } J_\mathrm{s} < C_{10} \end{cases} \tag{2.3–8a}$$

$$Y_\mathrm{P} = \frac{1}{2} \rho L d_\mathrm{s} (nD_\mathrm{P})^2 \begin{cases} A_1 + A_2 J_\mathrm{s} & \text{if } -0.35 \leq J_\mathrm{s} \leq -0.06, \\ A_3 + A_4 J_\mathrm{s} & \text{if } J_\mathrm{s} < -0.35, \\ A_5 & \text{if } -0.06 < J_\mathrm{s} \end{cases} \tag{2.3–8b}$$

$$N_\mathrm{P} = \frac{1}{2} \rho L^2 d_\mathrm{s} (nD_\mathrm{P})^2 \begin{cases} B_1 + B_2 J_\mathrm{s} & \text{if } -0.35 \leq J_\mathrm{s} \leq -0.06, \\ B_3 + B_4 J_\mathrm{s} & \text{if } J_\mathrm{s} < -0.35, \\ B_5 & \text{if } -0.06 < J_\mathrm{s} \end{cases}, \tag{2.3–8c}$$

where A_1 to A_5, B_1 to B_5, C_{10}, C_3, C_6, and C_7 are empirical coefficients to estimate these forces. More details are discussed in [51].

2.3.3 Rudder Forces

The forces and the moment induced by the rudder are very dependent on the inflow speed at the rudder U_{R}, which is different in every quadrant. This speed is the resultant of the inflow velocity at the rudder in the surge direction u_{R} and the inflow velocity at the rudder in the sway direction v_{R}, or $U_{\mathrm{R}} = \sqrt{{u_{\mathrm{R}}}^2 + {v_{\mathrm{R}}}^2}$. It affects the rudder normal force F_{N} that depends on the effective area of the rudder A_{R} and α_{R} (the effective inflow angle; the angle between U_{R} and the centerline of the rudder). The rudder normal force can be expressed as follows:

$$F_{\mathrm{N}} = \frac{1}{2}\rho A_{\mathrm{R}} U_{\mathrm{R}}^2 f_\alpha \sin\alpha_{\mathrm{R}}, \tag{2.3–9}$$

where f_α is the rudder lift gradient coefficient that is a function of the rudder aspect ratio λ as follows [52]:

$$f_\alpha = \frac{6.13\lambda}{(2.25 + \lambda)}. \tag{2.3–10}$$

Moreover, α_{R} can be estimated from the rudder angle δ as follows:

$$\alpha_{\mathrm{R}} = \delta - \mathrm{atan2}\left(v_{\mathrm{R}}/u_{\mathrm{R}}\right). \tag{2.3–11}$$

Regardless of the quadrant, the lateral inflow velocity v_{R} depends on the effective longitudinal position of the rudder from the midship, denoted by l_{R}, as follows [50]:

$$v_{\mathrm{R}} = \begin{cases} -\gamma_{\mathrm{P}}\left(v_{\mathrm{m}} + l_{\mathrm{R}} r\right) & \text{if } v_{\mathrm{m}} + x_{\mathrm{R}} r \geq 0 \\ -\gamma_{\mathrm{N}}\left(v_{\mathrm{m}} + l_{\mathrm{R}} r\right) & \text{if } v_{\mathrm{m}} + x_{\mathrm{R}} r < 0, \end{cases} \tag{2.3–12}$$

where x_{R} is the longitudinal position of the rudder from the midship, while γ_{N} and γ_{P} are the flow straightening coefficients.

In the 1st and 2nd quadrant when $n \geq 0$, the longitudinal inflow velocity u_{R} can be estimated as a function of the ratio between the wake fraction at the rudder position and that at the propeller position, denoted by ε, as follows [50]:

$$u_{\mathrm{R}} = \varepsilon \sqrt{\eta_{\mathrm{R}} \left\{ u_{\mathrm{P}} + \frac{k_x}{\varepsilon}\left(\sqrt{u_{\mathrm{P}}^2 + \frac{8K_{\mathrm{T}}\left(nD_{\mathrm{P}}\right)^2}{\pi}} - u_{\mathrm{P}}\right)\right\}^2 + \left(1 - \eta_{\mathrm{R}}\right) u_{\mathrm{P}}^2}, \tag{2.3–13}$$

where $u_{\mathrm{P}} = (1 - w_{\mathrm{P}})\, u$, η_{R} is the aspect ratio of the rudder: the ratio between D_{P} and the height of the rudder H_{R}, and k_x is an empirical coefficient.

In the 3rd quadrant, the estimation of u_{R} is quite intricate. It depends on some intermediate calculations as presented in [53] as follows:

$$u_{\mathrm{R}} = \mathrm{sgn}\left(u_{\mathrm{Rsq}}\right)\sqrt{|u_{\mathrm{Rsq}}|}, \tag{2.3–14}$$

where,

$$u_{\mathrm{Rsq}} = \mathrm{sgn}\left(u_{\mathrm{RPr1}}\right)\eta_{\mathrm{R}} u_{\mathrm{RPr1}}^2 + \left(1 - \eta_{\mathrm{R}}\right)\mathrm{sgn}\left(u_{\mathrm{RPr2}}\right) u_{\mathrm{RPr2}}^2 + C_{\mathrm{Pr}} u \tag{2.3–15}$$

$$u_{\mathrm{RPr1}} = u\varepsilon\left(1 - w_{\mathrm{P}}\right) + nD_{\mathrm{P}} k_{xr}\sqrt{8|K_{\mathrm{T}}|/\pi} \tag{2.3–16}$$

$$u_{\mathrm{RPr2}} = u\varepsilon\left(1 - w_{\mathrm{P}}\right). \tag{2.3–17}$$

In the above equations, k_{xr} is an empirical coefficient and C_{Pr} is a correction factor (for more details see [50, 53]). In the 4^{th} quadrant, at a low surge velocity, it is assumed that $u_{\mathrm{R}} = u$, i.e., the longitudinal inflow velocity is equal to the surge velocity [54].

With the information to calculate the rudder normal force F_{N} available, the forces and the moment induced by the rudder can be calculated as follows:

$$X_{\mathrm{R}} = -\left(1 - t_{\mathrm{R}}\right) F_{\mathrm{N}} \sin\delta \tag{2.3–18a}$$

$$Y_{\mathrm{R}} = -\left(1 - a_{\mathrm{H}}\right) F_{\mathrm{N}} \cos\delta \tag{2.3–18b}$$

$$N_{\mathrm{R}} = -\left(x_{\mathrm{R}} + a_{\mathrm{H}} x_{\mathrm{Hr}}\right) F_{\mathrm{N}} \cos\delta, \tag{2.3–18c}$$

where X_{R}, Y_{R}, and N_{R} are the force in the surge direction, the force in the sway direction, and the yaw moment, respectively, induced by the rudder. In the above equation, t_{R} is the steering resistance deduction factor and x_{R} is the longitudinal position of the rudder from the midship. Moreover, a_{H} is the factor for additional sway force due to the steering that acts at a location x_{Hr} from the midship [49].

2.3.4 Wind Forces

Suppose an apparent wind *blows to* at a speed of U_{T} and at an angle χ relative to x_0-axis of the earth-fixed coordinate system. Since the ship moves, the true wind is then transformed into an apparent wind, where the apparent wind speed is denoted by U_{A} and its angle relative to the surge velocity u is denoted by γ_{A}. The forces and the moment due to the apparent wind can be estimated with the following expressions:

$$X_{\mathrm{W}} = \frac{1}{2}\rho_{\mathrm{a}} U_{\mathrm{A}}^2 C_X A_{\mathrm{T}} \tag{2.3–19a}$$

$$Y_{\mathrm{W}} = \frac{1}{2}\rho_{\mathrm{a}} U_{\mathrm{A}}^2 C_Y A_{\mathrm{L}} \tag{2.3–19b}$$

$$N_{\mathrm{W}} = \frac{1}{2}\rho_{\mathrm{a}} U_{\mathrm{A}}^2 C_N L A_{\mathrm{L}}, \tag{2.3–19c}$$

where ρ_{a}, A_{T}, and A_{L} are the density of the air, the transversal (front) windage area, and the lateral (side) windage area, respectively. Meanwhile, C_X, C_Y, and C_N are the coefficients to calculate the forces and moment in the respective axes. These coefficients are calculated based on the regression formula presented in [55]. The procedure to transform the true wind parameters to the apparent wind parameters is given in detail in [56].

2.4 Dynamics

We have established the expression for the kinematics (section 2.1) and the kinetics (section 2.2), along with the descriptions of the forces (section 2.3). To form the *dynamics*, one can rearrange the kinetics from equation (2.2–1):

$$\begin{aligned}
\left(m + m_x\right)\dot{u} - \left(m + m_y\right) v_{\mathrm{m}} r - x_{\mathrm{G}} m r^2 &= X \\
\left(m + m_y\right)\dot{v}_{\mathrm{m}} + \left(m + m_x\right) u r + x_{\mathrm{G}} m \dot{r} &= Y \\
\left(I_{z\mathrm{G}} + {x_{\mathrm{G}}}^2 m + J_z\right)\dot{r} + x_{\mathrm{G}} m \left(\dot{v}_{\mathrm{m}} + u r\right) &= N_{\mathrm{m}},
\end{aligned}$$

into the following expressions:

$$\dot{u} = \frac{\mathrm{d}u}{\mathrm{d}t} = \frac{X + M_y v_\mathrm{m} r + x_\mathrm{G} m r}{M_x} \tag{2.4–2a}$$

$$\dot{v}_\mathrm{m} = \frac{\mathrm{d}v_\mathrm{m}}{\mathrm{d}t} = \frac{(Y - M_x u r)\, I_{z\mathrm{m}} - (N_\mathrm{m} - x_\mathrm{G} m u r)\, x_\mathrm{G} m}{M_y I_{z\mathrm{m}} - (x_\mathrm{G} m)^2} \tag{2.4–2b}$$

$$\dot{r} = \frac{\mathrm{d}r}{\mathrm{d}t} = \frac{(Y - M_x u r)\, x_\mathrm{G} m - (N_\mathrm{m} - x_\mathrm{G} m u r)\, M_y}{(x_\mathrm{G} m)^2 - M_y I_{z\mathrm{m}}}, \tag{2.4–2c}$$

where $M_x = m + m_x$, $M_y = m + m_y$, and $I_{z\mathrm{m}} = I_{z\mathrm{G}} + x_\mathrm{G}^2 + J_z$ are used for simplicity.

Combining the kinematics from equation (2.1–1) and the rearranged kinetics from equation (2.4–2), one can form a system of ODEs—a dynamical system— that represents the dynamics at the midship and describes how it evolves in time. This system can be expressed as follows:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_0 \\ y_0 \\ \psi \\ u \\ v_\mathrm{m} \\ r \end{bmatrix} = \begin{bmatrix} \mathbf{R}\,(\psi)\,\mathbf{v} \\ \\ \mathbf{k}\,(\mathbf{v}, \psi, n, \delta, U_\mathrm{T}, \chi) \end{bmatrix}, \tag{2.4–3}$$

where $\mathbf{k}$ is the right-hand side of equation (2.4–2). One can rearrange and rewrite equation (2.4–3) into a simplified notation for the dynamics as follows[5]:

$$\dot{\mathbf{x}}\,(t) = \mathbf{f}\,(\mathbf{x}\,(t)\,, \mathbf{u}\,(t)\,, U_\mathrm{T}, \chi)\,, \tag{2.4–4}$$

where,

$$\mathbf{x}\,(t) \triangleq \begin{bmatrix} x_0 & u & y_0 & v_\mathrm{m} & \psi & r \end{bmatrix}^\top \tag{2.4–5}$$

$$\mathbf{u}\,(t) \triangleq \begin{bmatrix} \delta & n \end{bmatrix}^\top. \tag{2.4–6}$$

Here, $\mathbf{x} \in \mathbb{R}^6$ is termed *state vector* or simply *states*, where its elements are called the *state variables*. Moreover, $\mathbf{u} \in \mathbb{R}^2$ is termed *control command vector* or simply *control*, where its elements are called the *control commands*.

By integrating the dynamics (2.4–4) with respect to time, one can obtain the states of the ship that evolved from an initial state vector, after some time, given a control command vector. As one can see in section 2.3, the expressions of the forces are very intricate and highly nonlinear. Thus, integration is usually performed numerically through explicit integration schemes like Runge-Kutta or Euler's method.

[5]Note that the function $\mathbf{f}$ does not explicitly depend on the time t. Thus, the dynamics in equation (2.4–4) is a system of autonomous ODEs, i.e., an autonomous dynamical system.

大阪大学 OSAKA UNIVERSITY

Appendices

II.A Principal particulars of the subject ships: the EOShip and the VTShip

For brevity, only the principle particulars of the EOShip are presented in this thesis. The parameters/coefficients that describe the forces are available in [10, 50]. The EOShip is the subject ship discussed in chapters III and IV.

Table 2-1. Particulars of the EOShip: a 1/108 scale model.

Parameters	Measurement
Length between perpendiculars: L (m)	3.000
Breadth: B (m)	0.489
Draft: d_{s} (m)	0.201
Longitudinal center of gravity from midship: x_{G} (m)	0.095
Mass: m (kg)	245.091

Moreover, although we do not have the mathematical model of the VTShip (because we do not present any simulations with it), its principal particulars are shown in this appendix. It will be relevant in chapter V.

Table 2-2. Particulars of the VTShip: a 1:50 scale model.

Parameters	Measurement
Length between perpendiculars: L (m)	3.000
Breadth: B (m)	0.484
Draft: d_{s} (m)	0.172
Longitudinal center of gravity from midship: x_{G} (m)	0.094
Longitudinal position of rudders from x_{G}: x_{fR} (m)	-1.657
Longitudinal position of bow thruster from x_{G}: x_{B} (m)	1.263

III

Efficient Planning of a Safe Docking Trajectory

3.1 Introduction

Commonly, a trajectory is planned by solving a case of OCP (optimal control problem) known as *trajectory optimization*. Throughout this chapter, the terms OCP and trajectory optimization will be used interchangeably. In practice, one directly solves a continuous OCP by discretizing and transcribing it into a finite-dimensional nonlinear optimization problem (NLP) via *direct* methods such as *shooting* method and *collocation* method [28, 29]. Upon solving the NLP, one obtains the optimal states and the optimal control command vector at the discretization points.

Planning a trajectory for a docking operation is rather different from when planning one for a general operation. General operations are those that were executed in the open sea where the ship moves at a service speed. The global nature of a general operation makes it possible to construct a trajectory through path smoothing [23,24]. However, the low speed and the high safety requirement of a docking operation result in a complicated dynamics (see chapter II) and subsequently the trajectory optimization itself.

The development of the trajectory planner for docking operations has seen various efforts to incorporate and balance the following things: low computational cost [18, 21], applicability to conventional underactuated ships [8, 10], and feasibility with respect to the dynamics of the environment [8, 12] and the geometric constraints [9, 14, 19, 21]. As explained in section 1.1.1, this is relevant because in practice a shipmaster has to replan a new trajectory when it is deemed not feasible (e.g., change in the wind condition). Even an experienced shipmaster may require some time to plan a new trajectory. In an autonomous ship, the trajectory planner must be able to act decisively like a shipmaster. For this reason, this thesis proposes a design of a relatively fast planner that can replan a safe and feasible trajectory, and applicable to conventional underactuated ships.

The trajectory planner for conventional underactuated ships introduced in [10, 14] can produce an almost-globally optimal state trajectory that is also safe. It requires a heavy computational cost and long computational time, which makes it unlikely to be implemented in real time. The one presented in [8] is relatively fast and can be implemented for a conventional ship, but there were no geometric constraints that can guarantee a safe trajectory. A fast trajectory planner that guarantees a safe trajectory is proposed by [21], but applicable only to an overly/fully actuated ship with a comparably less intricate mathematical model. Still related to the discussion, [12] introduces a fast trajectory planning for underactuated ship. However, the ship is regarded just as a particle

instead of multiple points (that resemble a body), and the allision[6] with the harbor's wall is avoided by expanding the harbor's boundaries.

By exploiting the advantages of each of the previously mentioned works, this thesis introduces a *semionline* trajectory planner—from here on abbreviated as SO-TP—that is capable of planning an allision-free trajectory for an conventional underactuated ships under wind disturbance in a static environment. The term *semionline* implies the characteristic of the computational time that is relatively fast in practice, but not strictly real time (online). This term also implies the ability of the trajectory planner to incorporate the dynamics of the wind at the instant the planner is executed. This planner is at the intermediate level between a high-level trajectory planner that takes into account static conditions and a middle-level trajectory planner that modifies the trajectory by taking into account the instantaneous wind dynamics and the ship's states.

3.1.1 Some Key Points

The most important aspect introduced in the SO-TP is that it uses an almost-globally optimal and feasible solution generated beforehand by the trajectory planner presented in [10] as a *warm start*. A warm start is an optimal solution of another simplified problem (yet identical) that is used as the initial guess when solving the actual optimization problem [57]. Every gradient-based solver requires an initial guess: initial value where the gradient descent starts. This warm start is expected to give a good starting point for the solver inside the SO-TP so that the gradient descent to the local optimum[7] can be completed faster than that without a warm start (*cold start*, any arbitrary initial guesses).

When evaluating the performance of the warm-started SO-TP and the overall scheme, the following points will be evaluated (EvP is an abbreviation for evaluation point):

EvP. 1: The applicability of just one warm start to speed up the SO-TP for various cases with different initial states and wind parameters. The resulting trajectories with and without a warm start (where a linear initial guess is used) from a total of 14 different cases are compared based on the computational time.

EvP. 2: The comparison of the resulting trajectories with two different warm starts: (1) generated via the trajectory planner from [10], and (2) generated by the SO-TP itself but with a linear initial guess. This is to demonstrate how significant a good initial guess is to warm-start the optimization.

This article can serve as a continuation and implementation of the works presented in [10, 14] where computational time was the biggest issue that significantly limits their practical use. To address this issue, the first major development implemented in the SO-TP unique to the previous works on automatic docking OCP-based trajectory planners for conventional ships is the use of a warm start to speed up the computational time.

The next major development is that the SO-TP implements an allision constraints generator via a *point-in-polygon* method [58]. This method is applicable to either convex

[6]The term allision refers to any physical contact with static objects like the harbor's walls. On the other hand, collision refers to any physical contact with moving objects like other ships.

[7]In nonlinear and nonconvex optimization where the search space is nonsmooth with many local optima (rugged), finding the global optimum is almost impractical for a gradient-based solver. The line search may not be able to direct the search out from a local optima.

or nonconvex regions The generated constraints are strictly and directly enforced in the NLP. Up to the limit of the author's knowledge, this direct implementation of the method is the first in the topic of trajectory optimization for marine use. This method can also be an alternative to the allision-collision constraints generators introduced in [9, 12, 21].

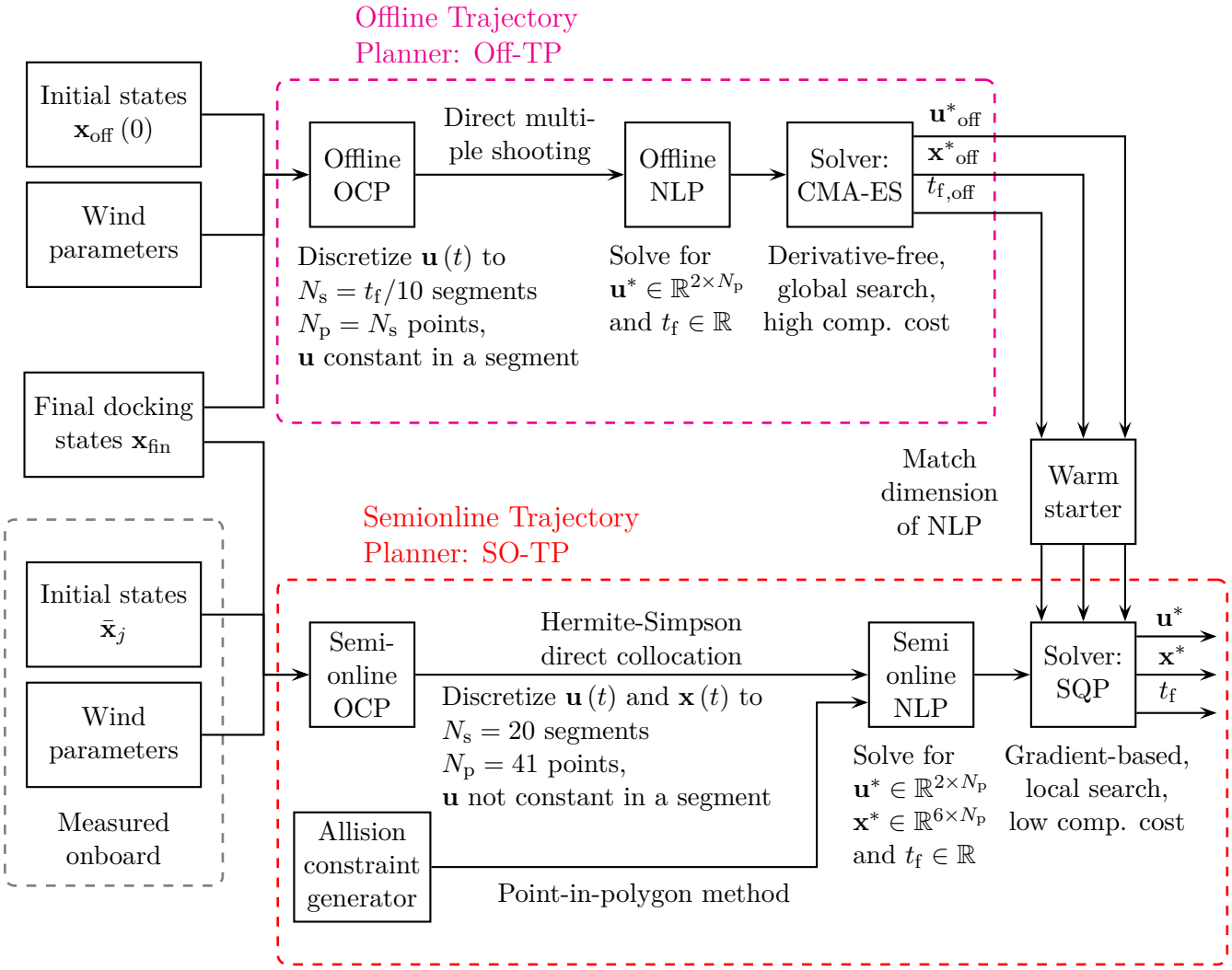


Figure 3-1. Overviews of the proposed scheme to efficiently plan a trajectory.

3.1.2 Proposed Scheme

The overview of the proposed scheme is shown in figure 3-1 above and summarized in the following steps:

1. First, a database of optimal controls and optimal state trajectories can be generated via the high-level trajectory planner in [10]. This planner is called *offline trajectory planner* (abbreviated as Off-TP). Moreover, an optimal solution in this database is termed *offline trajectory* or *offline solution*.
2. The offline solutions can be generated for several initial states and wind parameters. As a demonstration, this article shows how just one offline solution (without

considering the wind) can be used in many cases (evaluation point 1).
3. Then, the offline solution is used to warm-start the SO-TP (semionline trajectory planner).
4. The OCP in the SO-TP is discretized and transcribed into a finite-dimensional NLP with direct Hermite-Simpson collocation method. The form of the objective function is identical to that in the Off-TP.
5. With a *point-in-polygon* method [58], as the name suggests, one can check whether a point is inside or outside a polygon that defines an allision-free region. This results in several nonlinear and implicit geometric constraints to the NLP.
6. Finally, this NLP is solved by MATLAB `fmincon` package with SQP (sequential quadratic programming) [59] as its main solver. Such an optimization where a good initial guess (warm start) is provided is said to be *warm-started* [57].

The SO-TP and the above scheme will be tested and simulated with the mathematical model of a conventional underactuated ship: EOShip. The principal particulars of the ship are summarized in table 2-1 as an appendix. EOShip is a ship that is equipped with an electric motor that allows rapid switches of the propeller's mode (forward/reverse). It is also equipped with a rudder that is able to rotate at a rate of 70 deg/s.

3.2 Offline Trajectory Planner: Off-TP

This section explains briefly about the high-level trajectory planning introduced in [10]. The Off-TP is an OCP-based trajectory planning that has a main role of providing a warm start, i.e., an almost-globally optimal solution, as the initial guess to the middle-level SO-TP so that it can converge faster to the local optimum. Note that this warm start does not necessarily need to be the solution to the exact same optimization problem.

In the Off-TP, the OCP is discretized and transcribed into a finite-dimensional NLP by a modified version of multiple shooting method. The optimal control is discretized into several discretization points where at each point, the control command vector becomes the optimization variable. Thus, the goal of the constructed NLP is to find the optimal control command vector, denoted by $\mathbf{u}^*_{\text{off}}$, at all discretization points that drive the ship from the initial states $\mathbf{x}(0)$ to the desired final docking states $\mathbf{x}_{\text{fin}}$. Within a segment, $\mathbf{u}^*_{\text{off}}$ is assumed constant, which together with all the segments, form the optimal control $\mathbf{u}^*_{\text{off}}(t)$. This optimal control gives the optimal state trajectory $\mathbf{x}^*_{\text{off}}(t)$ that minimizes a modified terminal objective function: minimize the deviation of the states at the final time $t_{\text{f,off}}$ from $\mathbf{x}_{\text{fin}}$. These offline solution from the Off-TP: $\mathbf{u}^*_{\text{off}}$, $\mathbf{x}^*_{\text{off}}$, and $t_{\text{f,off}}$ are used to warm-start the optimization in the SO-TP (see figure 3-1). Note that the terms with an asterisk (*) denote the corresponding optimal values.

The offline NLP is solved by a nonconvex and derivative-free global optimization algorithm known as CMA-ES (covariance matrix adaptation evolution strategy) [60–62]. The implementation of this algorithm can be computationally demanding (a heuristic-stochastic algorithm, coupled with explicit integration of the dynamics and multiple shooting method). It performed very well in approximating the global optimum in the search space at the cost of computational time. Therefore, the offline solution from the Off-TP can be considered almost globally optimal. More detailed explanations about the Off-TP are given in [10].

大阪大学 OSAKA UNIVERSITY

3.3 Semionline Trajectory Planner: SO-TP

The SO-TP comes from the idea that in practice sometimes a shipmaster needs to replan a new trajectory for a docking operation based on the current environmental situation, i.e., wind. The updated trajectory is necessary to be kept safe from allision and feasible with respect to the dynamics of the ship and the wind dynamics. In this section, the formulation of the OCP, the transcription method, and the constructed NLP will be explained. We shall use the terms *predicted states* and *predicted dynamics* to refer to the states $\mathbf{x}$ and the dynamics $\mathbf{f}$, respectively, inside the SO-TP. We also use the term *predicted control* to refer to the control command vector $\mathbf{u}$ inside the SO-TP.

Let $\bar{t}$ denote the actual time outside the SO-TP where $\bar{t}_j$ denotes the j-th sampling of $\bar{t}$ and let $\bar{\mathbf{x}}_j$ be the actual states of the ship at $\bar{t}_j$. Moreover, let t be the normalized prediction time inside the SO-TP such that $t = 0$ is equal to $\bar{t}_j$. With this notation, the continuous OCP (before the transcription into NLP) of the semionline trajectory optimization problem at every $\bar{t}_j$ can be formulated as follows:

$$
\begin{aligned}
&\text{determine} && \mathbf{u}^*(t) = \begin{bmatrix} \delta^*(t) & n^*(t) \end{bmatrix}^\top \\
&\text{that min.} && J_{\text{on}} = \sum_{i=1}^{6} \left(x_i(t_\text{f}) - x_{\text{fin},i}\right)^2 \times \int_0^{t_\text{f}} \sum_{i=1}^{6} \left(x_i(t) - x_{\text{fin},i}\right)^2 \mathrm{d}t, && \text{(3.3–1a)} \\
&\text{subject to} && \dot{\mathbf{x}}(t) = \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t), U_\text{T}\left(\bar{t}_j\right), \chi\left(\bar{t}_j\right)\right), && \text{(3.3–1b)} \\
& && \mathbf{x}^*(0) = \bar{\mathbf{x}}_j,\ \mathbf{x}^*(t_\text{f}) = \mathbf{x}_{\text{fin}}, && \text{(3.3–1c)} \\
& && -\delta_{\max} \le \delta(t) \le \delta_{\max}, && \text{(3.3–1d)} \\
& && -n_{\max} \le n(t) \le n_{\max}, && \text{(3.3–1e)} \\
& && t \in [0, t_\text{f}], \quad \text{and} \quad t_\text{f} \in (0, \infty).
\end{aligned}
$$

Note that the wind parameters in the simulations are assumed constant when planning the trajectory, i.e., $U_\text{T}(t) = U_\text{T}\left(\bar{t}_j\right)$ and $\chi(t) = \chi\left(\bar{t}_j\right)$ at all t inside the SO-TP.

In equation (3.3–1), the objective function of the semionline OCP, denoted by J_{on}, is a kind of modified terminal problem. It will force the optimization to minimize the deviation of the predicted final states $\mathbf{x}(t_\text{f})$ from the desired final docking states $\mathbf{x}_{\text{fin}}$. The first summation term in equation (3.3–1a) describes the deviation at *terminal* condition: the deviation at the final time t_f. Meanwhile, the integral term captures the accumulated deviation of the states $\mathbf{x}(t)$ from $\mathbf{x}_{\text{fin}}$ in the whole prediction time. This objective function is identical to that in the Off-TP. This is because ideally the formulation of both the Off-TP and the SO-TP should be kept as identical as possible to exploit the warm start.

There are several constraints imposed on the OCP. The first is the dynamics $\mathbf{f}$ of the ship shown in equation (3.3–1b). The phrase *feasible with respect to the dynamics* means that this equation must be satisfied at all times. The next constraint is related to the initial states and the final states[8] shown in equation (3.3–1c). The constraint basically says that the actual states of the ship $\bar{\mathbf{x}}_j$ is also the optimal states at $t = 0$, or $\mathbf{x}^*(0)$. The next equation forces the optimization such that the optimal states at $t = t_\text{f}$, denoted by $\mathbf{x}^*(t_\text{f})$, is equal to the desired final docking states $\mathbf{x}_{\text{fin}}$.

[8]This is known as a BVP (boundary value problem). Most OCPs are BVP.

The next two constraints are imposed on the actuators/control commands. It is desirable to plan a reference trajectory that is "easy" to be tracked, for example, with less control effort. Such an idea is intended to compensate for the unmodeled dynamics and uncertainties that may affect the performance of the feedback controller. This can be implemented in two ways: (1) by adding a penalty due to the control in the objective function or (2) by explicitly limiting the allowable control command vector to some values $\delta_{\max}$ and $n_{\max}$ that are less than the actual capacity. The latter is much preferred since it is binding in its *box constraint* form. These are expressed in equations (3.3–1d) and (3.3–1e). The values of these limiting values are shown in table 3-1.

Table 3-1. Limiting/saturation values for the control command vector when generating the trajectory.

Control command	Limiting values	Actual capacity
δ (deg)	$\delta_{\max} = 25$	35
n (rps)	$n_{\max} = 15$	20

3.3.1 Hermite-Simpson Collocation

The continuous OCP formulated in equation (3.3–1) is solved *directly* by discretizing and transcribing it into a finite-dimensional nonlinear optimization problem (NLP). The term *direct* refers to the way the OCP is solved as an NLP instead of having to determine the necessary/sufficient optimality conditions of optimal control analytically via Pontryagin's minimum principle and Hamilton-Jacobi-Bellman (for a continuous problem) or dynamic programming (for a discrete problem) [28, 30]. The procedure of discretizing and transcribing an OCP into an NLP is called a transcription process. One of the transcription methods is the collocation method. In mathematics, the collocation method is a method to numerically obtain the solution of ODEs by approximating it as a polynomial and choosing a finite number of points in the domain (one of these points is called a *collocation point*). The approximating polynomial is called an *interpolation polynomial* or an *interpolant*. At a collocation point, the derivative of the interpolation polynomial should be equal to the ODE that one wants to solve.

3.3.1.1 Hermite Cubic Interpolation

The collocation starts with the discretization of the time histories of the dynamics, the states (trajectory), and the control, into $N_{\mathrm{s}} \in \mathbb{N}$ equal-spaced segments. Each of the segments is bounded by two *knot* points, each at the start and at the end of the segment. The length of each segment is $h_{\mathrm{s}} = t_{\mathrm{f}}/N_{\mathrm{s}}$. Let t_k be the time at the k-th knot point so that $h_{\mathrm{s}} = t_{k+1} - t_k$. For brevity, let $\mathbf{x}_k \equiv \mathbf{x}(t_k)$, $\mathbf{f}_k \equiv \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, U_{\mathrm{T}}, \chi)$, and $\mathbf{u}_k \equiv \mathbf{u}(t_k)$ be the predicted states, the predicted dynamics, and the predicted control, respectively, at the k-th knot point. Suppose that we have the information about $\mathbf{x}_k$ and $\mathbf{u}_k$ at the knot points through optimization[9], i.e., they are the optimization variables.

[9]The dynamics $\mathbf{f}$ is a function of $\mathbf{x}$ and $\mathbf{u}$. It is not an optimization variable.

Now, let the solution of the dynamics, i.e., the states, be approximated by a c-degree polynomial function. This interpolant, denoted by $x_{\mathrm{H}}(t)$, has the following form [63]:

$$x_{\mathrm{H}}(t) = \sum_{i=0}^{c} a_i \left(\frac{t - t_k}{h_{\mathrm{s}}}\right)^i, \quad t \text{ in } [t_k, t_{k+1}]. \tag{3.3–2}$$

A vector of interpolants is denoted by $\mathbf{x}_{\mathrm{H}}(t)$. Each segment has its own interpolant, i.e., piecewise polynomial function.

We know that the time-derivative of the states is equal by definition to the dynamics (see equations (2.4–4) and (3.3–1b)). This means the dynamics can be approximated as the time-derivative of equation (3.3–2), i.e., a one-order lower polynomial that has the following form:

$$\begin{aligned} \dot{x}_{\mathrm{H}}(t) &\triangleq f_{\mathrm{H}}(t) \\ &= \sum_{i=1}^{c} i \frac{a_i}{h_{\mathrm{s}}} \left(\frac{t - t_k}{h_{\mathrm{s}}}\right)^{i-1}, \quad t \text{ in } [t_k, t_{k+1}], \end{aligned} \tag{3.3–3}$$

where a vector of these functions is denoted as $\mathbf{f}_{\mathrm{H}} \triangleq \dot{\mathbf{x}}_{\mathrm{H}}$. Notice that the index i starts from 0 instead of 1 due to the loss of one degree.

Let b denote the number of the points in a segment. Let w be the maximum order of differentiation. We want to construct an interpolation polynomial where at all b points, it is differentiable up to the w-th order. Given any b and w, the required degree of the interpolation polynomial c satisfies the following relationship:

$$c = (w + 1)\, b - 1. \tag{3.3–4}$$

The next task is to find the coefficients of the interpolation polynomial $\mathbf{x}_{\mathrm{H}}$ (3.3–2), i.e., a_i, where $i = 0, 1, 2, \ldots, c$. The number of equations that are required to obtain these variables is at least equal to the number of these variables, or $c + 1$. Note that an interpolation polynomial that is equal to a function and its derivatives up to w-th order at b points is known as a w^{th}-order c^{th}-degree *Hermite interpolation polynomial*.

In a segment, there are two knot points, i.e., $b = 2$. It is required that the interpolant in equation (3.3–2) satisfies the predicted states such that $\mathbf{x}_{\mathrm{H}}(t) = \mathbf{x}(t)$ at all b points. It is also required that the interpolation polynomial is differentiable up to $w = 1$ (first order), that is, its derivative in equation (3.3–3) satisfies the predicted dynamics such that $\dot{\mathbf{x}}_{\mathrm{H}}(t) = \mathbf{f}(t)$. Thus, from equation (3.3–4), the required degree of the interpolation polynomial is $c = 3$; a cubic polynomial. This polynomial is a first-order third-degree Hermite interpolation polynomial.

A cubic polynomial requires four coefficients $(a_0, \ldots, a_3)$, i.e., four unknown variables. To solve this, at least four equations must be constructed. These equations are none other than the conditions that we want to impose at each of the knot points as follows:

$$\mathbf{x}_{\mathrm{H}}(t_k) = \mathbf{x}_k \tag{3.3–5a}$$

$$\mathbf{f}_{\mathrm{H}}(t_k) = \mathbf{f}_k \tag{3.3–5b}$$

$$\mathbf{x}_{\mathrm{H}}(t_{k+1}) = \mathbf{x}_{k+1} \tag{3.3–5c}$$

$$\mathbf{f}_{\mathrm{H}}(t_{k+1}) = \mathbf{f}_{k+1}. \tag{3.3–5d}$$

By analytically solving $\mathbf{x}_{\mathrm{H}}$ in equation (3.3–2) with the conditions shown in equation (3.3–5), one can obtain the interpolation polynomial coefficients as follows (see appendix III.A):

$$\begin{bmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & h_{\mathrm{s}} & 0 & 0 \\ -3 & -2h_{\mathrm{s}} & 3 & -h_{\mathrm{s}} \\ 2 & h_{\mathrm{s}} & -2 & h_{\mathrm{s}} \end{bmatrix} \begin{bmatrix} \mathbf{x}_k \\ \mathbf{f}_k \\ \mathbf{x}_{k+1} \\ \mathbf{f}_{k+1} \end{bmatrix}. \tag{3.3–6}$$

If the interpolation polynomial for the predicted states $\mathbf{x}_{\mathrm{H}}$ shown in equation (3.3–2) at each segment is combined (piecewise interpolation polynomials), they form a *cubic Hermite spline* that is continuous and once-differentiable everywhere in the whole time domain. Moreover, the interpolant for the predicted dynamics $\mathbf{f}_{\mathrm{H}}$ is also continuous but not differentiable in the whole time domain. In mathematics, one can express this continuity as follows:

$$\begin{aligned} \mathbf{x}_{\mathrm{H}} &\text{ is } C^1\left[0, t_{\mathrm{f}}\right], \text{ smooth: continuous and differentiable once,} \\ \dot{\mathbf{x}}_{\mathrm{H}} = \mathbf{f}_{\mathrm{H}} &\text{ is } C^0\left[0, t_{\mathrm{f}}\right], \text{ continuous but not differentiable,} \end{aligned}$$

where C^1 denotes that both $\mathbf{x}_{\mathrm{H}}$ and $\dot{\mathbf{x}}_{\mathrm{H}}$ are continuous. The notation C^w also defines that the resulting $\mathbf{x}_{\mathrm{H}}$ is *smooth* (continuous and differentiable up to w-th order). So *smoothness* defines the continuity and the differentiability; see figure 1-1. The above procedure of constructing a cubic Hermite spline is known as *Hermite cubic interpolation*.

While the interpolation polynomial at each segment for the predicted states must be differentiable, the interpolation polynomial for the predicted control does not need to be. It should have the same degree as the predicted dynamics or one degree lower [28]. The interpolation polynomial for the predicted control in a segment, denoted by $u_{\mathrm{H}}(t)$ or $\mathbf{u}_{\mathrm{H}}(t)$, can be expressed in monomial basis as follows:

$$u_{\mathrm{H}}(t) = \sum_{i=0}^{c} g_i \left(\frac{t - t_k}{h_{\mathrm{s}}}\right)^i, \quad t \text{ in } \left[t_k, t_{k+1}\right]. \tag{3.3–7}$$

Such a polynomial that does not need to satisfy the derivative of a function is known as *Lagrange interpolation polynomial*, or equivalent to zeroth-order Hermite interpolation polynomial. Thus, the spline formed by these piecewise interpolants has C^0 continuity, i.e., continuous but not differentiable.

3.3.1.2 Simpson Quadrature

The interpolants in equations (3.3–2) and (3.3–3) are constructed solely based on the information at the start and at the end points of a segment (where the end point of a segment is equal to the start point of the next segment). To ensure that these interpolants satisfy the predicted dynamics within a segment, the second fundamental theorem of calculus must be satisfied as follows:

$$\mathbf{x}_{k+1} - \mathbf{x}_k = \int_{t_k}^{t_{k+1}} \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t), U_{\mathrm{T}}, \chi\right) \mathrm{d}t. \tag{3.3–8}$$

This integral can be numerically approximated via any quadrature rule. In Hermite-Simpson collocation, as the name suggests, uses *Simpson quadrature* rule to approximate

this integration. This rule can be derived by approximating the integrand with a second-degree Lagrange interpolation polynomial. It can be expressed as follows:

$$\mathbf{x}_{k+1} - \mathbf{x}_k \approx \frac{h_\mathrm{s}}{6} \left(\mathbf{f}_k + 4\mathbf{f}_{k+0.5} + \mathbf{f}_{k+1} \right). \tag{3.3–9}$$

It is obvious that equation (3.3–9) is integrating $\mathbf{f}$ *implicitly*[10], that is, $\mathbf{f}_{k+1}$ depends on $\mathbf{x}_{k+1}$ that appears on the left-hand side of the equation.

Moreover, this quadrature requires the information at a new point: at the middle of the segment (at $t_{k+0.5}$). In the collocation method, this information can be either interpolated via the interpolation polynomial or searched directly as the optimization variables. The following sub-subsections explain this matter in detail.

3.3.1.3 Collocation Method: Transcription

Let us now combine the Hermite cubic interpolation and the Simpson quadrature in the preceding sub-subsections. Let us term all the knot points and the additional mid-segment points as the *discretization points*. Thus, the number of the discretization points, denoted by N_p, is $N_\mathrm{p} = 2N_\mathrm{s} + 1$. Moreover, the length between two consecutive discretization points is $h_\mathrm{p} = 0.5h_\mathrm{s}$.

Let s denote the s-th segment such that $s = 1, 2, \ldots, N_\mathrm{s}$ and let p denote the p-th discretization point such that $p = 1, 2, \ldots, N_\mathrm{p}$. In this notation, odd p is the knot point, i.e., $p = 2s-1$, while even p is the mid-segment point, i.e., $p = 2s$. It is very important to note that these two representations of the discretization points (by p and s) are equivalent and will be used interchangeably. Also, let t_p be the time at the p-th discretization point, so that $\mathbf{x}_p \equiv \mathbf{x}(t_p)$, $\mathbf{f}_p \equiv \mathbf{f}(t_p)$, and $\mathbf{u}_p \equiv \mathbf{u}(t_p)$ denote the predicted states, the predicted dynamics, and the predicted control, respectively. With the newly introduced notational system, one can rewrite the Simpson quadrature in equation (3.3–9) as follows:

$$\mathbf{x}_{2s+1} - \mathbf{x}_{2s-1} \approx \frac{h_\mathrm{s}}{6} \left(\mathbf{f}_{2s-1} + 4\mathbf{f}_{2s} + \mathbf{f}_{2s+1} \right). \tag{3.3–10}$$

Remember that up to this moment, only the information at the knot points is known, because it is one of the optimization variables. The information at the mid-segment point can be *interpolated* via the interpolants shown in equations (3.3–2) and (3.3–3) with the coefficients shown in equation (3.3–6). Given the information at the knot points, one can interpolate the information at the mid-segment point, i.e., at t_{2s}, with the following expressions (see appendix III.B):

$$\mathbf{x}_\mathrm{H}(t_{2s}) = \frac{1}{2} \left(\mathbf{x}_{2s-1} + \mathbf{x}_{2s+1} \right) + \frac{h_\mathrm{s}}{8} \left(\mathbf{f}_{2s-1} - \mathbf{f}_{2s+1} \right), \tag{3.3–11}$$

and,

$$\begin{aligned} \dot{\mathbf{x}}_\mathrm{H}(t_{2s}) &\triangleq \mathbf{f}_\mathrm{H}(t_{2s}) \\ &= -\frac{3}{2h_\mathrm{s}} \left(\mathbf{x}_{2s-1} - \mathbf{x}_{2s+1} \right) - \frac{1}{4} \left(\mathbf{f}_{2s-1} + \mathbf{f}_{2s+1} \right). \end{aligned} \tag{3.3–12}$$

[10] In an IVP (initial value problem), one only considers the initial conditions. In this case, one can opt to integrate using any explicit scheme like forward Euler's method or Runge-Kutta method, which is easier. However, we are dealing with a BVP at each segment where the boundary conditions are split, i.e., conditions at the knot points are known.

If the predicted control at the mid-segment point $\mathbf{u}_{2s}$ is added as one of the optimization variables, then one can compute the predicted dynamics at the mid-segment point $\mathbf{f}_{2s}$ with equation (3.3–1b) and substitute it to equation (3.3–10). However, the resulting equations are quite complicated. Such a treatment results in the *compressed* Hermite-Simpson collocation method because the predicted states at the mid-segment point $\mathbf{x}_{2s}$ is interpolated by the Hermite interpolant (3.3–2). In this compressed method, the optimization variables are the predicted states at the knot points, and the predicted control at the knot points and at the mid-segment points (all discretization points).

Another way is by adding both $\mathbf{x}_{2s}$ and $\mathbf{u}_{2s}$ as optimization variables along with those at the knot points. Since the predicted states in a segment is approximated with equation (3.3–2), then $\mathbf{x}_{2s}$ must be equal to equation (3.3–11). This results in an equality constraint called *Hermite interpolation constraint* as follows:

$$\mathbf{x}_{\mathrm{H}}\left(t_{2s}\right) = \mathbf{x}_{2s}. \tag{3.3–13}$$

If this Hermite interpolation constraint is satisfied, then at the knot points, the interpolation polynomial $\mathbf{x}_{\mathrm{H}}$ is equal to the predicted states $\mathbf{x}$, and $\dot{\mathbf{x}}_{\mathrm{H}}$ is equal to the predicted dynamics $\mathbf{f}$, which is exactly what we want to do (see sub-subsection 3.3.1.1).

With $\mathbf{x}_{2s}$ that satisfies the Hermite interpolation constraint (3.3–13) and $\mathbf{u}_{2s}$, one can calculate directly the predicted dynamics $\mathbf{f}_{2s}$ (3.3–1b). This must be equal to equation (3.3–12), otherwise $\dot{\mathbf{x}}_{\mathrm{H}} \neq \mathbf{f}_{\mathrm{H}}$, which is not true. The resulting equation is an equality constraint as follows:

$$\mathbf{f}_{\mathrm{H}}\left(t_{2s}\right) = \mathbf{f}_{2s}. \tag{3.3–14}$$

There is an interesting fact here, equation (3.3–14) results in the same expression as the Simpson quadrature shown in equation (3.3–10). This is the reason why equation (3.3–14) is called *Simpson quadrature constraint*. If equation (3.3–14) is satisfied, it is equivalent to implicitly integrating the predicted dynamics within the segment (see appendix III.C).

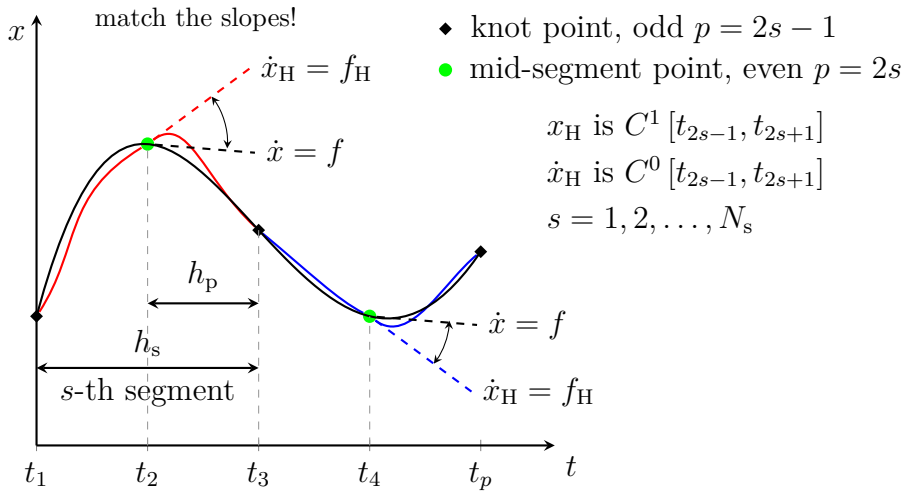


Figure 3-2. Hermite-Simpson collocation method: match the slopes.

This particular scheme is known as *separated* Hermite-Simpson collocation method, which is implemented in this study. In this method, one must look for the predicted states and the predicted control at all of the discretization points. Note that at each of these points, the derivative of the interpolation polynomial $\dot{\mathbf{x}}_{\mathrm{H}}$ must be equal to the predicted dynamics $\mathbf{f}$. This is why, in Hermite-Simpson collocation method, the discretization points are also the collocation points.

In summary, within a segment, the predicted state trajectory $\mathbf{x}(t)$ (shown as black curves in figure 3-2) is approximated as cubic Hermite interpolation polynomial $\mathbf{x}_{\mathrm{H}}$ (shown as red curve and blue curve). At the mid-segment point, the slope of the interpolation polynomial $\dot{\mathbf{x}}_{\mathrm{H}}$ should match with the slope of the predicted states $\dot{\mathbf{x}}$ (predicted dynamics $\mathbf{f}$). This is visualized in figure 3-2 where the red (and blue) dashed line should match the black dashed line at the mid-segment points. If the slopes match, then $\mathbf{x}_{\mathrm{H}}\left(\bar{t}_p\right)$ and $\dot{\mathbf{x}}_{\mathrm{H}}\left(\bar{t}_p\right) \triangleq \mathbf{f}_{\mathrm{H}}\left(\bar{t}_p\right)$ match $\mathbf{x}_p$ and $\mathbf{f}_p$, respectively, at all the discretization points $p = 1, 2, \ldots, N_{\mathrm{p}}$. This is why in this method all the discretization points are also the collocation points. Moreover, the piecewise interpolants at all segments form a continuous cubic Hermite spline that is once-differentiable everywhere in the time domain, i.e., C^1 continuity. In addition to the constraints of the OCP shown in equation (3.3–1), if the number of the state variables is 6, then there are $6 \times N_{\mathrm{s}}$ additional Hermite interpolation constraints (3.3–13) and $6 \times N_{\mathrm{s}}$ additional Simpson quadrature constraints (3.3–14).

3.3.2 Point-in-Polygon: Allision Constraints

This subsection discusses how to form the necessary constraints such that the resulting optimal state trajectory is safe from allision. Suppose $\mathbf{P}_{\mathrm{b}} \in \mathbb{R}^{2\times n_{\mathrm{v}}}$ is an array as follows:

$$\mathbf{P}_{\mathrm{b}} = \begin{bmatrix} \mathbf{p}_{\mathrm{b},1} & \mathbf{p}_{\mathrm{b},2} & \cdots & \mathbf{p}_{\mathrm{b},n_{\mathrm{v}}} \end{bmatrix}^{\top},$$

where the columns, $\mathbf{p}_{\mathrm{b}} = \begin{bmatrix} x_{\mathrm{b}} & y_{\mathrm{b}} \end{bmatrix}^{\top}$, are the position of the vertices that define a simple polygon in the earth-fixed coordinate system. This polygon represents the boundary of the harbor as shown in figure 3-3. Note that the number of the vertices, $n_{\mathrm{v}} \geq 3$ is necessary to form a closed region.

Let $\mathbf{p}_{\mathrm{s}} = \begin{bmatrix} x_{\mathrm{s}} & y_{\mathrm{s}} \end{bmatrix}^{\top}$ be the position of a point[11] of the ship in the earth-fixed coordinate system. This is the point that one wants to check whether it is inside or outside the polygon formed by $\mathbf{P}_{\mathrm{b}}$. Let $\theta_{\mathrm{s},i}$ denote the angle between two lines formed by $\mathbf{p}_{\mathrm{s}}$ and two adjacent vertices of the polygon, i.e., $\mathbf{p}_{\mathrm{b},i}$ and $\mathbf{p}_{\mathrm{b},i+1}$. In other words, $\theta_{\mathrm{s},i}$ is the angle between two lines: $\mathbf{p}_{\mathrm{s}} - \mathbf{p}_{\mathrm{b},i}$ and $\mathbf{p}_{\mathrm{s}} - \mathbf{p}_{\mathrm{b},i+1}$. If $\mathbf{p}_{\mathrm{s}}$ is inside the polygon, then the sum of all $\theta_{\mathrm{s},i}$ should be equal to 2π. where $i = 1, 2, \ldots, n_{\mathrm{v}}$. This is expressed as follows:

$$\sum_{i=1}^{n_{\mathrm{v}}} \theta_{\mathrm{s},i} = 2\pi. \tag{3.3–15}$$

Instead of just a point/particle, the ship is represented by multiple points. Let $\mathbf{P}_{\mathrm{s}} \in \mathbb{R}^{2\times n_{\mathrm{s}}}$ be the array whose columns are the position of the points of the ship as follows:

$$\mathbf{P}_{\mathrm{s}} = \begin{bmatrix} \mathbf{p}_{\mathrm{s},1} & \mathbf{p}_{\mathrm{s},2} & \cdots & \mathbf{p}_{\mathrm{s},n_{\mathrm{s}}} \end{bmatrix}^{\top},$$

[11] Do not confuse with the discretization point in the collocation method.

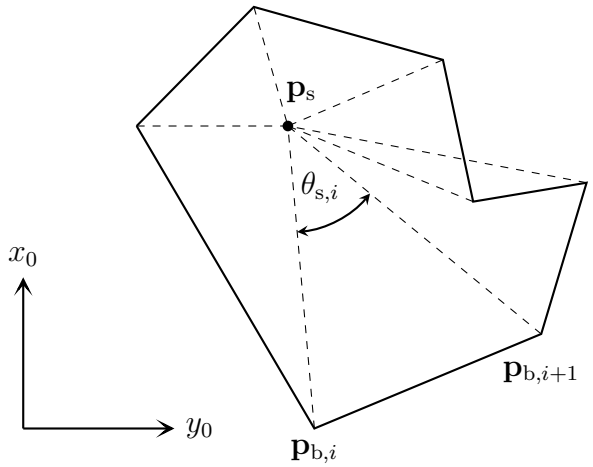


Figure 3-3. A point $\mathbf{p}_\mathrm{s}$ is inside the polygon if the sum of all $\theta_{\mathrm{s},i}$ (for $i = 1, 2, \ldots, n_\mathrm{v}$) is equal to 2π.

where $n_\mathrm{s} \geq 0$ denotes the number of the points[12] that define the ship. More points imply more stringent constraints that ensure more safety. These points are calculated based on the position of midship (x_0, y_0) and the yaw angle (ψ), which belong to the predicted states, i.e., the optimization variables. This implies that $\theta_{\mathrm{s},i}$ is a function of some of the optimization variables. For this reason, one can construct the following additional equality constraints as follows:

$$\sum_{i=1}^{n_\mathrm{v}} \theta_{\mathrm{s},i} \Big|_{j,p} = 2\pi. \tag{3.3–16}$$

Equation (3.3–16) basically says that at each discretization point $p = 1, 2, \ldots, N_\mathrm{s}$, equation (3.3–15) must be evaluated for all points of the ship $j = 1, 2, \ldots, n_\mathrm{s}$. This results in $n_\mathrm{s} \times N_\mathrm{p}$ equality constraints.

As a note, this is a *point-in-polygon* method that is also known as *angle summation* method. It is equivalent to *winding number* shown in [58]. In computer-aided design (CAD), this method is arguably not the best method to check whether a point is inside or outside a polygon. However, within the scope of the study, it is one of the most practical methods: analytically differentiable and applicable to any shape of harbor area (convex or nonconvex).

[12]One does not need to define the ship as a closed polygon, i.e., just a particle is also possible. This is why we use the term *point* instead of *vertex* here. A point is just a position in space, while a vertex is shared with multiple points: like how one defines a polygon.

3.3.3 Transcribed OCP: the NLP in the SO-TP

With the additional constraints constructed in the previous two subsections, one can discretize and transcribe the OCP in the SO-TP (3.3–1) into the following finite-dimensional NLP:

$$
\begin{aligned}
\text{determine} \quad & \delta_1^*, \delta_2^*, \ldots, \delta_{N_\mathrm{p}}^* \in \mathbb{R}, \\
& n_1^*, n_2^*, \ldots, n_{N_\mathrm{p}}^* \in \mathbb{R}, \\
& \mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_{N_\mathrm{p}}^* \in \mathbb{R}^6,\ t_\mathrm{f} \in \mathbb{R} \\
\text{that min.} \quad & J_\mathrm{on} = \sum_{i=1}^{6} \left(x_i\left(t_\mathrm{f}\right) - x_{\mathrm{fin},i}\right)^2 \times \int_0^{t_\mathrm{f}} \sum_{i=1}^{6} \left(x_i\left(t\right) - x_{\mathrm{fin},i}\right)^2 \mathrm{d}t, && \text{(3.3–17a)} \\
\text{subject to} \quad & \mathbf{x}_1^* = \bar{\mathbf{x}}_j,\ \mathbf{x}_{N_\mathrm{p}}^* = \mathbf{x}_\mathrm{fin}, && \text{(3.3–17b)} \\
& \mathbf{x}_{N_\mathrm{p}-2} = \mathbf{x}_\mathrm{fin} - \frac{h_\mathrm{s}}{6}\left(\mathbf{f}_{N_\mathrm{p}-2} + 4\mathbf{f}_{N_\mathrm{p}-1} + \mathbf{f}_{N_\mathrm{p}}\right), && \text{(3.3–17c)} \\
& \mathbf{x}_\mathrm{H}\left(t_{2s}\right) = \mathbf{x}_{2s}, \quad \text{for } s = 1, 2, \ldots, N_\mathrm{s}, && \text{(3.3–17d)} \\
& \mathbf{f}_\mathrm{H}\left(t_{2s}\right) = \mathbf{f}_{2s}, \quad \text{for } s = 1, 2, \ldots, N_\mathrm{s}, && \text{(3.3–17e)} \\
& \sum_{i=1}^{n_\mathrm{v}} \theta_{\mathrm{s},i}\Big|_{j,p} = 2\pi, \quad \text{for } j = 1, 2, \ldots, n_\mathrm{s}, && \text{(3.3–17f)} \\
& -\delta_\mathrm{max} \le \delta_p \le \delta_\mathrm{max}, \quad \text{for } p = 1, 2, \ldots, N_\mathrm{p}, && \text{(3.3–17g)} \\
& -n_\mathrm{max} \le n_p \le n_\mathrm{max}, \quad \text{for } p = 1, 2, \ldots, N_\mathrm{p}, && \text{(3.3–17h)} \\
& \text{and } t_\mathrm{f} \in (0, \infty)\,. && \text{(3.3–17i)}
\end{aligned}
$$

Notice that now the continuous OCP becomes a finite-dimensional NLP. The form of the objective function in equation (3.3–17a) is the same as that of the OCP (3.3–1a). The integral term in the objective function is numerically integrated with Simpson quadrature rule. The optimization variables are the control commands $\mathbf{u}_p^* = \begin{bmatrix} \delta_p^* & n_p^* \end{bmatrix}^\top$ and the states $\mathbf{x}_p^*$ at each of the discretization points, along with the final time t_f. In total, there are $N_\mathrm{p}(2+6)+1$ optimization variables.

The 6×2 equalities in equations (3.3–17b) and (3.3–17c) ensure that the initial states and the desired final states are included in the optimal state trajectory. Meanwhile, equations (3.3–17d) and (3.3–17e) are the $6 \times N_\mathrm{s}$ Hermite interpolation constraints and the $6 \times N_\mathrm{s}$ Simpson quadrature constraints, respectively. The $n_\mathrm{s} \times N_\mathrm{p}$ equalities shown in equation (3.3–17f) make sure that all the n_s points of the ship are inside the harbor area, otherwise allision occurs. The $N_\mathrm{p} \times 2$ box constraints (inequalities) shown in equations (3.3–17g) and (3.3–17h) saturate the rudder angle and the propeller revolution number to the given values, respectively. In total, there are $6 \times (2 + 2N_\mathrm{s}) + (n_\mathrm{s} \times N_\mathrm{p})$ equality constraints and $N_\mathrm{p} \times 2$ inequality constraints; all explicitly defined and enforced.

The NLP in equation (3.3–17) is solved by SQP (sequential quadratic programming) that is available in MATLAB `fmincon` package. It is a gradient-based method with quadratic local convergence rate[13] [59]. With a good initial guess to warm-start the op-

[13] This method ensures that once a feasible direction (during the gradient descent) is found, the suc-

timization, even fewer iterations are expected such that the optimization converges even faster to the local optimum (see footnote 7).

3.3.4 Conditions for Simulations

In this study, only one offline solution from the Off-TP, i.e., $\mathbf{x}^*_{\text{off}}$, $\mathbf{u}^*_{\text{off}}$, and $t_{\text{f,off}}$, will be used to warm-start several different cases (evaluation point 1). The offline solution was obtained at an ideal situation where $U_{\text{T}} = 0$ and $\chi = 0$ (no-wind condition). The initial states $\mathbf{x}^*_{\text{off}}(0)$, the desired final states $\mathbf{x}_{\text{fin}}$, the final states $\mathbf{x}^*_{\text{off}}(t_{\text{f,off}})$ of the offline solution are given as follows:

$$\mathbf{x}^*_{\text{off}}(0) = [\ 16.50\ \ 0.12\ \ -7.50\ \ 0.00\ \ 2\pi/3\ \ 0.00\ ]^\top \tag{3.3–18a}$$

$$\mathbf{x}_{\text{fin}} = [\ -0.50\ \ 0.01\ \ -0.50\ \ 0.00\ \ \pi\ \ 0.00\ ]^\top \tag{3.3–18b}$$

$$\mathbf{x}^*_{\text{off}}(t_{\text{f,off}}) = [\ -0.51\ \ 0.01\ \ -0.51\ \ 0.00\ \ 3.14\ \ 0.00\ ]^\top \tag{3.3–18c}$$

$$t_{\text{f,off}} = 160\ \text{s}. \tag{3.3–18d}$$

At the final docking point, the ship must have a velocity normal to the berth's wall between 0.03 m/s to 0.04 m/s [64]. In the simulation, this normal velocity is just the sway velocity v_{m} because at the final docking point, the ship is parallel to the berth's wall. The surge velocity u, on the other hand, should be as low as possible but does not need to be zero. This is because the port mooring/rigging will be used to completely fix the ship to the berth. For this reason, the desired surge velocity at the final docking point (3.3–18b) is set to 0.01 m/s (approximately equal to 0.1 m/s for a full-scale ship) to relax the NLP.

To evaluate the performance of the warm-started SO-TP, it is simulated for four main cases: M1 to M4, and ten additional trial cases that were chosen arbitrarily: A1 to A10. The initial states for each of the cases are expressed in terms of how far it is from $\mathbf{x}^*_{\text{off}}(0)$. Since $\bar{\mathbf{x}}_j \triangleq \bar{\mathbf{x}}\left(\bar{t}_j\right)$, this can be written as follows:

$$\bar{x}_i\left(\bar{t}_j\right) = d_i\, x^*_{\text{off},i}(0)\,, \tag{3.3–19}$$

where d_i is the element of a vector of multipliers $\mathbf{d}$. One can quantify how different the initial states at each case is from $\mathbf{x}^*_{\text{off}}(0)$ by computing the Euclidean norms L_2 as follows:

$$\begin{aligned} L_2 &= \sqrt{\sum_{i=1}^{6}\left[\frac{\left(\bar{x}_i\left(\bar{t}_j\right) - x^*_{\text{off},i}(0)\right)}{x^*_{\text{off},i}(0)}\right]^2} \\ &= \sqrt{\sum_{i=1}^{6}(d_i - 1)^2}. \end{aligned} \tag{3.3–20}$$

The summary of the simulation conditions: initial states and wind parameters for each of the cases are shown in table 3-2.

cessive solutions are guaranteed to converge to a value quickly: fewer iterations.

Table 3-2. Cases in evaluating the performance of the SO-TP.

Case	Multiplier $\mathbf{d}$ $\bar{x}_i\left(\bar{t}_j\right) = d_i\, x^*_{\mathrm{off},i}\left(0\right)$	Initial velocity $\bar{u}\left(0\right)$ (m/s)	Initial velocity $\bar{u}\left(0\right)$ Full scale (knots)	$\bar{\psi}\left(0\right)$	$U_{\mathrm{T}}\left(\bar{t}_j\right)$ (m/s)	$\chi\left(\bar{t}_j\right)$ (deg)	Norm $L_2 = \sqrt{\sum_{i=1}^{6}\left(d_i-1\right)^2}$
M1	$[1.0\ \ 1.0\ \ 1.0\ \ 1.0\ \ 1.0\ \ 1.0]^\top$	0.12	2.43	$2\pi/3$	No wind		0.00
M2	$[1.1\ \ 2.0\ \ 1.1\ \ 1.0\ \ 1.2\ \ 1.0]^\top$	0.24	4.86	$4\pi/5$	0.75	45	1.03
M3	$[0.9\ \ 2.0\ \ 0.8\ \ 1.0\ \ 0.9\ \ 1.0]^\top$	0.24	4.86	$3\pi/5$	0.50	225	1.03
M4	$[1.0\ \ 3.0\ \ 0.9\ \ 1.0\ \ 1.5\ \ 1.0]^\top$	0.36	7.28	π	0.75	45	2.06
A1	$[1.2\ \ 1.2\ \ 1.1\ \ 1.0\ \ 1.1\ \ 1.0]^\top$	0.14	2.83	$11\pi/15$	0.75	45	0.32
A2	$[1.2\ \ 1.4\ \ 1.2\ \ 1.0\ \ 1.2\ \ 1.0]^\top$	0.17	3.44	$4\pi/5$	0.75	180	0.53
A3	$[1.3\ \ 1.6\ \ 0.9\ \ 1.0\ \ 1.4\ \ 1.0]^\top$	0.19	3.84	$14\pi/15$	0.75	125	0.79
A4	$[1.4\ \ 1.8\ \ 0.8\ \ 1.0\ \ 0.9\ \ 1.0]^\top$	0.22	4.45	$3\pi/5$	0.50	135	0.92
A5	$[1.5\ \ 0.8\ \ 1.2\ \ 1.0\ \ 1.1\ \ 1.0]^\top$	0.10	2.02	$11\pi/15$	0.50	90	0.58
A6	$[1.0\ \ 0.8\ \ 0.7\ \ 1.0\ \ 1.2\ \ 1.0]^\top$	0.10	2.02	$4\pi/5$	0.50	315	0.41
A7	$[1.2\ \ 2.2\ \ 0.8\ \ 1.0\ \ 0.8\ \ 1.0]^\top$	0.26	5.26	$8\pi/15$	0.50	250	1.25
A8	$[1.2\ \ 2.0\ \ 0.0\ \ 1.0\ \ 1.5\ \ 1.0]^\top$	0.24	4.86	π	0.50	90	1.51
A9	$[1.0\ \ 1.0\ \ -1.0\ \ 1.0\ \ 2.0\ \ 1.0]^\top$	0.12	2.43	$4\pi/3$	0.75	0	2.24
A10	$[0.8\ \ 1.5\ \ -0.9\ \ 1.0\ \ 2.2\ \ 1.0]^\top$	0.18	3.64	$22\pi/15$	0.50	45	2.31

3.3.5 Defining the Harbor's Wall and the Ship's Points

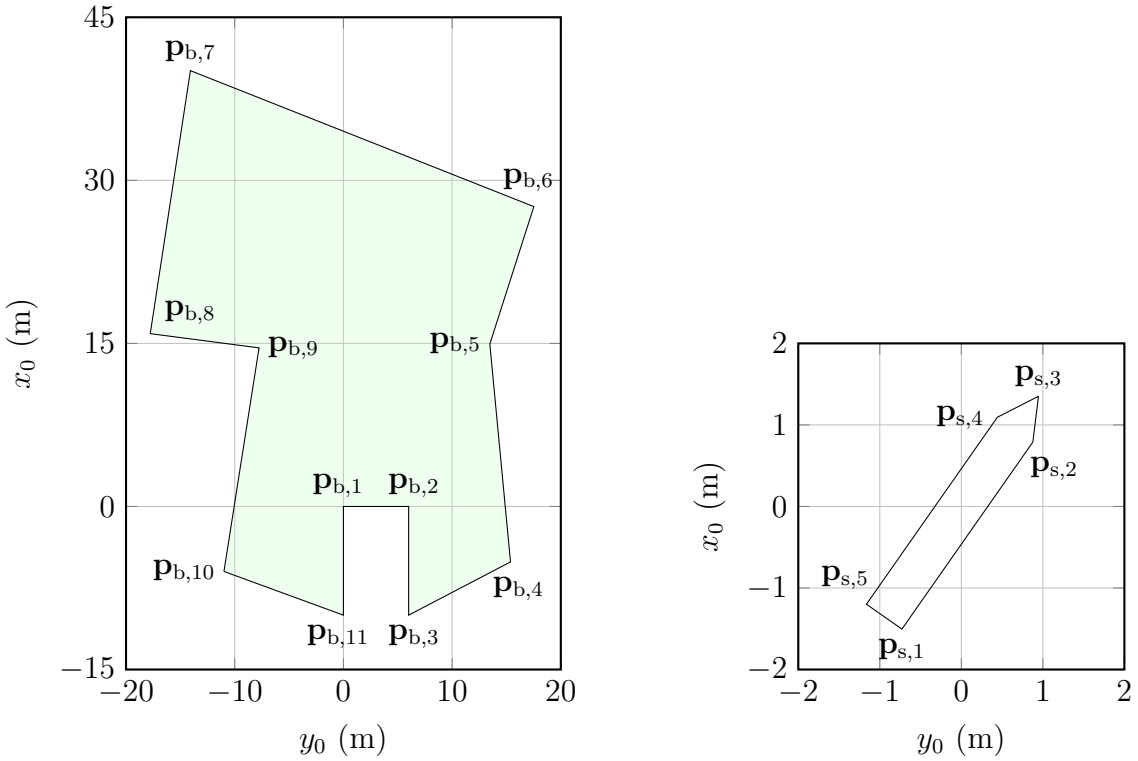
This section explains the numerical parameters related to the allision constraints: how to define the geometry of the harbor and the ship. In this study, the docking operation is simulated in the Inukai experiment pond of Osaka University. From the satellite image, the shape of the pond can be approximated as a simple polygon $\mathbf{P}_{\mathrm{b}}$ with $n_{\mathrm{v}} = 11$ vertices. This is shown in figure 3-4a. In the simulation, the origin of the earth-fixed coordinate system coincides with $\mathbf{p}_{\mathrm{b},1}$.

Moreover, one must consider carefully how to choose the number of the points n_{s} that define the ship. More points guarantee a safer trajectory at the cost of computational time because the number of allision constraints is increased by N_{p} for every additional point. In the simulation, the ship is represented as a closed polygon with $n_{\mathrm{s}} = 5$ points, which is considered sufficient. This is shown in figure 3-4b; note that the scale between the two figures is different.

3.3.6 Discretization and Interpolation

In collocation method, one must divide the trajectory into several segments. In the SO-TP, the trajectory is divided into $N_{\mathrm{s}} = 20$ segments, such that there are $N_{\mathrm{p}} = 41$ discretization points. Recall that the NLP only searches for the optimal states and the optimal control command vector at the discretization points. After solving the optimization, one must interpolate the values between the discretization points.

In the Off-TP, the control within a segment is assumed constant (see section 3.2). However, this is not the case with the SO-TP. As shown in equation (3.3–7), one can approximate the control within a segment by an interpolation polynomial $\mathbf{u}_{\mathrm{H}}$ that has the same degree as the dynamics or one degree lower, but need not be differentiable. Now, there are three points within a segment, i.e., $b = 3$ and $w = 1$. From equation (3.3–4), the required degree of $\mathbf{u}_{\mathrm{H}}$ is $c = 2$, or a quadratic polynomial.



(a) Allision-free region formed by a simple polygon $\mathbf{P}_{\mathrm{b}}$ with $n_{\mathrm{v}} = 5$ vertices

(b) The points that represent the ship, $\mathbf{P}_{\mathrm{s}}$ with $n_{\mathrm{s}} = 5$ points

Figure 3-4. The geometry of the Inukai experiment pond and the ship when generating allision constraints. Scales are different.

In consideration of the mechanical limitation of the actuators (gears, engine, etc.), the control between two consecutive discretization points is assumed linear. This means, the interpolation polynomial $\mathbf{u}_{\mathrm{H}}$ has a degree of $c = 1$, i.e., a first-degree Lagrange interpolation polynomial. This treatment is equivalent to dividing the segment into two sub-segments where $b = 2$ and $w = 0$, which indeed gives $c = 1$. A linear interpolation polynomial has two coefficients (g_1 and g_2) in equation (3.3–7) that can be solved with two equations as follows:

$$\mathbf{u}_{\mathrm{H}}(t_p) = \mathbf{u}^*_p \tag{3.3–21a}$$

$$\mathbf{u}_{\mathrm{H}}(t_{p+1}) = \mathbf{u}^*_{p+1}. \tag{3.3–21b}$$

Upon solving for the coefficients, one can construct a linear equation to interpolate the control between two consecutive discretization points, denoted by $\mathbf{u}^*_{\mathrm{H}}$, as follows:

$$\mathbf{u}^*_{\mathrm{H}}(t) = \mathbf{u}^*_p + \left(\mathbf{u}^*_{p+1} - \mathbf{u}^*_p\right) \frac{t - t_p}{h_{\mathrm{p}}}, \quad t \text{ in } [t_p, t_{p+1}], \tag{3.3–22}$$

for $p = 1, 2, \ldots, N_{\mathrm{p}} - 1$.

In the NLP, one has used an interpolation polynomial with $b = 2$ points in a segment. However, upon solving the NLP, because of the point at the mid-segment, one must construct an interpolation polynomial $\mathbf{x}_{\mathrm{H}}$ to interpolate the optimal states with

$b = 3$ points in a segment. This $\mathbf{x}_{\mathrm{H}}$ must also be differentiable up to $w = 1$. Thus, with equation (3.3–4), the required degree for this is $c = 5$. It is a fifth-degree Hermite interpolation polynomial with six coefficients that require six equations to obtain. Of course, one has all the information required to obtain these coefficients: the optimal states $\mathbf{x}_p^*$ and its corresponding dynamics $\mathbf{f}_p^*$ at all the three points. The resulting interpolant will be a fifth-degree polynomial that is somehow complicated.

Fortunately, there is a preferred interpolation method that retains the accuracy of the Hermite cubic interpolation method. This method is widely known as *natural cubic spline interpolation* [63]. The resulting polynomial has a higher order of continuity: C^2, that is, continuous up to its second derivative/differentiable twice. One can construct a natural cubic spline interpolant by integrating a second-degree Lagrange polynomial in Lagrange basis [29, 63]. The interpolant to interpolate the optimal states —denoted by $\mathbf{x}_{\mathrm{H}}^*$—within a segment, i.e., t in $[t_p, t_{p+2}]$, can be expressed as follows:

$$\begin{aligned}\mathbf{x}_{\mathrm{H}}^*(t) = \mathbf{x}_p^* + \mathbf{f}_p^*(t - t_p) - \left(3\mathbf{f}_p^* - 4\mathbf{f}_{p+1}^* + \mathbf{f}_{p+2}^*\right)\frac{(t-t_p)^2}{2h_{\mathrm{s}}} \\ + \left(2\mathbf{f}_p^* - 4\mathbf{f}_{p+1}^* + 2\mathbf{f}_{p+2}^*\right)\frac{(t-t_p)^3}{3h_{\mathrm{s}}^2},\end{aligned} \tag{3.3–23}$$

for all odd p such that $p = 2s-1$ where $s = 1, 2, \ldots, N_{\mathrm{s}}$. Here $\mathbf{f}_p^*$ denotes the dynamics $\mathbf{f}_p^* \equiv \mathbf{f}\left(\mathbf{x}_p^*, \mathbf{u}_p^*, U_{\mathrm{T}}, \chi\right)$ that is evaluated at the p-th discretization point with the corresponding optimal states and optimal control command vector.

3.4 Result of the Simulations and Evaluation

This section is dedicated to the discussion and the evaluation of the performance of the warm-started SO-TP. In accordance with evaluation point 1, the simulations are intended to demonstrate the applicability of just one almost-globally optimal solution as a warm start for several different cases in the SO-TP (refer to table 3-2). This almost-globally optimal solution was obtained for a differently formulated problem in the Off-TP. These differences include the transcription method, the objective function, and how the control is assumed in a segment. Note that in this subsection we shall use the terms offline solution and warm start to refer to the same entity.

Because we focus on the computational time affected by the warm start, it is necessary to quantify the speedup relative to that without a warm start (cold start). Let T_{w1} be the computational time with a warm start from the Off-TP and let T_{cs} be the computational time without a warm start (linear initial guess). With these notations, one can compute the relative computational speedup with the following equation:

$$\%\,\text{comp. speedup} = \frac{T_{\mathrm{cs}} - T_{\mathrm{w1}}}{T_{\mathrm{cs}}} \times 100\%. \tag{3.4–1}$$

Moreover, a cold start can be in any arbitrary form of initial guess. In this study, it was a linear guess where the states are linearly changing from the given initial states to the desired final docking states. The initial guess for the control was constant, that is, $0.5\delta_{\max}$ and $0.5n_{\max}$ for the rudder angle and the propeller revolution number, respectively. For the final time t_{f}, the initial guess was 160 s, same to that of the warm start/offline solution.

3.4.1 Case M1: Same Initial States with no Wind

Before testing the warm-started SO-TP for other cases, it is necessary to verify whether it is properly defined. With the same initial states and no-wind condition of which the warm start was obtained (see 3.3–18b), the warm-started SO-TP must converge to a solution that is similar to the warm start. The optimal control and the optimal state trajectory planned by the SO-TP with and without a warm start are shown in figures 3-5 and 3-6, where the legend CMA-ES guess refers to the warm start/offline solution from the Off-TP.

As can be observed, the warm-started SO-TP indeed converged to the solution (shown in red curves) that is very much similar to the warm start. Compared to the SO-TP without a warm start (shown in blue curves), it converged relatively quickly: less than 0.5 s. This is a direct and clear demonstration to show that the solution of the OCP in the SO-TP indeed belongs to the feasible set of the OCP in the Off-TP, despite being formulated differently from the OCP in the Off-TP in all aspects: the transcription method, the objective function, and how the control is assumed.

3.4.2 Case M2: Farther Initial States with Wind Blows Away from the Berth

In this case, when the SO-TP is executed, the ship is at an initial location farther from the berth and with an initial yaw angle that is less sharp relative to the berth. The ship moves at initial surge velocity twice to that of the warm start, i.e., $\bar{u}\,(0) = 2u_{\text{off}}\,(0)$. The wind blows with a speed of 0.75 m/s at an angle of 45 deg (quarterly wind) away from the berth. The optimal control and optimal state trajectory from the SO-TP are shown in figures 3-7 and 3-8. As can be seen, the time t_{f} required to reach the desired final docking states $\mathbf{x}_{\text{fin}}$ is shorter because the initial surge velocity is doubled (0.24 m/s which is equivalent to 4.86 knots for the full-scale ship), even if the initial location is farther.

With the given warm start, the SO-TP converged to a feasible and allision-free solution in a much shorter computational time compared with that without a warm start. This shows that the one warm start is also applicable to warm-start the SO-TP in a situation where wind disturbance exists and the initial states are different with Euclidean norm $L_2 = 1.03$.

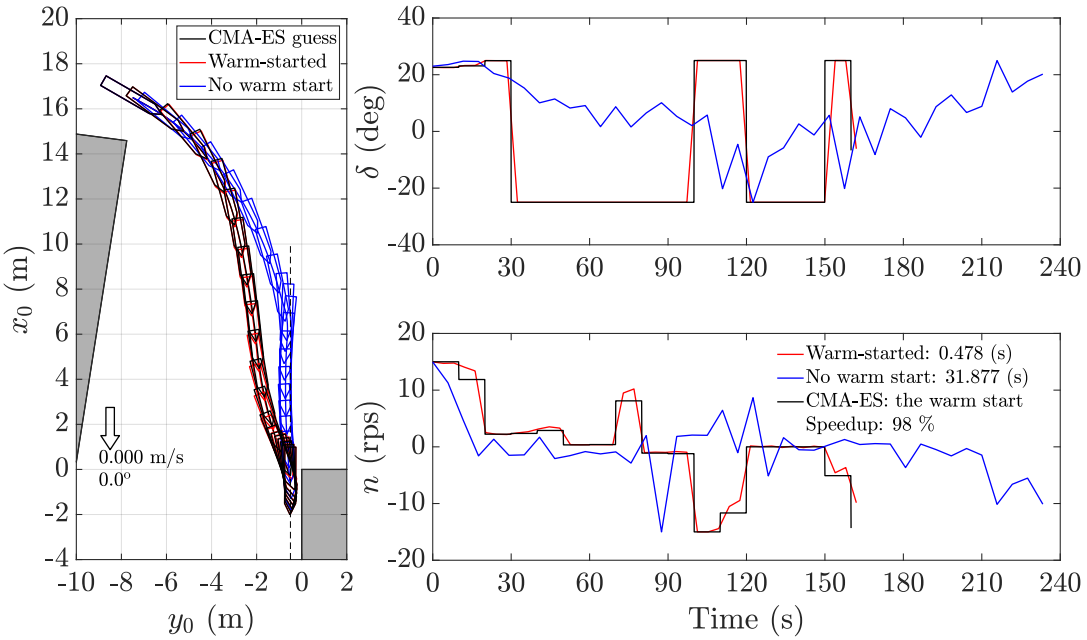


Figure 3-5. Visualization and optimal control for case M1 with computational time of 0.478 s.

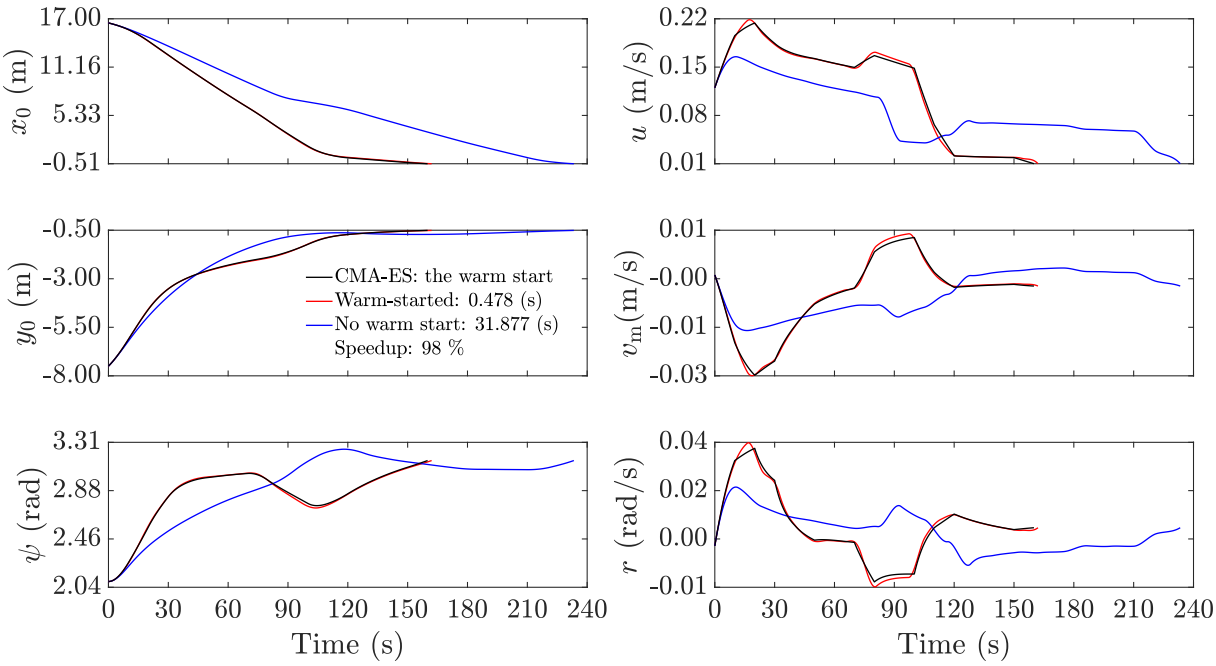


Figure 3-6. Summary of optimal state trajectory for case M1 with computational time of 0.478 s.

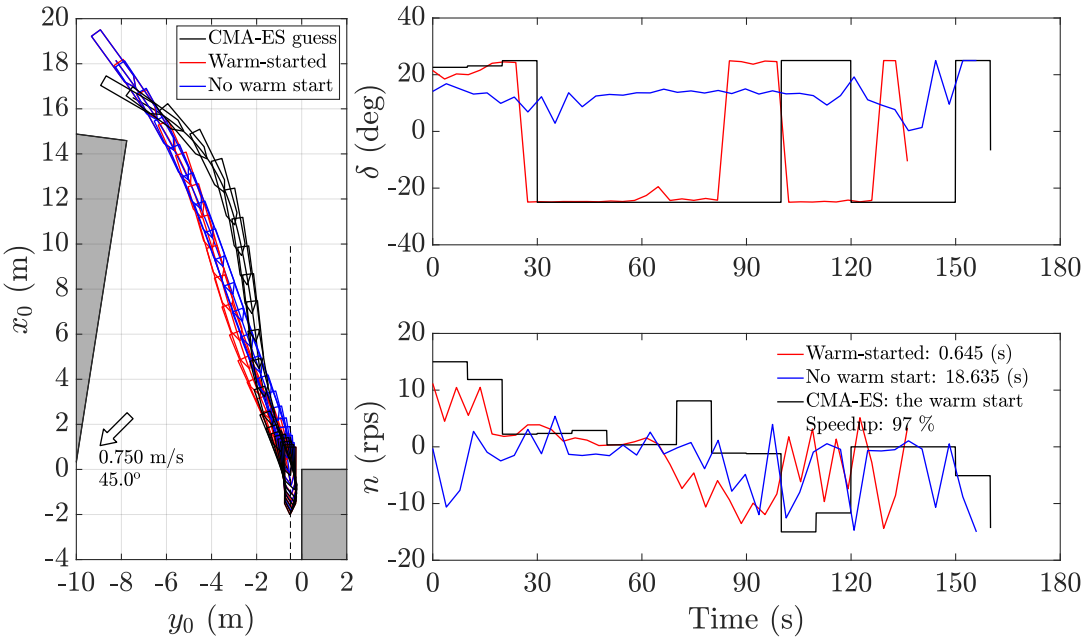


Figure 3-7. Visualization and optimal control for case M2 with computational time of 0.645 s.

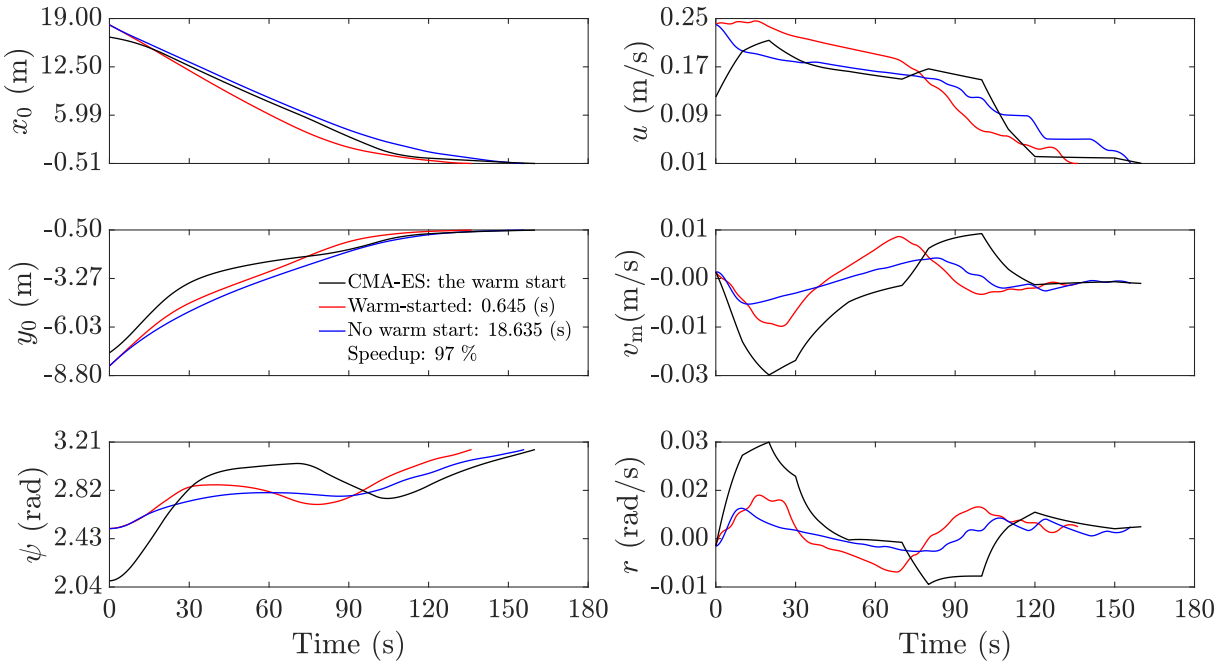


Figure 3-8. Summary of optimal state trajectory for case M2 with computational time of 0.645 s.

3.4.3 Case M3: Closer Initial States with Wind Blows to the Berth

In this case, the ship starts at an initial position that is closer to the berth and at a yaw angle that is almost perpendicular to the berth. Such an initial condition restricts the waterway for the ship to maneuver. Similar to case M2, the ship's initial surge velocity is 0.24 m/s. Moreover, the wind blows with a speed of 0.5 m/s at an angle 225 deg: quarterly wind to the berth. The optimal control and optimal state trajectory for this case are visualized in figures 3-9 and 3-10.

In this case, the problem becomes more difficult, especially for a conventional underactuated ship because it has to execute a sharp turn in such a relatively narrow and restricted area. This is hinted by the longer computational time for the SO-TP to converge to the solution. Nevertheless, the warm start was still able to speed up the computational time 68 % faster than that without a warm start. Moreover, the time t_f is shorter than that of the warm start because, similar to case M2, the initial surge velocity is faster. A small thrust from the propeller can give the ship enough momentum to execute a sharp turning maneuver.

It is interesting to see the same tendency (see figures 3-7 and 3-9) that the optimal rudder angle history δ is similar to the warm start; either stretched out or squeezed in with respect to the time. Intuitive reasoning for this interesting tendency is that the provided warm start for δ gives the value of the objective function that lies in the neighborhood of the curvature of the Hermite interpolation constraint (3.3–17d) and the Simpson quadrature constraint (3.3–17e), while that for n may be farther from the neighborhood of these constraints. This tendency also demonstrates the significance of the thrust from the propeller to the maneuverability of the ship.

3.4.4 Case M4: Starts Parallel with Wind Blows Away from Berth

In this case, the ship starts at the same location as the initial location of the warm start but at a different yaw angle: parallel to the berth. The ship starts an even higher surge velocity $\bar{u}\,(0) = 0.36\,\mathrm{m/s}$; equivalent to 7.28 knots for the full-scale ship. Such an initial states is very different from that of the warm start with a norm of $L_2 = 2.06$. Moreover, the wind blows away from the berth with a speed of 0.75 m/s. The optimal control and the optimal state trajectory for this case are shown in figures 3-11 and 3-12.

The warm-started SO-TP still gave a significantly faster computational time compared to that without a warm start; almost 51 %. This demonstrates that the same warm start used in the previous cases is still applicable to be a good starting point for the solver to start the gradient descent, even in a different condition from that when obtaining the warm start.

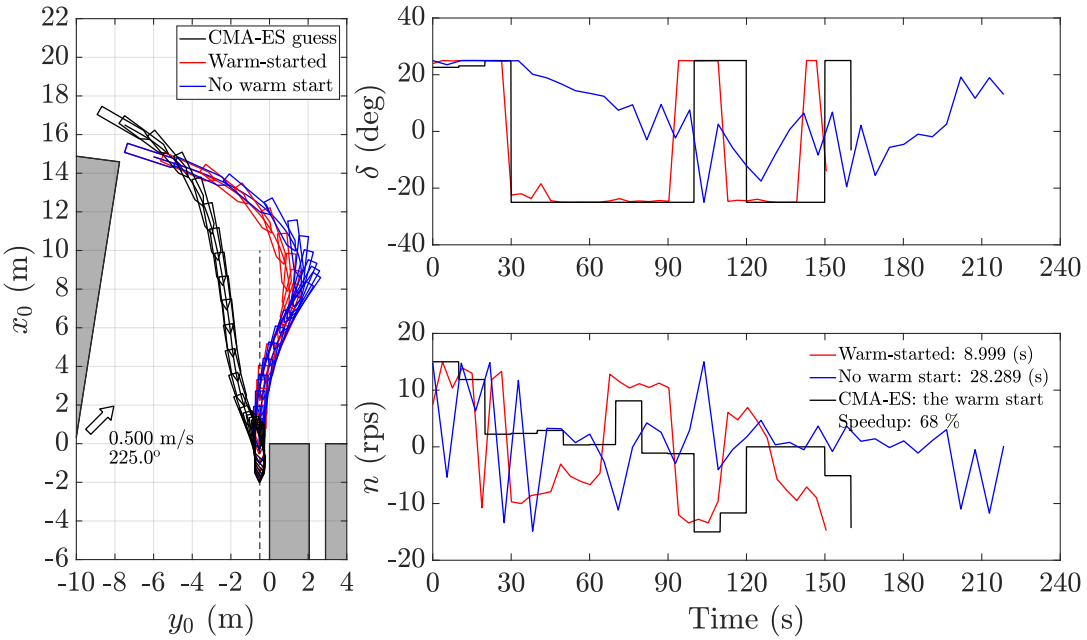


Figure 3-9. Visualization and optimal control for case M3 with computational time of 8.999 s.

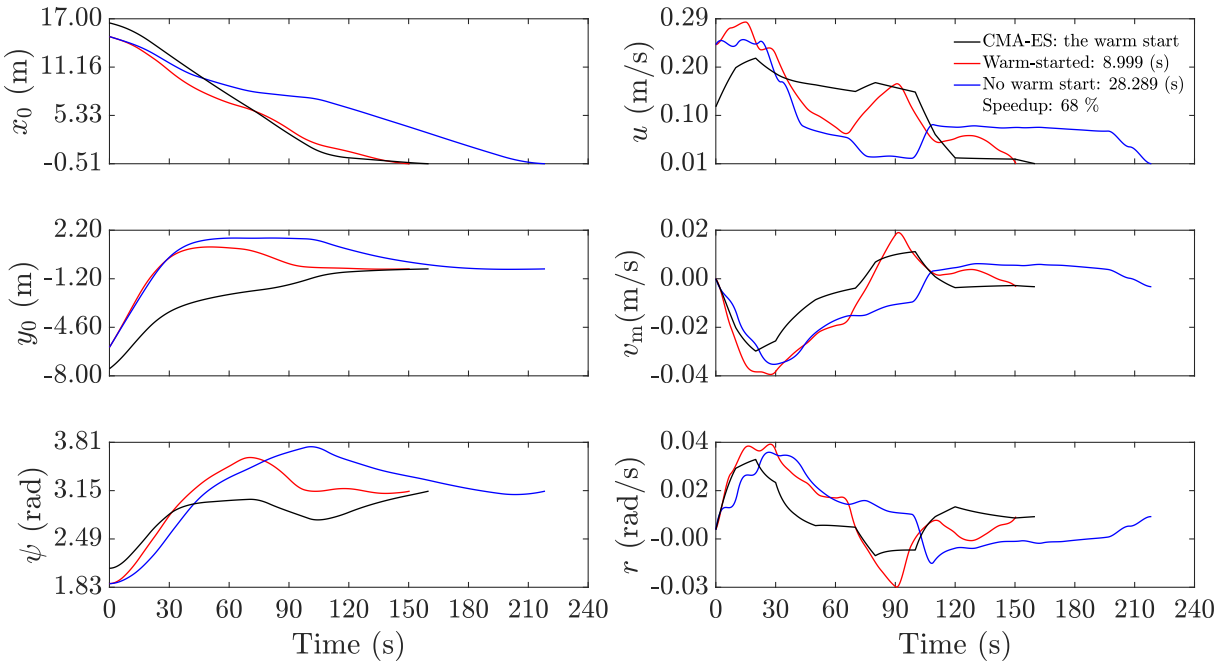


Figure 3-10. Summary of optimal state trajectory for case M3 with computational time of 8.999 s.

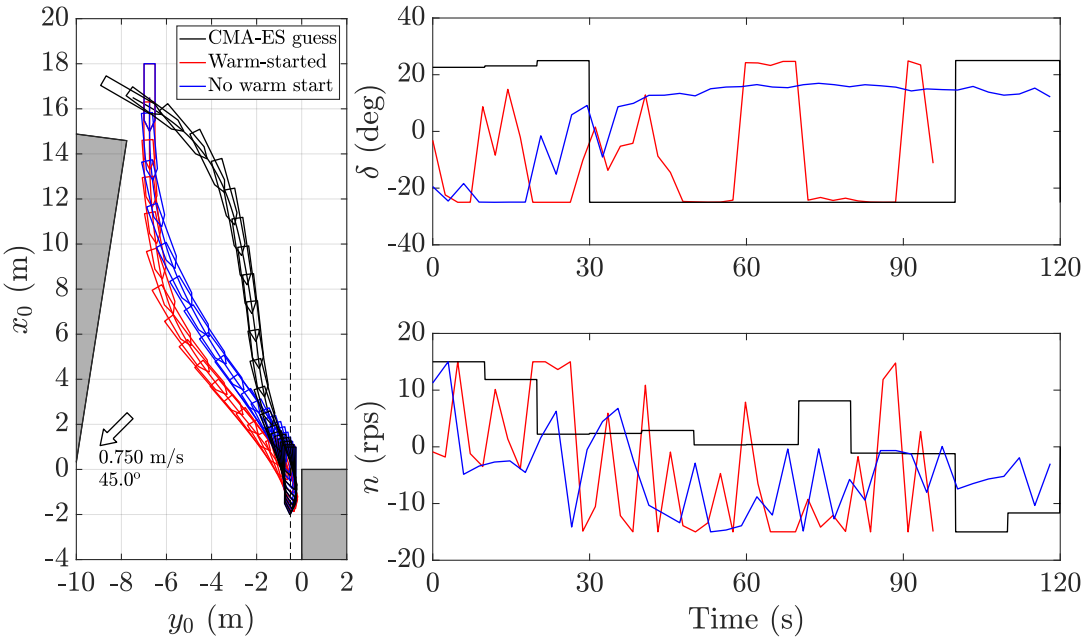


Figure 3-11. Visualization and optimal control for case M4 with computational time of 11.664 s.

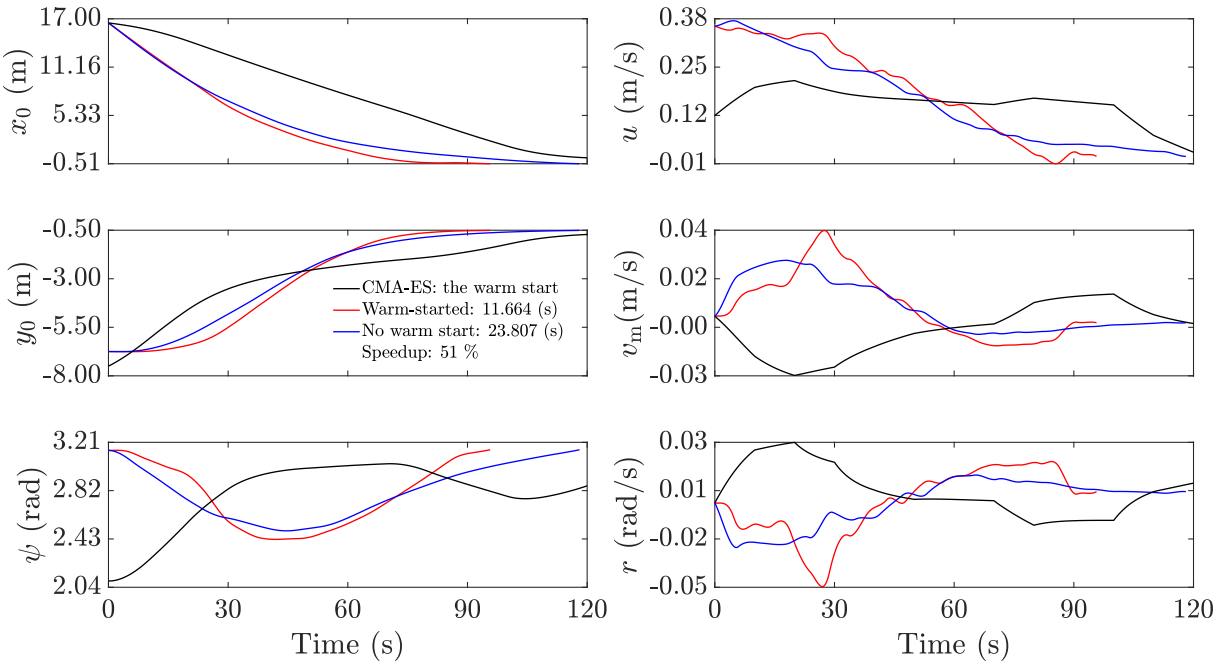


Figure 3-12. Summary of optimal state trajectory for case M4 with computational time of 11.664 s.

3.4.5 Additional Cases

The previous subsections discussed several cases in adherence to evaluation point 1, that is, the applicability of how just one almost-globally optimal solution from the Off-TP to warm-start the SO-TP for different initial states and condition.

This subsection extends the evaluation to more cases. Here, ten additional cases: A1 to A10, were considered. The norm of the initial states L_2 varies between 0 to 2.3 as shown in table 3-3. Note that these cases were chosen arbitrarily to represent the unpredictable nature of an actual docking operation. In other words, these cases cover all (as representatives of) the possible situations in the Inukai experiment pond. For each of these cases, the SO-TP uses the same warm start used in the previous cases. Case A9 and A10 are the cases where the ship executes the docking operation from the East, or from the positive y_0-axis. For the rest of the cases: case M1 to A8, the ship proceeds with the operation from the West. A compact visualization of the optimal trajectories for all of the cases is presented in figures 3-13 to 3-15.

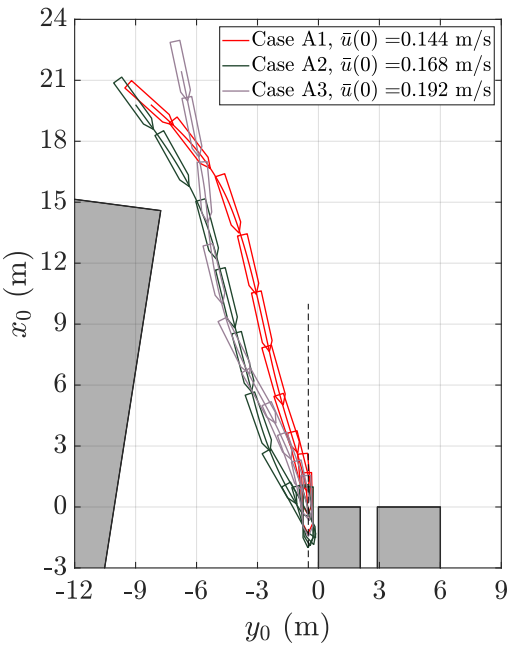


Figure 3-13. Visualizations of the optimal trajectory for case A1, A2, and A3. See tables 3-3 and 3-4 for a more detailed comparison.

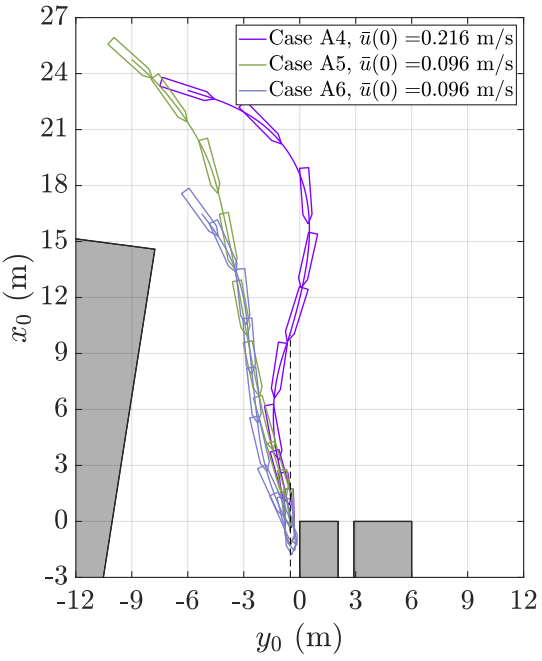


Figure 3-14. Visualizations of the optimal trajectory for case A4, A5, and A6.

From table 3-3, one can see that the warm start obtained by the Off-TP gave a shorter computational time with an average of 9.211 s (see also figure 3-16). This is comparably faster than that without a warm start where the average is 34.292 s. On average, the warm start was able to speed up the computational time up to 71 %. Moreover, in almost all cases the warm-started SO-TP gave a shorter t_f, which is indeed desirable.

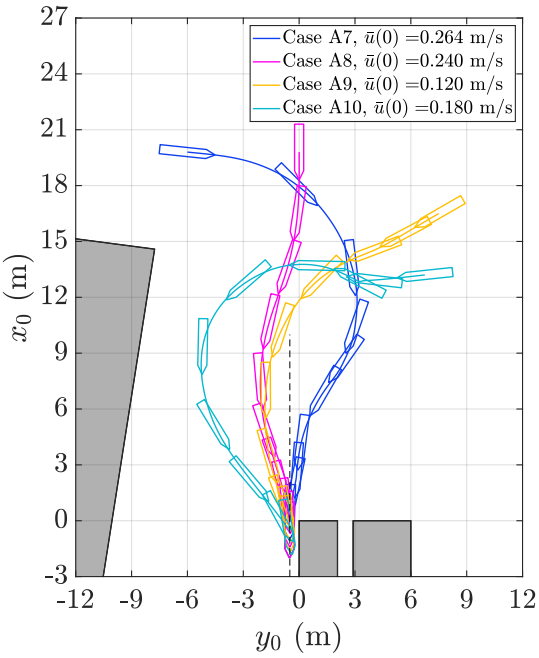


Figure 3-15. Visualizations of the optimal trajectory for case A7, A8, A9, and A10. See tables 3-3 and 3-4 for a more detailed comparison.

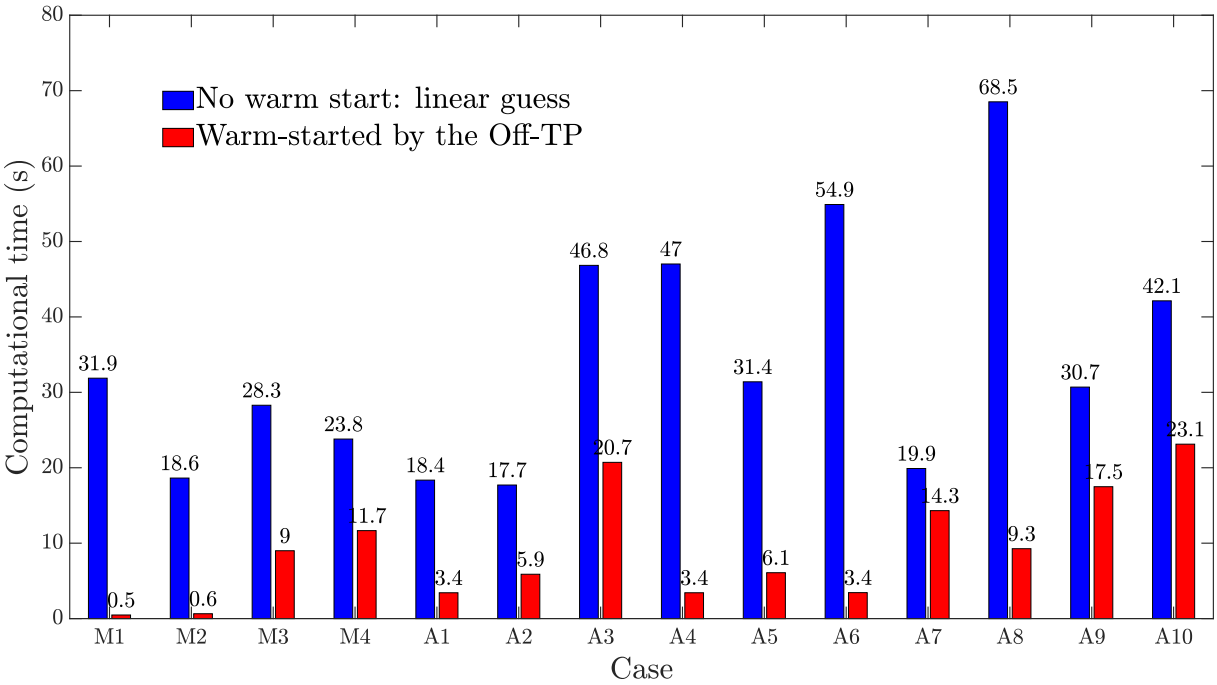


Figure 3-16. Comparison of the computational time between the SO-TP with a warm start and without a warm start.

Table 3-3. Comparison of SO-TP: with a warm start from the Off-TP and without a warm start (16 GB RAM, 8-core processor laptop).

Case	Norm $L_2 = \sqrt{\sum_{i=1}^{6}(d_i-1)^2}$	$U_{\mathrm{T}}\left(\bar{t}_j\right)$ (m/s)	$\chi\left(\bar{t}_j\right)$ (deg)	Warm start by Off-TP		No warm start (linear)		% Comp. speedup $\frac{T_{\mathrm{cs}}-T_{\mathrm{w1}}}{T_{\mathrm{cs}}}\times 100\%$
				t_{f} (s)	Comp. time T_{w1} (s)	t_{f} (s)	Comp. time T_{cs} (s)	
M1	0.00	No wind		162.0	0.478	233.3	31.877	98%
M2	1.03	0.75	45	136.2	0.645	156.0	18.635	97%
M3	1.03	0.50	225	150.5	8.999	218.4	28.289	68%
M4	2.06	0.75	45	95.7	11.664	118.1	23.807	51%
A1	0.32	0.75	45	151.9	3.425	177.1	18.361	81%
A2	0.53	0.75	180	163.1	5.881	231.8	17.711	67%
A3	0.79	0.75	125	146.5	20.718	166.1	46.836	56%
A4	0.92	0.50	135	167.7	3.426	289.4	47.019	93%
A5	0.58	0.50	90	175.5	6.084	217.5	31.397	81%
A6	0.41	0.50	315	159.6	3.442	256.1	54.906	94%
A7	1.25	0.50	250	207.0	14.313	205.9	19.903	28%
A8	1.51	0.50	90	137.4	9.268	159.7	68.519	86%
A9	2.24	0.75	0	180.5	17.487	214.5	30.697	43%
A10	2.31	0.50	45	203.7	23.128	192.7	42.133	45%
				AVG	9.211	AVG	34.292	71%

Table 3-4. Comparison of the SO-TP: warm-started by the Off-TP and warm-started by the cold-started SO-TP (with no warm start/linear guess) for case M1.

Case	Warm start by Off-TP				Warm start by SO-TP (M1, no warm start)				Preferred warm start (based on comp. time)
	t_f (s)	Feasible?	Constraint violation	Comp. time T_{w1} (s)	t_f (s)	Feasible?	Constraint violation	Comp. time (s)	
M2	136.2	Yes	2.5×10^{-11}	0.645	–	No	2.1×10^{-3}	77.111	Off-TP
M3	150.5	Yes	2.5×10^{-11}	8.999	230.7	Yes	1.3×10^{-11}	6.094	SO-TP
M4	95.7	Yes	1.7×10^{-11}	11.664	–	No	8.3×10^{-3}	110.403	Off-TP
A1	151.9	Yes	4.5×10^{-12}	3.425	172.1	Yes	6.7×10^{-13}	17.657	Off-TP
A2	163.1	Yes	2.2×10^{-11}	5.881	245.3	Yes	5.4×10^{-13}	12.682	Off-TP
A3	146.5	Yes	1.4×10^{-11}	20.718	–	No	2.2×10^{-3}	161.069	Off-TP
A4	167.7	Yes	2.0×10^{-11}	3.426	–	No	3.1×10^{-3}	37.161	Off-TP
A5	175.5	Yes	2.4×10^{-12}	6.084	–	No	1.3×10^{-3}	66.414	Off-TP
A6	159.6	Yes	5.2×10^{-12}	3.442	–	No	3.2×10^{-3}	47.519	Off-TP
A7	207.0	Yes	9.7×10^{-11}	14.313	248.4	Yes	1.2×10^{-11}	27.535	Off-TP
A8	137.4	Yes	5.9×10^{-12}	9.268	152.1	Yes	1.1×10^{-11}	297.179	Off-TP
A9	180.5	Yes	9.7×10^{-14}	17.487	210.1	Yes	3.3×10^{-11}	44.545	Off-TP
A10	203.7	Yes	4.0×10^{-12}	23.128	325.5	Yes	6.4×10^{-11}	29.416	Off-TP

3.4.6 The Significance of the Off-TP in the Scheme

It has been demonstrated that the warm start from the Off-TP can reduce the computational time of the SO-TP in many cases. One might ask, what will happen if we use the result from cold-started SO-TP as a warm start? This is exactly what evaluation point 2 tries to answer. Suppose that one has a locally optimal solution obtained by the SO-TP for case M1, without a warm start (use linear guess). This locally optimal solution is shown in blue curves in figures 3-5 and 3-6. One can use this solution from the SO-TP to warm-start the rest of the cases (M2 to A10).

In addition to the computational time and the t_{f} (like the previous subsection), the feasibility status is also used to compare the results of the two warm start methods: by the Off-TP and cold-started SO-TP. This feasibility is determined by how much the optimization variables violate the given constraints shown in equations (3.3–17b) to (3.3–17f). Feasibility is important because some gradient-based solvers can converge to a solution that violates the constraints: infeasible. Here, the infeasibility of a solution is reported if there is at least one optimization variable that violates one or more constraints larger than the given tolerance value: 1×10^{-6}.

The comparison between the two warm-start methods is summarized in table 3-4. It is directly noticeable that the cold-started SO-TP (no warm start) resulted in almost half of the cases being infeasible: constraint violation larger than the default tolerance value. Moreover, computational speedups due to the warm start from the Off-TP are observed in all the cases, except case M3. It is also demonstrated that one should not use the solution from the SO-TP without a warm start as an initial guess for other problems. This solution may not be able to direct the search to a feasible solution because of the nonlinearity and difficulty of the NLP (escalated by the underactuation).

To maximize the potential of the SO-TP (or any scheme that uses a gradient-based solver), a good initial guess is necessary. It has a significant effect on the computational time required to solve the given trajectory optimization. Such an initial guess can be generated by the Off-TP whose applicability to different problems has been demonstrated. To conclude this section, the SO-TP and the Off-TP should be implemented in a scheme that allows each of them to exploit their respective advantages to compromise the disadvantages of the other (see figure 3-1).

3.5 Concluding Remark

This chapter introduces a scheme that uses two different trajectory plannings where the OCP in each planner is transcribed and solved differently. The first trajectory planner is called Off-TP (offline trajectory planner) in which the corresponding NLP is solved by a gradient-free stochastic/heuristic algorithm known as CMA-ES (covariance matrix adaptation evolution strategy). The planner is termed *offline* because it requires a really significant computational time to generate the trajectory. However, the generated solution is almost globally optimal. This planner is a high-level trajectory planner because it takes into account only the static conditions of the environment.

The next step is to use this solution to warm-start the NLP in the second trajectory planner which is called semionline trajectory planner (SO-TP). This SO-TP is applicable to conventional underactuated ship, is able to guarantee that any feasible trajectory is

safe from allision, and is able to take into account the dynamics of the wind to some extent. The warm start by the offline solution from the Off-TP proved to increase the efficiency of the warm-started SO-TP. This is evaluated and quantified by the significant reduction in the computational time of the SO-TP. Moreover, the versatility of only one offline solution from the Off-TP to warm-start the SO-TP in many different cases has also been demonstrated. As has been shown in table 3-3, the average of the computational time required by the warm-started SO-TP is less than 30 seconds, which is relatively fast for a full-scale ship. This is why, along with the ability to take into account the wind dynamics, the term *semionline* is used to indicate the nature of the computational time that is not strictly in real time (online).

Note that because the SO-TP considers only the geometry of the harbor it can be regarded as a high-level trajectory planner. However, it also modifies the trajectory based on the instantaneous conditions of the wind and the ship's states which is the purpose of a middle-level trajectory planner. This results in the SO-TP being in the intermediate level between a high-level planner and a middle-level planner.

Even though a real-time performance has not been achieved yet, this study can be considered as a contribution to realizing an online trajectory planner for conventional underactuated ships. More importantly, this study is a significant improvement to the previous studies shown in [10, 14] where computational time was the biggest disadvantage of the proposed methods. Moreover, the ability of the SO-TP to enforce stringent constraints on the states is also one of the biggest merits of the SO-TP that the methods in [10, 14] do not possess.

Furthermore, the warm-started SO-TP can be implemented in a scenario that is relevant to the idea of MASS, in particular **AS degree 1**. It can be used by a shipmaster as a support system for a docking operation. For instance, the shipmaster can use the generated trajectory as a reference when planning a trajectory or when maneuvering the ship to dock (by tracking the generated trajectory). Another example is, the feasibility status that is reported by the SO-TP can be used as a qualitative measure to decide whether a docking operation can be executed or not. In this sense, when the SO-TP fails to find a feasible solution, within the accuracy of the mathematical model of the ship, the shipmaster can conclude that the ship is unable to reach the desired final docking states by itself, given the initial states, the geometric constraints, and the wind dynamics.

For reference, the preliminary study of this scheme had been presented online at the annual meeting of the Japan Society of Naval Architects and Ocean Engineers (Autumn meeting, 2021). It is a part of the published proceeding: **publication [D]**. The final development of the SO-TP with significant improvements in terms of computational time and allision constraints is published in an academic journal as **publication [B]**. These two studies are part of **contribution I** to develop a practical and fast trajectory planner that is much faster than those introduced by the author's colleagues.

Appendices

III.A Hermite Interpolation Polynomial: Coefficients

The interpolation polynomial $\mathbf{x}_{\mathrm{H}}$ in equation (3.3–23) and its derivative in equation (3.3–3) at the start of the segment, i.e., $t = t_k$, are as follows:

$$\mathbf{x}_{\mathrm{H}}(t_k) = \mathbf{a}_0 \tag{III.A–1a}$$

$$\mathbf{f}_{\mathrm{H}}(t_k) = \frac{1}{h_{\mathrm{s}}}\mathbf{a}_1. \tag{III.A–1b}$$

At the end of the segment, i.e., $t = t_{k+1}$ we have,

$$\mathbf{x}_{\mathrm{H}}(t_{k+1}) = \mathbf{a}_0 + \mathbf{a}_1 + \mathbf{a}_2 + \mathbf{a}_3 \tag{III.A–2a}$$

$$\mathbf{f}_{\mathrm{H}}(t_{k+1}) = \frac{1}{h_{\mathrm{s}}}\mathbf{a}_1 + \frac{2}{h_{\mathrm{s}}}\mathbf{a}_2 + \frac{3}{h_{\mathrm{s}}}\mathbf{a}_3. \tag{III.A–2b}$$

Equating the above equations with the conditions shown in equation (3.3–5) and putting them all into a linear transformation, one can have the following expression:

$$\begin{bmatrix} \mathbf{x}_k \\ \mathbf{f}_k \\ \mathbf{x}_{k+1} \\ \mathbf{f}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1/h_{\mathrm{s}} & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1/h_{\mathrm{s}} & 2/h_{\mathrm{s}} & 3/h_{\mathrm{s}} \end{bmatrix} \begin{bmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{bmatrix}. \tag{III.A–3}$$

To find the coefficients, one can inverse the transformation analytically or perform substitution. It is straightforward that

$$\mathbf{a}_0 = \mathbf{x}_k \tag{III.A–4a}$$

$$\mathbf{a}_1 = h_{\mathrm{s}}\mathbf{f}_k. \tag{III.A–4b}$$

Substituting the above to the last two rows of equation (III.A–3) gives us the following equations:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h_{\mathrm{s}}\mathbf{f}_k + \quad \mathbf{a}_2 + \quad \mathbf{a}_3 \tag{III.A–5a}$$

$$\mathbf{f}_{k+1} = \qquad \mathbf{f}_k + \frac{2}{h_{\mathrm{s}}}\mathbf{a}_2 + \frac{3}{h_{\mathrm{s}}}\mathbf{a}_3. \tag{III.A–5b}$$

Multiplying equation (III.A–5b) with h_{s} and subtract it from equation (III.A–5a) results in the following equations:

$$\mathbf{x}_{k+1} - h_{\mathrm{s}}\mathbf{f}_{k+1} = \mathbf{x}_k - \mathbf{a}_2 - 2\mathbf{a}_3$$

$$\mathbf{a}_3 = \frac{\mathbf{x}_k - \mathbf{a}_2 - \mathbf{x}_{k+1} + h_{\mathrm{s}}\mathbf{f}_{k+1}}{2}. \tag{III.A–6}$$

Substituting $\mathbf{a}_3$ from the above equation to equation (III.A–5b) gives us the expression for $\mathbf{a}_2$ as follows:

$$\begin{aligned}
\mathbf{f}_{k+1} &= \mathbf{f}_k + \frac{2}{h_\mathrm{s}}\mathbf{a}_2 - \frac{3}{2h_\mathrm{s}}\mathbf{a}_2 + \frac{3}{2h_\mathrm{s}}\mathbf{x}_k - \frac{3}{2h_\mathrm{s}}\mathbf{x}_{k+1} + \frac{3}{2}\mathbf{f}_{k+1} \\
\mathbf{0} &= \mathbf{f}_k + \left(\frac{4-3}{2h_\mathrm{s}}\right)\mathbf{a}_2 \quad + \frac{3}{2h_\mathrm{s}}\mathbf{x}_k - \frac{3}{2h_\mathrm{s}}\mathbf{x}_{k+1} + \left(\frac{3-2}{2}\right)\mathbf{f}_{k+1} \\
-\frac{1}{2h_\mathrm{s}}\mathbf{a}_2 &= \mathbf{f}_k \qquad\qquad + \frac{3}{2h_\mathrm{s}}\mathbf{x}_k - \frac{3}{2h_\mathrm{s}}\mathbf{x}_{k+1} + \frac{1}{2}\mathbf{f}_{k+1}.
\end{aligned}$$

Multiplying both sides with $2h_\mathrm{s}$ we have

$$-\mathbf{a}_2 = 2h_\mathrm{s}\mathbf{f}_k + 3\mathbf{x}_k - 3\mathbf{x}_{k+1} + h_\mathrm{s}\mathbf{f}_{k+1},$$

which yields,

$$\mathbf{a}_2 = -3\mathbf{x}_k - 2h_\mathrm{s}\mathbf{f}_k + 3\mathbf{x}_{k+1} - h_\mathrm{s}\mathbf{f}_{k+1}. \tag{III.A–7}$$

Substituting $\mathbf{a}_2$ from the above back to equation (III.A–6) we have,

$$\begin{aligned}
\mathbf{a}_3 &= \frac{1}{2}\mathbf{x}_k - \frac{1}{2}\mathbf{a}_2 - \frac{1}{2}\mathbf{x}_{k+1} + \frac{h_\mathrm{s}}{2}\mathbf{f}_{k+1} \\
\mathbf{a}_3 &= \frac{1}{2}\mathbf{x}_k - \frac{1}{2}\left(-3\mathbf{x}_k - 2h_\mathrm{s}\mathbf{f}_k + 3\mathbf{x}_{k+1} - h_\mathrm{s}\mathbf{f}_{k+1}\right) - \frac{1}{2}\mathbf{x}_{k+1} + \frac{h_\mathrm{s}}{2}\mathbf{f}_{k+1} \\
\mathbf{a}_3 &= \frac{1}{2}\mathbf{x}_k + \frac{3}{2}\mathbf{x}_k + h_\mathrm{s}\mathbf{f}_k - \frac{3}{2}\mathbf{x}_{k+1} + \frac{h_\mathrm{s}}{2}\mathbf{f}_{k+1} \qquad - \frac{1}{2}\mathbf{x}_{k+1} + \frac{h_\mathrm{s}}{2}\mathbf{f}_{k+1}.
\end{aligned}$$

Simplifying the above,

$$\mathbf{a}_3 = \left(\frac{1+3}{2}\right)\mathbf{x}_k + h_\mathrm{s}\mathbf{f}_k - \left(\frac{3+1}{2}\right)\mathbf{x}_{k+1} + \left(\frac{h_\mathrm{s}+h_\mathrm{s}}{2}\right)\mathbf{f}_{k+1},$$

yields the expression for $\mathbf{a}_3$ as follows:

$$\mathbf{a}_3 = 2\mathbf{x}_k + h_\mathrm{s}\mathbf{f}_k - 2\mathbf{x}_{k+1} + h_\mathrm{s}\mathbf{f}_{k+1}. \tag{III.A–8}$$

Finally, one can put all the expressions of the coefficients from equations (III.A–4a), (III.A–4b), (III.A–7) and (III.A–8) into a linear transformation as follows:

$$\begin{bmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & h_\mathrm{s} & 0 & 0 \\ -3 & -2h_\mathrm{s} & 3 & -h_\mathrm{s} \\ 2 & h_\mathrm{s} & -2 & h_\mathrm{s} \end{bmatrix} \begin{bmatrix} \mathbf{x}_k \\ \mathbf{f}_k \\ \mathbf{x}_{k+1} \\ \mathbf{f}_{k+1} \end{bmatrix},$$

which is exactly equal to equation (3.3–6).

III.B Interpolation at the Mid-segment Point

One can denote the mid-segment point as $t_{k+0.5} = t_k + h_s/2$. Substituting $t = t_k + h_s/2$ into the interpolation polynomial $\mathbf{x}_H$ shown in equation (3.3–2), one can have the following:

$$\begin{aligned}\mathbf{x}_H(t_{k+0.5}) = \mathbf{x}_k + \frac{h_s}{2}\mathbf{f}_k - \frac{3}{4}\mathbf{x}_k - \frac{2h_s}{4}\mathbf{f}_k + \frac{3}{4}\mathbf{x}_{k+1} - \frac{h_s}{4}\mathbf{f}_{k+1} \\ + \frac{2}{8}\mathbf{x}_k + \frac{h_s}{8}\mathbf{f}_k - \frac{2}{8}\mathbf{x}_{k+1} + \frac{h_s}{8}\mathbf{f}_{k+1}.\end{aligned}$$

Simplifying the above expression,

$$\begin{aligned}\mathbf{x}_H(t_{k+0.5}) &= \left(\frac{8-6+2}{8}\right)\mathbf{x}_k + \left(\frac{4-4+1}{8}\right)h_s\mathbf{f}_k + \left(\frac{6-2}{8}\right)\mathbf{x}_{k+1} + \left(\frac{-2+1}{8}\right)h_s\mathbf{f}_{k+1} \\ &= \frac{1}{2}\mathbf{x}_k + \frac{1}{8}h_s\mathbf{f}_k + \frac{1}{2}\mathbf{x}_{k+1} - \frac{1}{8}h_s\mathbf{f}_{k+1},\end{aligned}$$

yields an expression as follows:

$$\mathbf{x}_H(t_{k+0.5}) = \frac{1}{2}(\mathbf{x}_k + \mathbf{x}_{k+1}) + \frac{h_s}{8}(\mathbf{f}_k - \mathbf{f}_{k+1}). \tag{III.B–1}$$

We expressed everything in terms of k or knot points. Now we shall express k in terms of s or segments. The time at the mid-segment point is denoted by t_{2s}. Thus, one can obtain the equivalent expressions for k as follows:

$$\begin{aligned}t_k &= t_{2s-1} \\ t_{k+0.5} &= t_{2s} \\ t_{k+1} &= t_{2s+1}.\end{aligned}$$

With this, we have the same expression as equation (3.3–11) as follows:

$$\mathbf{x}_H(t_{2s}) = \frac{1}{2}(\mathbf{x}_{2s-1} + \mathbf{x}_{2s+1}) + \frac{h_s}{8}(\mathbf{f}_{2s-1} - \mathbf{f}_{2s+1}).$$

Doing the same: substituting $t = t_k + h_s/2$ into the time-derivative of $\mathbf{x}_H$ shown in equation (3.3–3), one can have the following:

$$\mathbf{f}_H(t_{k+0.5}) = \mathbf{f}_k + \frac{2}{2h_s}(-3\mathbf{x}_k - 2h_s\mathbf{f}_k + 3\mathbf{x}_{k+1} - h_s\mathbf{f}_{k+1}) + \frac{3}{4h_s}(2\mathbf{x}_k + h_s\mathbf{f}_k - 2\mathbf{x}_{k+1} + h_s\mathbf{f}_{k+1})$$

$$\begin{aligned}\mathbf{f}_H(t_{k+0.5}) = \mathbf{f}_k - \frac{3}{h_s}\mathbf{x}_k - 2\mathbf{f}_k + \frac{3}{h_s}\mathbf{x}_{k+1} - \mathbf{f}_{k+1} \\ + \frac{3}{2h_s}\mathbf{x}_k + \frac{3}{4}\mathbf{f}_k - \frac{3}{2h_s}\mathbf{x}_{k+1} + \frac{3}{4}\mathbf{f}_{k+1}.\end{aligned}$$

Simplifying the above, one can get

$$\begin{aligned}\mathbf{f}_H(t_{k+0.5}) &= \left(\frac{-6+3}{2h_s}\right)\mathbf{x}_k + \left(\frac{6-3}{2h_s}\right)\mathbf{x}_{k+1} + \left(\frac{4-8+3}{4}\right)\mathbf{f}_k + \left(\frac{-4+3}{4}\right)\mathbf{f}_{k+1} \\ &= -\frac{3}{2h_s}\mathbf{x}_k + \frac{3}{2h_s}\mathbf{x}_{k+1} - \frac{1}{4}\mathbf{f}_k - \frac{1}{4}\mathbf{f}_{k+1},\end{aligned}$$

which yields,

$$\mathbf{f}_{\mathrm{H}}\left(t_{k+0.5}\right) = -\frac{3}{2h_{\mathrm{s}}}\left(\mathbf{x}_k - \mathbf{x}_{k+1}\right) - \frac{1}{4}\left(\mathbf{f}_k + \mathbf{f}_{k+1}\right).$$

In terms of s (segment), the above expression is equivalent to equation (3.3–12) as follows:

$$\mathbf{f}_{\mathrm{H}}\left(t_{2s}\right) = -\frac{3}{2h_{\mathrm{s}}}\left(\mathbf{x}_{2s-1} - \mathbf{x}_{2s+1}\right) - \frac{1}{4}\left(\mathbf{f}_{2s-1} + \mathbf{f}_{2s+1}\right).$$

III.C Simpson Quadrature Constraints: Implicit Integration

Suppose at the mid-segment point, an optimization variable $\mathbf{x}_{2s}$ satisfies the Hermite interpolation constraint (3.3–13). With this $\mathbf{x}_{2s}$ and $\mathbf{u}_{2s}$, one can of course compute the corresponding predicted dynamics $\mathbf{f}_{2s}$ with equation (3.3–1b). Let us equate this $\mathbf{f}_{2s}$ with the time-derivative of the interpolation polynomial for the states at the mid-segment point, i.e., $\mathbf{f}_{\mathrm{H}}\left(t_{2s}\right)$ as follows (the Simpson quadrature constraint):

$$\begin{aligned}
\mathbf{f}_{2s} &= \mathbf{f}_{\mathrm{H}}\left(t_{2s}\right) \\
\mathbf{f}_{2s} &= -\frac{3}{2h_{\mathrm{s}}}\left(\mathbf{x}_{2s-1} - \mathbf{x}_{2s+1}\right) - \frac{1}{4}\left(\mathbf{f}_{2s-1} + \mathbf{f}_{2s+1}\right) \\
\mathbf{0} &= \frac{3}{2h_{\mathrm{s}}}\left(\mathbf{x}_{2s+1} - \mathbf{x}_{2s-1}\right) - \frac{1}{4}\left(\mathbf{f}_{2s-1} + 4\mathbf{f}_{2s} + \mathbf{f}_{2s+1}\right).
\end{aligned}$$

Multiplying both sides with $2h_{\mathrm{s}}/3$ one obtains the following:

$$\mathbf{0} = \left(\mathbf{x}_{2s+1} - \mathbf{x}_{2s-1}\right) - \frac{h_{\mathrm{s}}}{6}\left(\mathbf{f}_{2s-1} + 4\mathbf{f}_{2s} + \mathbf{f}_{2s+1}\right),$$

which, upon rearranging, yields the Simpsons quadrature as shown in equation (3.3–9) as follows:

$$\mathbf{x}_{2s+1} - \mathbf{x}_{2s-1} = \frac{h_{\mathrm{s}}}{6}\left(\mathbf{f}_{2s-1} + 4\mathbf{f}_{2s} + \mathbf{f}_{2s+1}\right),$$

as required. If $\mathbf{x}_{2s}$ satisfies the Hermite interpolation constraint (3.3–13) and together with $\mathbf{u}_{2s}$ also satisfies the Simpson quadrature (3.3–9), then it means we have performed an implicit integration of the predicted dynamics within the segment.

大阪大学 OSAKA UNIVERSITY

IV

Trajectory Tracking and Path Following Controller with Evolution Strategy for Docking Operation

4.1 Introduction

As explained in the general introduction (section 1.1.2), an autonomous ship must be capable of maneuvering and controlling itself based on the planned trajectory/path. The low-speed criterion of a docking operation requires that the propeller revolution number be kept as minimal as possible. This effectively reduces the steering force and the overall maneuverability of the ship. Moreover, sometimes it is required to change the propeller's mode (reverse/mode), which results in a very nonlinear and intricate mathematical model as explained in chapter II.

This complicated mathematical model greatly limits the mathematical tools that can be implemented when designing the controller to maneuver the ship to the final docking point. Several studies have implemented controllers for nonlinear dynamical systems like backstepping for conventional underactuated ships in a setpoint stabilization operation [35, 36], a general trajectory tracking operation [42], and even a crabbing maneuver [45]. Backstepping also saw implementation in automating a docking operation of a fully actuated ship [20]. This backstepping method is based on the recursive design of control-Lyapunov functions which proves its worth in stabilizing nonlinear systems as reported in the previously mentioned works. However, one must go through the extensive process from recursively choosing the Lyapunov function candidates to analytically deriving the control law that stabilizes the system. Moreover, a highly nonlinear dynamical system like the one described in chapter II is nonsmooth and may have discontinuities that can make everything more complicated when designing the controller; not to mention the additional step to incorporate the constraints on the control. Because of these limitations, when handling such a highly nonlinear dynamical system, one can always opt to design an NMPC (nonlinear model predictive control).

The idea of NMPC is to solve multiple trajectory optimizations [31] like explained in chapter III, regularly at a certain period, in a closed-loop system. Closed-loop means the controller takes into account the instantaneous states of the ship when formulating a new trajectory optimization problem[14]. Several studies have reported its implementation in a setpoint stabilization and a trajectory tracking operation for underactuated ships [65, 66],

[14]This is what the SO-TP also tries to achieve, but in a middle level rather than in a low level like a controller.

and automatic docking operations for underactuated ships [8, 11] and for a fully actuated ship [16, 18]. All showed very promising potentials both in simulations and experiments.

One of the most important aspects in NMPC is that the NLP (the trajectory optimization) must be solved fast enough: *online*. The most common choices of the solver are gradient-based solvers like IPM (interior point method) or SQP (sequential quadratic programming). We have discussed and demonstrated in section 3.3 that a gradient-based method does require a good initial guess for it to be fast enough: a good initialization [67]. A not-so-complicated mathematical model and full actuation can greatly reduce the difficulty of the NLP in the NMPC, thus allowing the solver to solve the NLP in real time/online [8, 11, 16, 18]. However, this is not the case when the mathematical model is very nonlinear as described in chapter II.

This thesis attempts to investigate the real-time application of a gradient-free and global nonlinear optimization solver known as CMA-ES (covariance matrix adaptation evolution strategy) [46, 60] to solve the NLP in an NMPC for a docking operation of the conventional underactuated EOShip (see table 2-1). CMA-ES has proven itself in solving the Off-TP [10, 14] without any initial guesses. Its almost-globally optimal solution can also be used to warm-start the NLP in the SO-TP for many cases (chapter III). This thesis aims to extend the previous works with CMA-ES [10, 50] to the real-time application of CMA-ES as the algorithm to solve the NLP in an NMPC for trajectory tracking and path following. Note that here the terms NMPC and *controller* refer to the same entity.

4.2 Scopes

For a large commercial ship that has no side thrusters (conventional and underactuated), a *complete docking* operation can be divided into two stages: (1) *harbor-entrance* stage when the ship moves from an initial location to a location in the proximity of the final docking point and (2) *approach-to-berth* stage when the ship slows down and proceeds to the final docking point with external auxiliary devices (tugboats, etc.). Since the subject ship EOShip is not equipped with any side thrusters and there are no tugboats that are required for the approach-to-berth stage, the simulations are limited to the complete docking operation and the harbor-entrance operation. The former is a trajectory tracking operation and the latter is a path following operation.

4.2.1 Trajectory Tracking: Complete Docking Operation

In this operation, first a reference trajectory is obtained beforehand by the Off-TP [10] (see section 3.2). We shall use the same offline trajectory $\mathbf{x}^*_{\text{off}}$ that was used as the warm start in the SO-TP (see section 3.3.4). Then, an OCP for trajectory tracking is formulated. The objective is to track all the state trajectory, i.e., the ship has to be at the desired states at the scheduled time (temporal constraint exists). The OCP is then transcribed into a finite-dimensional NLP by following the transcription method used in the Off-TP: direct shooting method. Finally, this NLP is solved with CMA-ES where the optimization variables are the control command vector at the discretization point. This control command vector is expected to drive the ship to track all the reference states from the initial point until finally stops at the final docking point.

In this chapter, let us assume that the offline trajectory is the standard trajectory

for the particular harbor. Based on this standard trajectory, one can divide the whole complete docking operation into several periods such as *acceleration*, *main turn*, *deceleration*, and *stationing*. This division into substages is considered in the formulation of the OCP inside the controller.

4.2.2 Path Following: Harbor-entrance Operation

For this operation, one must modify the formulation of the trajectory optimization in the Off-TP. A location near the actual final docking point is chosen as the desired location where the ship does not need to completely stop but has to be parallel with the berth. Because the velocities are not the main concern, the propeller revolution number n is set as constant when generating the reference trajectory with the Off-TP. This reduces the optimization variables to just one propeller revolution number n, the final time, and the rudder angle δ at each discretization point (see equation (3.3–17)).

In this operation, the NMPC has to maintain the ship at the desired *pose*: path and yaw angle. The ship does not have to be at the desired states at the scheduled time. Thus, the goal of the OCP is to find the optimal rudder angle to maintain the ship at the desired pose with a constant propeller revolution number. This effectively simplifies the trajectory tracking problem into a path following problem that has no temporal constraint.

4.3 Formulation of the OCP for Motion Control

Let us use the same notations introduced in chapter III where t, $\mathbf{x}$, $\mathbf{u}$ denote the normalized prediction time, the predicted states, and the predicted control, inside the NMPC, respectively. Let $\bar{t}$ and $\bar{\mathbf{x}}$ denote the actual time and the actual states of the ship outside the NMPC, such that $\bar{t}_j$ and $\bar{\mathbf{x}}_j \triangleq \mathbf{x}\left(\bar{t}_j\right)$ denote the j-th sampling of the respective variables. Moreover, denote the reference trajectory from the Off-TP as $\mathbf{x}_{\text{ref}}$.

We will formulate an identical yet different OCP (trajectory optimization problem) from that formulated in the SO-TP (equation (3.3–1)). One distinction is, here the final time t_{f} is set as a constant, which is known as the *prediction horizon*, denoted by $\Delta\bar{t}$. This is equivalent to setting the final time in the OCP as $\Delta\bar{t}$. With this, the OCP for the motion control of a docking operation can be formulated as follows:

$$
\begin{aligned}
&\text{determine} && \mathbf{u}^*(t) = \begin{bmatrix} \delta^*(t) & n^*(t) \end{bmatrix}^\top \\
&\text{that min.} && J_{\text{mc}} = \sum_{i=1}^{6} w_i \, F_i^2\left(\bar{t}_j\right) \times \int_0^{\Delta\bar{t}} \sum_{i=1}^{6} G_i^2\left(\bar{t}_j\right) \, \mathrm{d}t, && \text{(4.3–1a)} \\
&\text{subject to} && \dot{\mathbf{x}}(t) = \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t)\right), && \text{(4.3–1b)} \\
& && \mathbf{x}^*(0) = \bar{\mathbf{x}}_j, && \text{(4.3–1c)} \\
& && \mathbf{x}^*\left(\Delta\bar{t}\right) = \mathbf{x}_{\text{ref}}\left(\bar{t}_j + \Delta\bar{t}\right), && \text{(4.3–1d)} \\
& && -\delta_{\max} \leq \delta(t) \leq \delta_{\max}, && \text{(4.3–1e)} \\
& && -n_{\max} \leq n(t) \leq n_{\max}, && \text{(4.3–1f)} \\
& && t \in \left[0,\, \Delta\bar{t}\right] \equiv t \in \left[\bar{t}_j,\, \bar{t}_j + \Delta\bar{t}\right],
\end{aligned}
$$

where in the objective function J_{mc}, the functions F_i and G_i are expressed as follows:

$$F_i\left(\bar{t}_j\right) = \frac{x_i\left(\Delta\bar{t}\right) - x_{\text{ref},i}\left(\bar{t}_j + \Delta\bar{t}\right)}{x_{\text{ref},i}\left(\bar{t}_j + \Delta\bar{t}\right)} \tag{4.3–2}$$

$$G_i\left(\bar{t}_j\right) = \frac{x_i\left(t\right) - x_{\text{ref},i}\left(\bar{t}_j + \Delta\bar{t}\right)}{x_{\text{ref},i}\left(\bar{t}_j + \Delta\bar{t}\right)}. \tag{4.3–3}$$

The function F_i is related to the terminal condition while function G_i is related to the accumulated deviation of the states from the desired reference states within the prediction horizon. In the objective function, these deviations are normalized by the reference states to balance the search space. Numerical treatment is also introduced to prevent singularity due to this normalization, that is, whenever $x_{\text{ref},i} = 0$, a value of 10^{-7} is used instead. Moreover, weight factors $w_i \in \mathbf{w}$ are introduced to give some sense of priority or difficulties to the states.

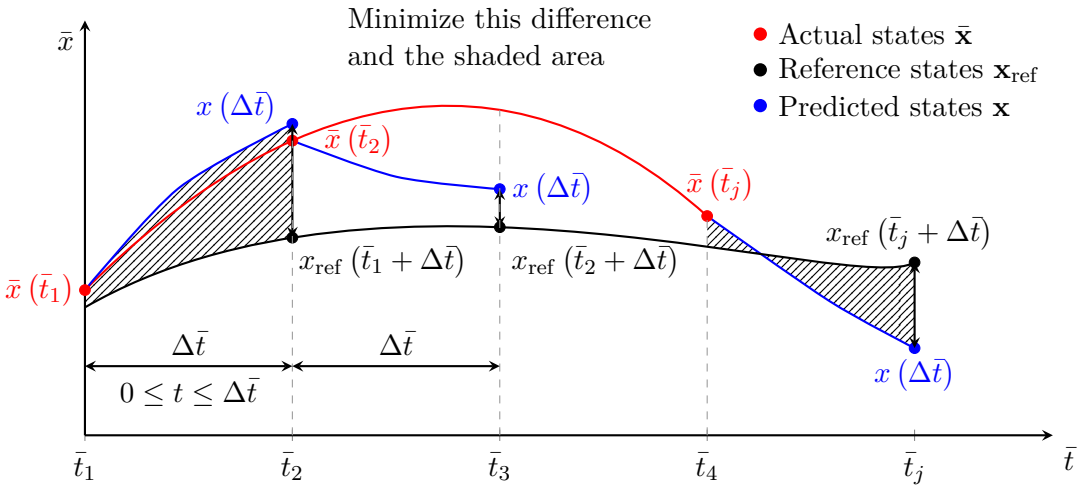


Figure 4-1. Objective of the OCP for motion control.

The objective function forces the optimization to find the optimal control command vector that will drive the ship such that the deviation/difference of the predicted states at the end of the prediction horizon $\mathbf{x}\left(\Delta\bar{t}\right)$ from the desired reference states is minimized. This is visualized in figure 4-1 where at every $\bar{t}_j$, the OCP is solved to find the optimal control that drives the ship from the initial states (shown in red) such that at $\bar{t}_j + \Delta\bar{t}$ the predicted states (shown in blue) is approximately equal to the reference states (shown in black). Here we let the sampling period of the actual states equal to the prediction horizon $\Delta\bar{t}$, that is, the OCP must be solved within $\Delta\bar{t}$.

4.4 Formulation of the NLP for Motion Control

The OCP formulated in equation (4.3–1) is transcribed into a finite-dimensional NLP by direct single shooting method where the optimal control command vector at the start of the prediction horizon is the only optimization variable. Since we let the prediction

大阪大学 OSAKA UNIVERSITY

horizon equal to the sampling period of the actual states, it is small enough that a single shooting method can be implemented. Consequently, one can assume that the predicted control is constant within $\Delta\bar{t}$, which is why it is different from the conventional NMPC. The general NLP that is to be solved by CMA-ES at each $\bar{t}_j$ in less than $\Delta\bar{t}$ second can be formulated as follows:

$$
\begin{aligned}
&\text{determine} && \mathbf{u}^*(0) = \begin{bmatrix} \delta^*(0) & n^*(0) \end{bmatrix}^\top \\
&\text{that min.} && J_{\mathrm{mc}} = \sum_{i=1}^{6} w_i F_i^2\left(\bar{t}_j\right) \times \int_0^{\Delta\bar{t}} \sum_{i=1}^{6} G_i^2\left(\bar{t}_j\right) \mathrm{d}t, && \text{(4.4–1a)} \\
&\text{subject to} && \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(0), \mathbf{u}(0)), && \text{(4.4–1b)} \\
& && \mathbf{x}(0) = \bar{\mathbf{x}}_j, && \text{(4.4–1c)} \\
& && \mathbf{x}\left(\Delta\bar{t}\right) = \mathbf{x}_{\mathrm{ref}}\left(\bar{t}_j + \Delta\bar{t}\right), && \text{(4.4–1d)} \\
& && -\delta_{\max} \leq \delta(0) \leq \delta_{\max}, && \text{(4.4–1e)} \\
& && -n_{\max} \leq n(0) \leq n_{\max}, && \text{(4.4–1f)} \\
& && t \in \left[0,\, \Delta\bar{t}\right] \equiv t \in \left[\bar{t}_j,\, \bar{t}_j + \Delta\bar{t}\right],
\end{aligned}
$$

As a note, the predicted dynamics $\mathbf{f}$ within the prediction horizon is integrated explicitly with the fourth-order Runge-Kutta method (RK4). The solver: CMA-ES tries to find the optimal control command vector that results in this integration being close to the desired reference states. Moreover, the integration in the objective function J_{mc} is approximated by the right Riemann sum.

4.5 Simulation: Complete Docking Operation

As mentioned in subsection 4.2.1, the same offline trajectory that had been used to warm-start the SO-TP was used as the reference trajectory in the simulation. It is assumed that this offline trajectory is a standard trajectory such that all ships in the harbor must follow the similar trajectory. This reference trajectory is shown in figure 4-2. Based on the surge velocity u, the reference trajectory can be divided into several periods: acceleration (shaded with green), main turning (shaded with purple), deceleration (shaded with cyan), and stationing (shaded with yellow). This is the logical sequence for a shipmaster to dock the ship in such a situation.

This division into several periods enables us to design some constraints on the control commands to attenuate their tendency to switch rapidly, despite the fact that EOShip is able to switch its actuators almost instantaneously. For the moment, the constraints are only imposed on the propeller revolution number n as it has a dominant effect on the overall maneuverability at a low speed. At each of the periods, a different constraint is introduced based on some of the state variables. In simple words, the idea is to slightly modify the OCP (and thus the NLP) based on some criteria.

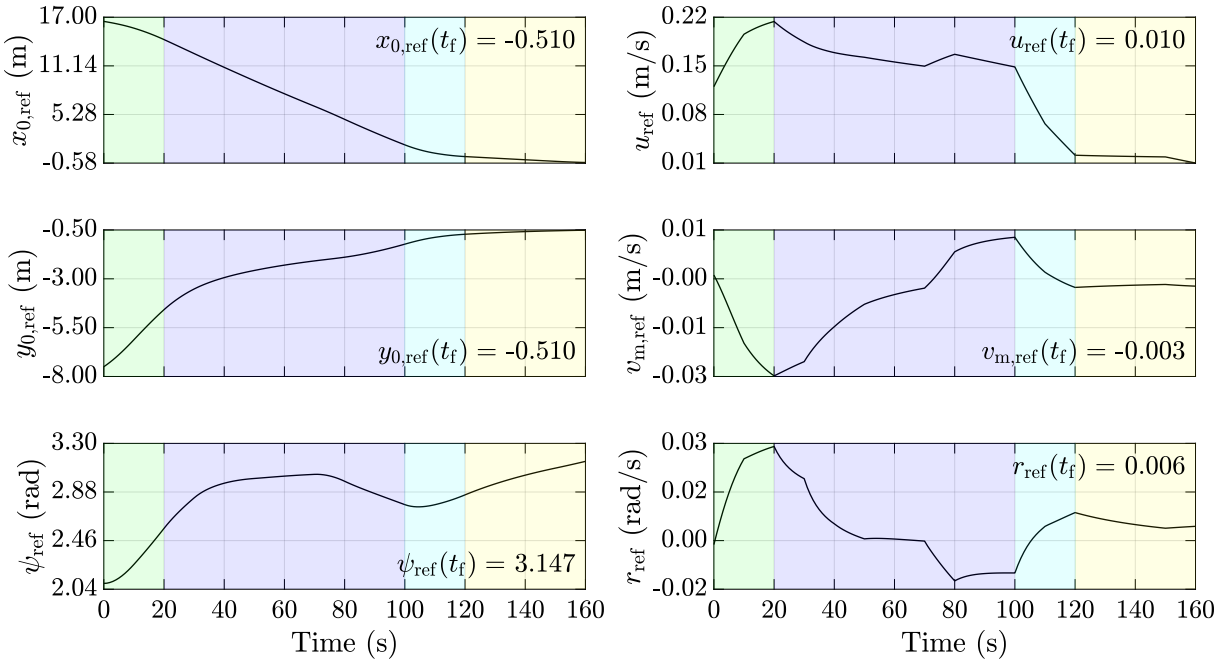


Figure 4-2. The reference trajectory in the complete docking operation.

Acceleration. The ship enters the harbor area at some low surge velocity. The ship has to accelerate such that the inflow velocity at the rudder is sufficient to generate the steering force required to do the main turn. This period is easily distinguishable by the steep increase in the reference surge velocity; shaded in green. In this period, to achieve such a steep acceleration, the propeller must always be in forward mode or at rest if the difference between the current and the previous reference surge velocity over $\Delta\bar{t}$ (acceleration) is not less than a threshold value $\dot{u}_\mathrm{acc}$. Instead of equation (4.4–1f), the following modified constraint on the propeller revolution number is imposed[15]:

$$0 \leq n\left(0\right) \leq n_\mathrm{max}, \quad \text{if} \quad \frac{u_\mathrm{ref}\left(\bar{t}_j + \Delta\bar{t}\right) - u_\mathrm{ref}\left(\bar{t}_j\right)}{\Delta\bar{t}} \geq \dot{u}_\mathrm{acc}. \tag{4.5–1}$$

Main turn. In this period, the general constraint shown in equation (4.4–1f) is active and not modified. This period is shaded in purple. In this period, the propeller is allowed to switch its mode rapidly.

Deceleration. Now the ship has to decelerate. The deceleration period is shaded in cyan. This period is characterized by a steep decrease in the reference surge velocity. To achieve this rapid deceleration, the propeller revolution number should always be in the reverse mode or at rest if the difference between the current and the previous reference surge velocity over $\Delta\bar{t}$ (acceleration) is not less than a threshold value $\dot{u}_\mathrm{dec}$. Moreover, such a restriction is only active only when the reference position is within the region where it is supposed to decelerate, i.e., x_dec. The modified constraint on the propeller revolution

[15]Do not get confused. The notation $n\left(0\right)$ is consistent with the fact that the time is normalized inside the NLP such that $t = 0$ is equivalent to $t = \bar{t}_j$.

number can be expressed as follows[16]:

$$-n_{\max} \leq n\left(0\right) \leq 0, \quad \text{if } \frac{u_{\text{ref}}\left(\bar{t}_j\right) - u_{\text{ref}}\left(\bar{t}_j + \Delta\bar{t}\right)}{\Delta\bar{t}} \geq \dot{u}_{\text{dec}} \text{ and } x_{0,\text{ref}}\left(\bar{t}_j + \Delta\bar{t}\right) \leq x_{\text{dec}}. \tag{4.5–2}$$

Stationing. This is the last period of the whole docking operation when the ship has to stabilize itself at a very low speed such that mooring/rigging operations can be executed. This period is shaded in yellow. Whenever the ship is within x_{dec} and the absolute surge velocity is within an acceptable value u_{rest}, the propeller is set to rest $n = 0$. This marks the start of the stationing period where the rudder has to track the reference yaw angle and its rate with whatever surge velocity the ship has. During this period, once set to rest, the propeller can not be turned on again. Thus, the optimization variable is reduced to just one, that is, the rudder angle. The modified constraint can be expressed as follows:

$$n\left(0\right) \longleftarrow 0, \quad \text{if } \left|\bar{u}\left(\bar{t}_j\right)\right| \leq u_{\text{rest}} \text{ and } x_{0,\text{ref}}\left(\bar{t}_j + \Delta\bar{t}\right) \leq x_{\text{dec}}. \tag{4.5–3}$$

In addition to equations (4.5–1) to (4.5–3), whenever the absolute difference between the reference and the actual surge velocity over $\Delta\bar{t}$ (acceleration) is less than an acceptable limit $\dot{u}_{\text{lim}}$, the previously obtained propeller revolution number is used instead of regarding it as an optimization variable. This is to further minimize the unnecessary switch in the propeller's mode. This constraint will bypass all of the above constraints when it is active. This constraint can be expressed as follows:

$$n\left(0\right) \longleftarrow n\left(\bar{t}_j - \Delta\bar{t}\right), \quad \text{if } \left|\frac{u_{\text{ref}}\left(\bar{t}_j + \Delta\bar{t}\right) - \bar{u}\left(\bar{t}_j\right)}{\Delta\bar{t}}\right| \leq \dot{u}_{\text{lim}}. \tag{4.5–4}$$

The additional threshold values for the surge velocity and the surge acceleration introduced in equations (4.5–1) to (4.5–3), i.e., $\dot{u}_{\text{acc}}$, $\dot{u}_{\text{dec}}$, u_{rest}, and $\dot{u}_{\text{lim}}$, were determined based on the reference trajectory. They can be obtained manually or automatically by taking the difference between two consecutive surge velocities that are separated by $\Delta\bar{t}$. This means they are basically accelerations. Moreover, the longitudinal position that defines the starts of the deceleration region x_{dec} can also be determined automatically from the reference trajectory, e.g., the location when the first condition in equation (4.5–2) is true. It can also be manually chosen based on practical considerations, e.g., a restricted region where all ships must be strictly slow.

4.5.1 Simulation Condition

The trajectory tracking was simulated with the mathematical model of EOShip under a constant wind disturbance (constant U_{T} and χ). The initial states of the ship at the start of the simulation was equal to that of the reference trajectory or $\bar{\mathbf{x}}\left(0\right) = \mathbf{x}_{\text{ref}}\left(0\right)$. To

[16]Do not get confused. At the current time $\bar{t}_j$, the reference state variables are the those that are $\Delta\bar{t}$ ahead, thus the notation $\bar{t}_j + \Delta\bar{t}$ to refer to the current reference state variables. As a consequence, the previous reference state variables is denoted with $\bar{t}_j$.

consider the real-time application of CMA-ES as the solver in the NMPC, the simulation was done in Simulink which allows real-time (clock-time) simulations. The block diagram of the simulation is shown in figure 4-3. Moreover, the parameters used in the simulation for trajectory tracking of a complete docking operation with CMA-ES are summarized in table 4-1.

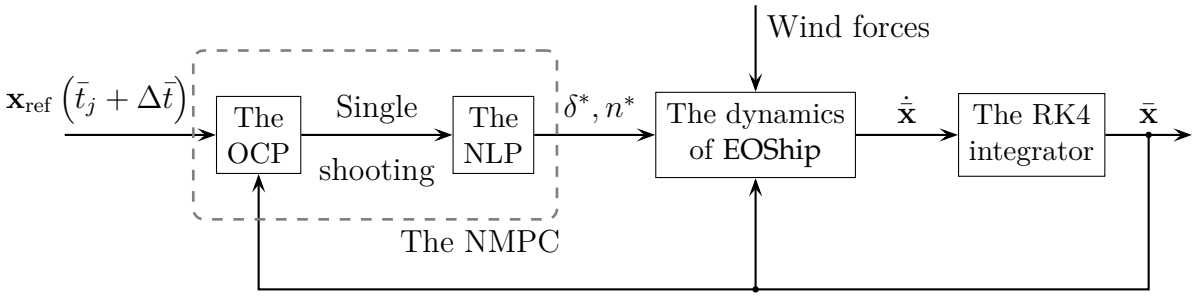


Figure 4-3. Block diagram of the trajectory tracking simulation.

Table 4-1. Parameters used in the simulation of the trajectory tracking with CMA-ES.

Parameters	Values	Units
$\Delta\bar{t}$	1	s
$\delta_{\max}$	30	deg
$n_{\max}$	15	rps
U_{T}	0.477	m/s
χ	82.5	deg
$\dot{u}_{\text{acc}}$	1×10^{-3}	m/s^2
$\dot{u}_{\text{dec}}$	4×10^{-3}	m/s^2
u_{rest}	2.5×10^{-2}	m/s
$\dot{u}_{\text{lim}}$	1×10^{-4}	m/s^2
x_{dec}	1.6	m
$\bar{\mathbf{x}}\,(0)$	$\mathbf{x}_{\text{ref}}\,(0)$	
$\mathbf{w}$	$\begin{bmatrix} 1 & 10 & 1 & 1 & 10 & 10 \end{bmatrix}^\top$	

The prediction horizon $\Delta\bar{t}$ should be chosen not too short nor too long. The shorter the prediction horizon is, the more likely the control commands to aggressively switch their mode. Longer prediction horizon is not preferred because the obtained optimal control commands may fail to counter the external disturbances that are accumulating outside the NMPC (less reactive). In the simulation, the prediction horizon was set as $\Delta\bar{t} = 1\,\text{s}$. This was chosen because it is also the average time the CMA-ES could solve the given OCP. In other words, the CMA-ES must solve the NLP in less than $\Delta\bar{t}$. Note that in the simulation, the prediction horizon is also equal to the sampling period of the actual states. The sampling period refers to the instant when OCP is formulated. It also refers to the increment of the actual time outside the NMPC $\bar{t}$ when integrating the dynamics in the simulation and the update rate of the actuators; see figure 4-3.

大阪大学 OSAKA UNIVERSITY

The additional threshold values for the surge velocity and the surge acceleration were rounded down to the first nonzero decimal. The weight factors, $w_i \in \mathbf{w}$, were equal to 10 for $i = 2, 5, 6$, i.e., for the surge velocity $\bar{u}$, the yaw angle $\bar{\psi}$, and the yaw rate $\bar{r}$. This is to give a sense of importance to those state variables. The rest of w_i were set to 1.

4.5.2 Simulation Result

The visualization of the simulation of a trajectory tracking for docking operation is presented in figure 4-4. Meanwhile, the comparison between the reference states and the actual states is presented in figure 4-5. In both figures, the reference trajectory is shown in black and the actual trajectory is shown in red. The periods are shown in the same colors as explained in section 4.5. However, one must understand that in this simulation, the periods were divided not based on the actual time $\bar{t}$. Instead, they were determined based on the conditions shown in equations (4.5–1) to (4.5–3).

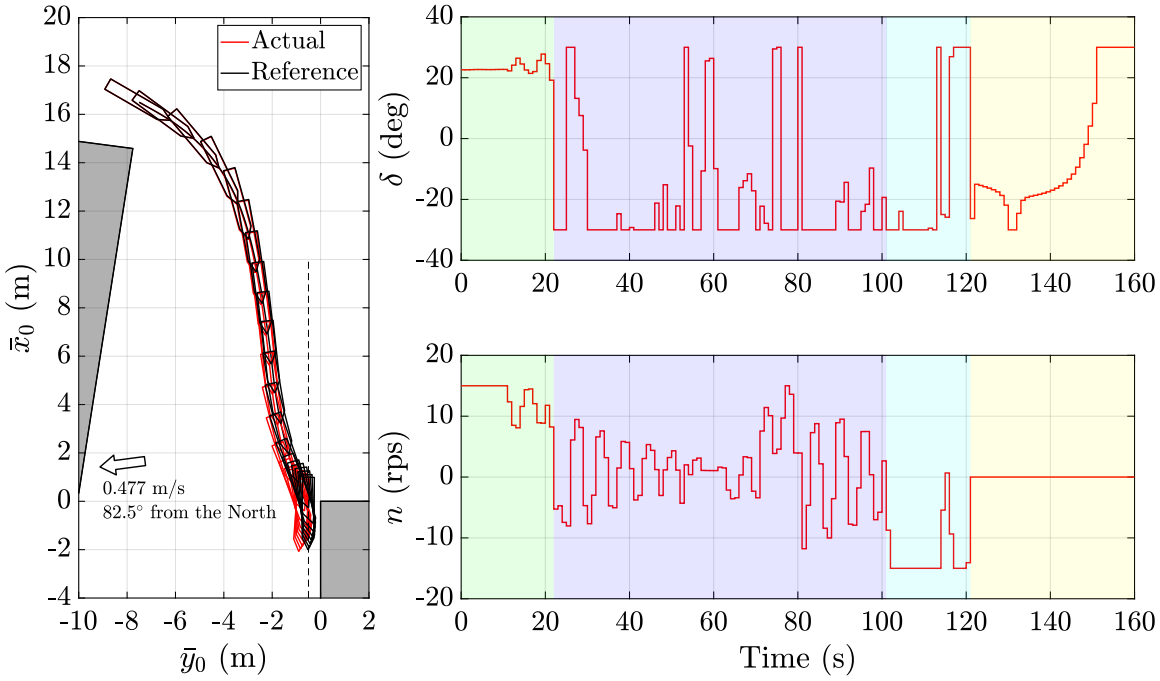


Figure 4-4. Simulation of trajectory tracking for a complete docking operation with CMA-ES: visualization and control commands histories.

From figure 4-4, it can be seen that within the main turn period (shown in purple), the time history of the propeller revolution number n shows a decaying trend with oscillation (mode switch). This results in the surge velocity (and consequently other velocities) oscillating as shown in figure 4-5. In the deceleration period, one can observe that the propeller was always in reverse mode as desired. This allowed the ship to decelerate quickly as shown in cyan in figure 4-5. Moreover, in the stationing period (shown in yellow), the ship struggled to position itself against the wind to be at the desired yaw angle with almost zero surge velocity. The deviations of the actual states from the reference states at the final time t_f (the end of the trajectory), i.e., $e_i\left(t_\mathrm{f}\right) = \bar{x}_i\left(t_\mathrm{f}\right) - x_{\mathrm{ref},i}\left(t_\mathrm{f}\right)$ are acceptable enough, considering that EOShip is a conventional underactuated ship.

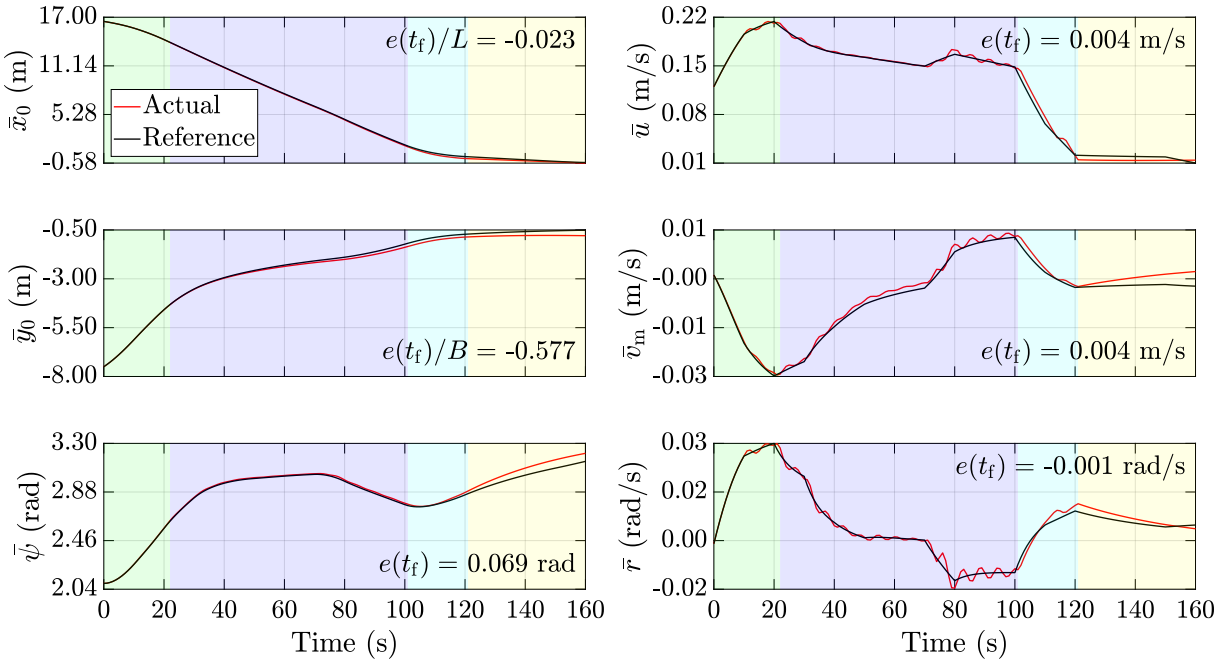


Figure 4-5. Comparison between the actual states and the reference states in the simulation. B is the EOShip's draft (see table 2-1).

The formulated OCP and the NLP performed quite nicely in the simulation to generate the necessary control commands such that the ship was kept at the desired trajectory. This is demonstrated by the decaying and converging trend of the surge velocity, the sway velocity, and the yaw rate. In other words, stabilization of the state variables with respect to the reference state variables was observed. Moreover, modifying the OCP inside the NMPC based on some criteria gave the desired result, that is, the unnecessary switch of the propeller revolution number was attenuated to some extent.

4.6 Simulation: Harbor-entrance Operation

In section 4.2.2 we have established that this operation is formulated as a path following operation. Recall that when generating the reference trajectory for this operation, the velocities were not the main concern such that the propeller revolution number was assumed constant. Moreover, at the end of the operation, the ship only needs to be at a location near the actual final docking point. In this location, the ship does not have to stop completely. The NLP for this operation is formulated by simplifying the general NLP shown in equation (4.4–1). This can be formulated as follows:

$$\begin{aligned}
&\text{determine} && \delta^{*}\left(0\right) \\
&\text{that min.} && J_{\mathrm{mc}} = \sum_{i=1}^{3} w_i \left(\frac{\eta_i\left(\bar{t}_j\right) - \eta_{\mathrm{ref},i}}{\eta_{\mathrm{ref},i}} \right) && \text{(4.6–1a)} \\
&\text{subject to} && \dot{\mathbf{x}}\left(t\right) = \mathbf{f}\left(\mathbf{x}\left(0\right), \mathbf{u}\left(0\right)\right), && \text{(4.6–1b)} \\
& && \mathbf{x}\left(0\right) = \bar{\mathbf{x}}_j,\ \boldsymbol{\eta}\left(\Delta\bar{t}\right) = \boldsymbol{\eta}_{\mathrm{ref}}, && \text{(4.6–1c)}
\end{aligned}$$

$$-\delta_{\max} \leq \delta\,(0) \leq \delta_{\max}. \tag{4.6–1d}$$

Here the control objective is to maintain the ship at the desired pose: position and yaw angle, without considering the velocities. Thus, the objective of the NLP is to find the optimal rudder angle such that the deviation of the predicted pose $\boldsymbol{\eta}$ from the reference pose $\boldsymbol{\eta}_{\text{ref}}$ is minimized. Note that the pose belongs to the states as shown in equation (2.1–2):

$$\boldsymbol{\eta} = \begin{bmatrix} x_0 & y_0 & \psi \end{bmatrix}^\top.$$

The weight factors were 1 for $i = 1, 2$, i.e., for the position x_0 and y_0. Moreover, the weight for the yaw angle was 100. Using only the pose in J_{mc} is equivalent to setting the weight factors for the velocities in equation (4.4–1a) to 0.

In a trajectory tracking operation, the reference states at each sampling period is determined with respect to time, i.e., that at $\Delta\bar{t}$ second ahead. However, in a path following operation, the reference pose is determined based on the instantaneous position of the ship. This is why we dropped the time-varying notation. The reference pose is updated whenever the ship gets inside a *radius of acceptance*, denoted by R_{ac}. In this simulation it was set as $R_{\text{ac}} = 0.1\,\text{m}$. The module that is responsible for selecting the reference pose is called the *guidance* as shown in figure 4-6.

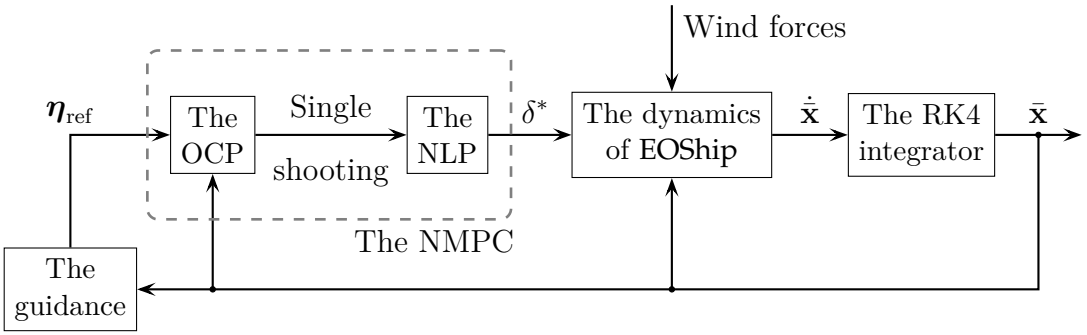


Figure 4-6. Block diagram of the path following simulation.

The simulation was done with a sampling period $\Delta\bar{t} = 1\,\text{s}$ under a constant wind disturbance; the same as that in the simulation of trajectory tracking shown in table 4-1. The propeller revolution number was set as constant in the whole simulation at a very low number, i.e., $n = 5\,\text{rps}$. Note that one can use any values. However, a small revolution number is preferred to prevent excessive and unnecessary acceleration. Meanwhile, the rudder angle δ was limited to the same $\delta_{\max}$ shown in table 4-1.

The visualization of the simulation of a path following for the harbor-entrance operation is shown in figure 4-7. On the other hand, the comparison between the actual states and the reference states is shown in figure 4-5. It can be seen that CMA-ES performed very well to find the optimal rudder angle to keep the ship at the desired pose (position and yaw angle) until it reached the desired final location: about $L/2$ m before and $L/3$ m perpendicular to the actual final docking point (shown as green dot in figure 4-8). The ship arrived at this location with a surge velocity of $0.135\,\text{m/s}$ and $178\,\text{deg}$ yaw angle, which was very close to the reference values.

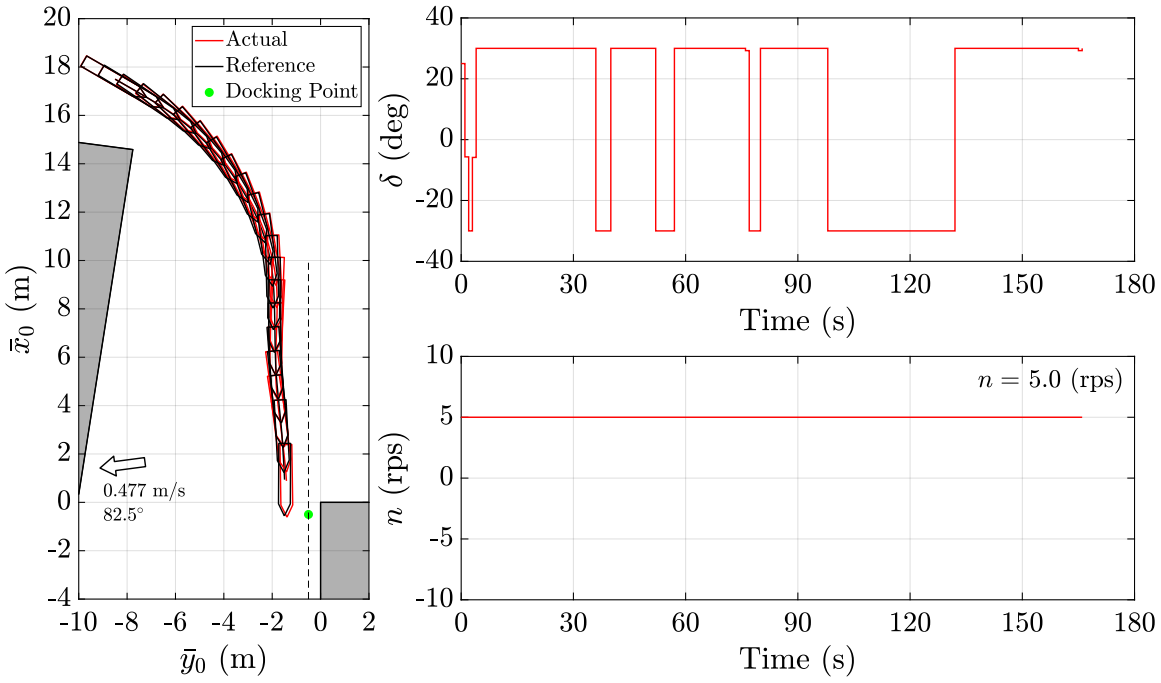


Figure 4-7. Simulation of path following for the harbor-entrance operation with CMA-ES: visualization and control commands histories.

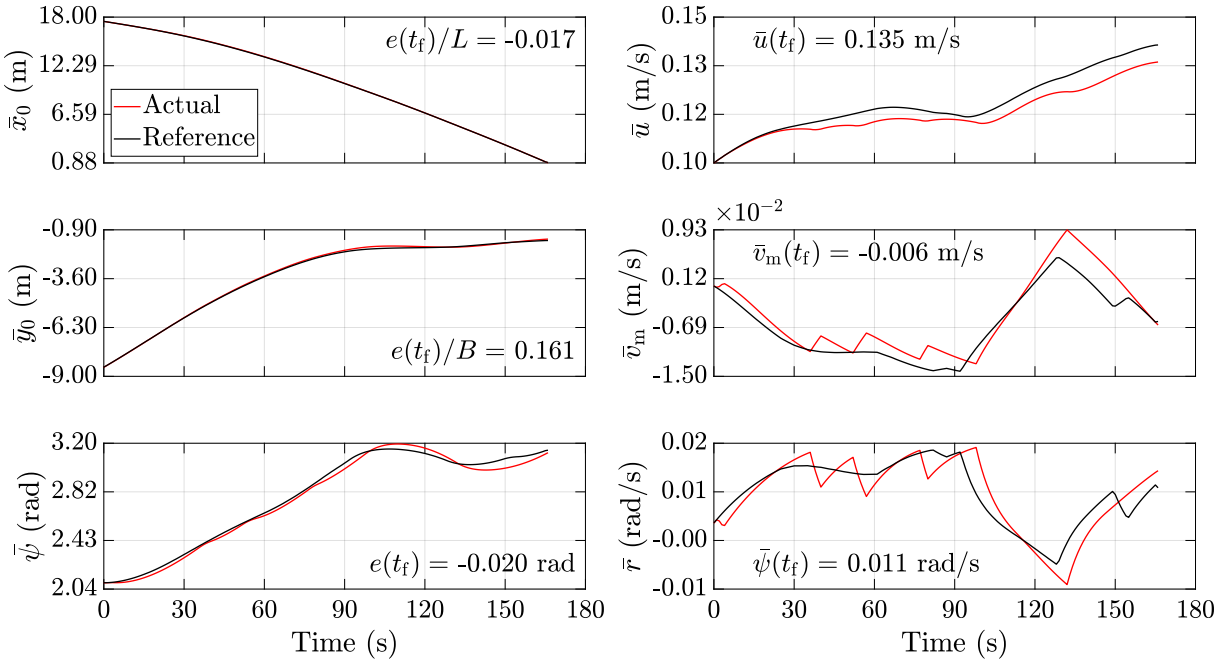


Figure 4-8. Comparison between the actual states and the reference states in the simulation. B is the EOShip's draft (see table 2-1).

It is very important to note that this harbor-entrance operation is quite unrealistic. It is shown for the sake of demonstration on the real-time application of CMA-ES for marine use. Moreover, the computational cost from using CMA-ES is quite exorbitant because the NLP in equation (4.6–1) is just a one-dimensional optimization problem that

can be solved easily with any gradient-based solvers. Nonetheless, it is still presented and discussed as a demonstration.

4.7 Concluding Remark

This chapter discusses how a computationally demanding global-optimization solver known as CMA-ES (covariance matrix adaptation evolution strategy) can be used to solve OCPs related to automatic docking operation. It is very powerful when it comes to solving a very complicated and ill-conditioned problem without the need to provide some initial guesses. However, such an advantage comes at the cost of long computational time. Thus, its real-time application must be investigated, which is demonstrated in this study.

Here two simulations were performed: a trajectory tracking for a complete docking operation and a path following for the harbor-entrance operation. In the trajectory tracking simulation, CMA-ES proved itself able to find the optimal control command vector to track all six reference state variables to some extent, that is, the ship was at the desired states and at the scheduled time. Practical matters regarding mechanical limitations on the ship's propeller were also partially considered by modifying the OCP inside the NMPC based on certain criteria. The resulting time histories of the states show decaying and oscillating behaviors that indicate stabilization. Nonetheless, judging by the states at the stationing stage, external devices (or auxiliary actuators) are necessary to ensure the overall safety of the docking operation. In the path following simulation, despite its substantially excellent performance in finding the optimal rudder angle that kept the ship at the desired pose, one can say that CMA-ES is a little bit too overpowered to solve a one-dimensional optimization problem.

CMA-ES, with its main advantage being a gradient-free solver, can be used to solve many optimization problems related to most conventional underactuated ships that are only equipped with a rudder and a propeller. As shown in chapter II, the mathematical model of such a ship is rather complicated and highly nonlinear. Consequently, the optimization problem becomes more complicated and highly nonlinear with the possibility of discontinuities in the search space. Such nonlinearity and discontinuity can greatly affect the performance of a gradient-based solver, over which CMA-ES may and shall prevail.

The NMPC/controller discussed in this chapter is relevant to the idea of a general MASS; for **AS degree 1** to **AS degree 4**. Provided that a mathematical model of the ship is available, the shipmaster can use the formulated NMPC to automate a trajectory tracking operation or just a path following operation. As a caveat, the ship must have enough computational power to implement the NMPC with CMA-ES to the fullest. For reference, the proposed NMPC and its discussion has been presented online at the 1st International Conference on the Stability and Safety of Ships and Ocean Vehicle (June, 2021). It is available online as a part of the conference proceeding: **publication [A]**. Furthermore, this study corresponds to **contribution II** to design a motion control for automatic docking operation with CMA-ES.

V

Multipurpose Low-speed Positioning System with VecTwin Rudder

5.1 Introduction

Docking operations have to be executed at a low speed and sometimes in a narrow waterway. It may involve special maneuvers like crabbing/sway, rotation, and reversal; all done at a low speed. However, the low-speed requirement is met by keeping the main thrust minimal. This makes it very difficult for a conventional underactuated ship to execute even just a rotation with just the rudder, that is, without side thrusters or any help from tugboats.

For an overly/fully actuated ship, such special low-speed maneuvers are termed dynamic positioning (DP) [33, 47]. A ship that is capable of doing such low-speed maneuvers—a DP ship—usually equipped with redundant main thrusters or multiple side/azimuth thrusters. This makes it very maneuverable at a low speed even in a narrow waterway. The dynamic positioning system (DPS) on board is able to automatically execute such low-speed maneuvers that are necessary to achieve the control objective (see table 1-2), especially for position/station keeping and positioning in the presence of wave, wind, and current [38,39,68,69]. A DP controller also saw implementation in automating docking operations as reported in [18,19,21].

Installing and maintaining multiple azimuth thrusters can be quite expensive. As an alternative to this arrangement, it is hypothesized that a high-performance VecTwin rudder (designed by Japan Hamworthy & Co., Ltd. [48]) can be utilized to design a novel automatic system that is capable of executing low-speed maneuvers in a calm harbor area. This VecTwin rudder comprises twin fish-tailed rudders that can move independently up to 140 degree. In this thesis, the positioning system is called VecTwin positioning system, abbreviated as VTPS. With at least one bow thruster, the system can execute special maneuvers like *hover*, crabbing/sway, rotation, and reversal, all at a constant propeller revolution number in forward mode [70] (see figure 1-2b). This makes the system suitable to automate a docking operation that has to be carried out at a low speed. In principle, this arrangement does not make the ship fully actuated since there is no actuator that directly accelerates the ship in the sway direction.

Hypothesis H.V: Ability of the VecTwin rudder

The ability of a ship equipped with a VecTwin rudder to execute special maneuvers like hover, crabbing/sway, rotation, and reversal—all at a constant propeller revolution number in forward mode and with at least one bow thruster —makes it possible to design a novel automatic system for low-speed operations in a calm waterway.

5.1.1 On the Design of the VTPS

Hypothesis **H.V** and the overall design of the VTPS are based on the ability of the VecTwin rudder to achieve *hovering*. The term *hover* refers to a condition when the ship is almost stationary but the propeller is still rotating normally. In a hover condition, the forces in all directions: surge, sway, and yaw are almost in equilibrium such that the ship does not accelerate: stationary. The hover condition can be achieved when the rudders are at a certain pair of rudder angles termed as the *hover rudder angle*. The standard hover rudder angles are -75 deg for the port rudder and 75 deg for the starboard rudder. The hover rudder angle has a very important role in the design of the system. It is assumed that the rudder angle commands δ within a range around the hover rudder angle are proportional to the resulting forces, i.e., have a linear relationship. In other words, the true relationship is linearized around the hover rudder angle.

Assumption A.5-1: Linear relationship at around hover rudder angle

The relationship between the rudder angle commands and the resulting forces is linear within a range around the hover rudder angle. This means one can find some constants that map the rudder angle commands to the forces.

Ideally, one can obtain the commands-forces relationship through experiments of bollard pull tests (zero speed) with different combinations of rudder angles. However, when the resources and time to conduct such experiments are limited, one can opt to simulate them in CFD (computational fluid dynamics) simulations. Each of these simulations results in the total forces due to the interaction between the propeller, the rudder, and the hull, at a bollard pull condition. From these results, one can do multiple linear regression to obtain the linear relationship between the forces and the rudder angle command.

In general, the approach when designing the VTPS is rather different from the conventional design of a DPS. In the usual DPS, each thruster has its own commands-forces relationship; each can contribute thrust independently. However, this is not the case for the VecTwin rudder where both rudders act as one system instead of two independent systems. It resulted in a nondiagonal commands-forces matrix (usually known as the force coefficient matrix [33]) and a compact control forces allocation module; different from those in the usual PID-based DPS [33, 69, 71, 72]. Furthermore, the VTPS does not incorporate a wave filtering method that is crucial for offshore DPS [73, 74] to eliminate the low-amplitude high-frequency motions from the waves. This is because the VTPS is designed to be used mainly for operations in a calm harbor area where oscillatory motions

due to high-frequency waves are assumed very small and negligible. Nonetheless, the VTPS still incorporates the traditional decoupled PID controller to regulate the required forces in the surge, the sway, and the yaw motions.

The most unique novelty of the VTPS is that the main propeller is set at a constant revolution in forward mode, all the time. This breaks the standard of the usual DPS where the thrusters are actively contributing to fulfilling the required forces. In the VTPS, the thrust is constant (passive); it is the VecTwin rudder that is mainly responsible for allocating the constant thrust to meet the required forces.

5.1.2 Scopes

The novel VTPS was tested in several combined automatic docking (as positioning operations) and position/station keeping experiments on the VTShip: a scale model ship that is equipped with VecTwin rudder, a fixed-pitch single-screw propeller, and a bow thruster (see table 2-2 for its principal particulars). These experiments were conducted at the Inukai experiment pond that belongs to Osaka University. During these experiments, the VTPS performed under random external disturbances from the environment, dominantly from wind and gusts. Based on its performance observed during the experiments, the VTPS has the potential to be an addition to automatic docking systems that were tested in experiments [6–8, 13, 21] and those that were tested in simulations [4, 11, 15, 36].

This study serves as a preliminary investigation to test hypothesis **H.V** that one can indeed incorporate VecTwin rudder to design a positioning system. This hypothesis was tested directly in scale model experiments. The VTPS was tested and validated first to see that it practically works well in the experiments; bypassing verification with numerical simulation like the usual design procedure. Thus, one can say that the VTPS is still *experimental*; indicating that it is, in some sense, a working prototype.

5.2 Commands-Forces Relationship

This section explains the relationship between the control commands $\mathbf{u}$ and the resulting forces. This relationship is termed the *commands-forces relationship*. We shall use the same notation introduced in the previous chapters. However, since the VTShip is equipped with a bow thruster and two rudders that can move independently, the control command vector (now denoted as $\mathbf{u}_\mathrm{c}$) is expressed as follows:

$$\mathbf{u}_\mathrm{c} = \begin{bmatrix} \delta_\mathrm{p} & \delta_\mathrm{s} & n_\mathrm{B} \end{bmatrix}^\top, \tag{5.2–1}$$

where δ_p and δ_s denote the port and the starboard rudder angle, respectively, while n_B denotes the revolution number of the bow thruster. It is also convenient to put a pair of rudder angles (port and starboard) into a vector $\mathbf{u}_\delta$ as follows:

$$\mathbf{u}_\delta = \begin{bmatrix} \delta_\mathrm{p} & \delta_\mathrm{s} \end{bmatrix}^\top. \tag{5.2–2}$$

It is worth noting that the propeller is actually an actuator. However, the propeller revolution number n is kept constant which renders it unavailable as a free actuator (can not actively contribute). Nonetheless, the interactions between the dynamics of the propeller on the rudders and the hull were of course considered in the CFD simulations. This matter is discussed in the following subsection.

5.2.1 Propeller-VecTwin-Hull Interactions

Each rudder of the VecTwin rudder can move independently up to 140 deg: $\delta_p \in [-105, 35]$ for the port rudder and $\delta_s \in [35, 105]$ for the starboard rudder. This wide range of rudders, with the attached reaction fins, greatly increases the number of executable maneuvers [48].

The interactions between the forces exerted by the propeller, the rudders, and the hull make it possible for the ship to hover and even reverse at a constant propeller revolution in forward mode. The intricacy of these interactions can make it challenging enough to theoretically model the relationship between the rudder angle commands and the resulting forces [75]. Fortunately, one can always try to obtain the commands-forces relationship empirically, e.g., regression or curve fitting of the results from either experiments or simulations.

It is undoubtedly preferred to conduct tank tests: bollard pull tests with a scale model ship, and then find the relationship from the results. As mentioned earlier, when the resources and opportunity to carry out such tests are limited or unavailable, one can opt to simulate them in CFD simulations. However, it must be done on the condition that they are simulated with acceptable fidelity and that the final implementation indeed results in the desired outcomes. The following sub-subsections explain these simulations and how to obtain the relationship.

5.2.1.1 CFD Simulations

Recall from subsection 5.1.2 and assumption **A.5-1**, it is assumed that the rudder angle commands within a range around the hover rudder angle have a linear relationship with the corresponding forces, i.e., linearized around the hover rudder angle. For this reason, the CFD simulations should be carried out for combinations of rudder angles within a range around the standard hover rudder angle. We know that the standard hover rudder angle is at $\delta_p = -75$ deg and $\delta_s = 75$ deg. On this basis, the simulations were performed with combinations of $\delta_p \in [-80, -70]$ and $\delta_s \in [70, 80]$, at 5-degree intervals. Meanwhile, the propeller was constant at $n = 10$ rps. In total, there were nine combinations of rudder angles, thus nine CFD simulations had to be performed.

The simulation was performed and solved by a commercial solver STAR-CCM+ ver.2021.2.3 with steady Reynolds-averaged Navier-Stokes (RANS) approach. The computational domain was rectangular with the ship's center of gravity (x_G, y_G) as the origin. The domain spans $5L$ in the longitudinal direction x, spans $4L$ in the lateral direction y, and spans $2L$ in the vertical direction. The computational domain is shown in figure 5-1. The top edge of the hull indicates the location of the water surface and the top boundary of the computational domain.

The boundary conditions and the initial conditions of the computational domain were set such that the free stream velocity U_∞ and the ship's velocity u are both zero: a bollard pull condition. Because of the zero-speed condition, wave generations (free-surface effects) that occur around the hull are negligible. For this reason, double-body assumption was applied and thus symmetric boundary condition at the water surface. No-slip wall condition was applied on the surface of the hull of the VTShip model. Moreover, pressure boundary conditions were applied on the four sides of the domain and symmetric boundary condition was applied on the bottom of the domain.

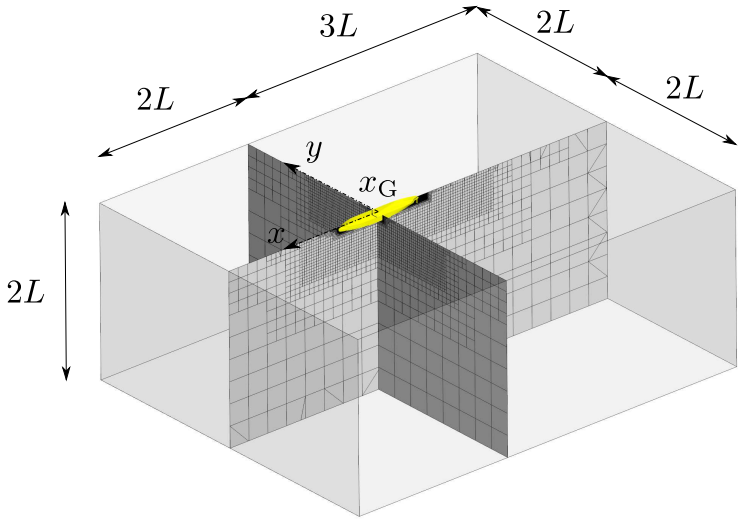


Figure 5-1. Computational domain of the CFD simulations.

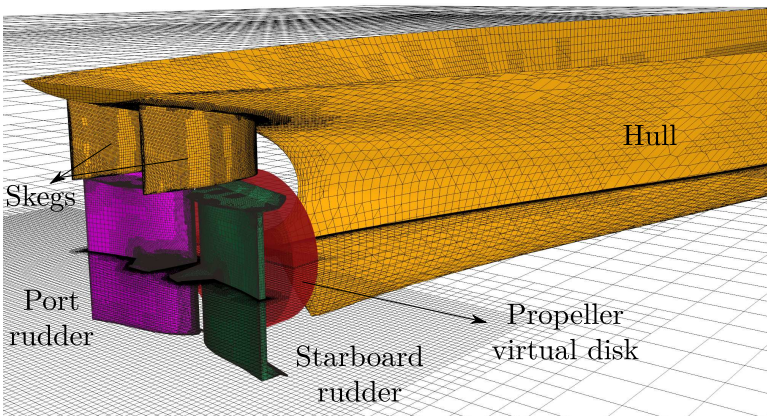


Figure 5-2. Modeling of the propeller, the rudder, the hull, and the skegs.

As mentioned earlier, the simulations must be performed at an acceptable fidelity. Here, the mesh density of the computational domain was verified and validated with captive model tests and free-running tests of VTShip shown in [76]. On average, there were 11 million cells that built up the whole computational domain at each simulation. A detailed setup and conditions of the simulations are presented in [76].

The propeller was modeled as a virtual disk as shown in figure 5-2. The forces induced by the propeller were modeled via the blade element method [77]. This is to accurately capture the fluid interaction between the propeller, the rudders, and the hull, within a moderate computational cost. In addition, the arising turbulence was modeled using the shear stress transport (SST) $k-\omega$ model [78]. Note that the discussion on the CFD is not within the scope of this thesis.

5.2.1.2 On the Results of Simulations

The results from these simulations are the total forces and moment due to the propeller-rudder-hull interaction in the surge direction X_{CT}, in the sway direction Y_{CT}, and the yaw

moment N_{CT}. They act on the center of gravity x_{G}. These total forces can be expanded as follows:

$$X_{\mathrm{CT}} = X_{\mathrm{HC}} + X_{\mathrm{P}} + X_{\mathrm{Rp}} + X_{\mathrm{Rs}} \tag{5.2–3a}$$

$$Y_{\mathrm{CT}} = Y_{\mathrm{HC}} + Y_{\mathrm{Rp}} + Y_{\mathrm{Rs}} \tag{5.2–3b}$$

$$N_{\mathrm{CT}} = N_{\mathrm{HC}} + X_{\mathrm{Rp}} y_{\mathrm{Rp}} + X_{\mathrm{Rs}} y_{\mathrm{Rs}} + Y_{\mathrm{Rp}} x_{\mathrm{Rp}} + Y_{\mathrm{Rs}} x_{\mathrm{Rs}}, \tag{5.2–3c}$$

where subscripts P, Rp, and Rs denote the forces due to the propeller, the port rudder, and the starboard rudder, respectively, from the CFD simulations. Meanwhile, subscript HC denotes the *forces acting on the hull's surface due to the actuators*. The coordinates $(x_{\mathrm{Rp}}, y_{\mathrm{Rp}})$ and $(x_{\mathrm{Rs}}, y_{\mathrm{Rs}})$ denote the position of the port rudder and the starboard rudder from x_{G}, respectively.

The total forces X_{CT} and Y_{CT} act on a location near the rudder, denoted by $(x_{\mathrm{fR}}, y_{\mathrm{fR}})$, that can be computed as follows:

$$x_{\mathrm{fR}} = \frac{Y_{\mathrm{Rp}} x_{\mathrm{Rp}} + Y_{\mathrm{Rs}} x_{\mathrm{Rs}}}{Y_{\mathrm{CT}}} \tag{5.2–4a}$$

$$y_{\mathrm{fR}} = \frac{X_{\mathrm{Rp}} y_{\mathrm{Rp}} + X_{\mathrm{Rs}} y_{\mathrm{Rs}}}{X_{\mathrm{CT}}}. \tag{5.2–4b}$$

It is assumed that the location of the acting point of these forces is equal to the longitudinal position of the rudder. Moreover, to simplify everything, we shall ignore the effect of N_{CT}. These are summarized into the following assumptions.

Assumption A.5-2: Constant x_{fR} and y_{fR}

Actually, the x_{fR} and y_{fR} vary with the rudder angle. However, for the sake of practicality and simplicity, it is assumed that the forces act on a fixed location. Here, x_{fR} is assumed to be equal to the longitudinal distance of the rudder from x_{G}.

Since the rudder position is very close to the x-axis ($y_{\mathrm{Rs}} \ll x_{\mathrm{Rs}}$, $y_{\mathrm{Rp}} \ll x_{\mathrm{Rp}}$), the contribution of X_{Rp} and X_{Rs} to the moment is very small. Thus, one can assume that y_{fR} is at the origin or $y_{\mathrm{fR}} = 0$, hence X_{CT} acts along the centerline. This is shown in figure 5-3.

Assumption A.5-3: Neglect the effect of N_{HC}

In the study, we leave the N_{HC} out of consideration. Consequently, the N_{CT} is also left out. Here we thrive to find an approach that put performance on top of accuracy. In the end, everything is implemented in a closed-loop feedback control where unmodeled dynamics, uncertainties, and all assumptions that we have made, can be corrected by the feedback control.

The resulting forces X_{CT} and Y_{CT} from the nine CFD simulations are summarized in table 5-1.

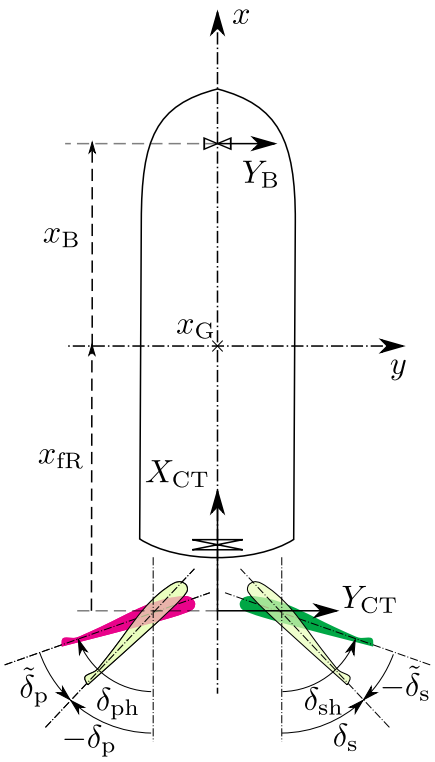


Figure 5-3. Ship-fixed coordinate system and rudder angle definition.

Table 5-1. Summary of the total forces from the CFD simulations at $n = 10\,\text{rps}$.

δ_p (deg)	δ_s (deg)	X_{CT} (N)	Y_{CT} (N)
−80	70	0.0735	−0.0400
−80	75	−0.0620	−0.0482
−80	80	−0.2061	0.0507
−75	70	0.1298	0.0544
−75	75	−0.0063	0.0655
−75	80	−0.1089	0.1816
−70	70	0.3724	0.1152
−70	75	0.1380	0.1552
−70	80	0.0030	0.2678

5.2.1.3 Regression: Linearity Around the Hover Rudder Angle

Let us put the total forces X_{CT} and Y_{CT} into a vector $\mathbf{f}_{CT}$ as follows:

$$\mathbf{f}_{CT} = \begin{bmatrix} X_{CT} & Y_{CT} \end{bmatrix}^\top. \tag{5.2–5}$$

Assuming that these forces are uncorrelated to each other[17], one can construct a system of linear regression models as follows:

$$\mathbf{f}_{CT} = \tilde{\mathbf{V}}\mathbf{u}_\delta + \mathbf{f}_{itcp}, \tag{5.2–6}$$

where $\mathbf{f}_{itcp}$ is the intercept of the regression model, i.e., the total forces when $\mathbf{u}_\delta = \mathbf{0}$. In the regression model, $\tilde{\mathbf{V}}$ is a symmetric matrix whose rows are the regression coefficients (or slope coefficients, see figures 5-4 and 5-5) obtained upon solving the least square problem for the data points shown in table 5-1. The entries of $\tilde{\mathbf{V}}$ is as follows:

$$\tilde{\mathbf{V}} = \begin{bmatrix} \tilde{V}_{11} & \tilde{V}_{12} \\ \tilde{V}_{21} & \tilde{V}_{22} \end{bmatrix} = \begin{bmatrix} 0.0236 & -0.0296 \\ 0.0192 & 0.0123 \end{bmatrix}. \tag{5.2–7}$$

Recall that the ship is in a hovering state when the total forces are in equilibrium such that the ship does not accelerate. This is equivalent to saying that $\mathbf{f}_{CT} = \mathbf{0}$. Moreover,

[17] There are no possible constants α and β in $X_{CT} = \alpha Y_{CT} + \phi_i$ and $Y_{CT} = \beta X_{CT} + \phi_i$, i.e., no linear relationship between the two forces.

recall that a pair of rudder angles that result in a hovering state is called the hover rudder angle. Setting $\mathbf{f}_{\mathrm{CT}} = \mathbf{0}$ in equation (5.2–6) gives us,

$$\mathbf{u}_{\delta\mathrm{h}} = -\tilde{\mathbf{V}}^{-1}\mathbf{f}_{\mathrm{itcp}}, \tag{5.2–8}$$

where $\mathbf{u}_{\delta\mathrm{h}}$ is the hover rudder angle (yes, even if it is written in singular, it refers to the pair of rudder angles). Now let δ_{ph} and δ_{sh} be the port rudder angle and the starboard rudder angle, respectively, that satisfy the above relation: hovering state. Thus, $\mathbf{u}_{\delta\mathrm{h}}$ is defined as follows:

$$\mathbf{u}_{\delta\mathrm{h}} = \begin{bmatrix} \delta_{\mathrm{ph}} & \delta_{\mathrm{sh}} \end{bmatrix}^{\top}, \tag{5.2–9}$$

which, upon solving the inverse in equation (5.2–8), gives us the corrected values of the hover rudder angle as follows:

$$\mathbf{u}_{\delta\mathrm{h}} = \begin{bmatrix} -78.60 & 73.38 \end{bmatrix}^{\top}. \tag{5.2–10}$$

Notice that the hover rudder angle from the regression shown in equation (5.2–10) is very similar to the standard hover rudder angle $\begin{bmatrix} -75 & 75 \end{bmatrix}^{\top}$. This similarity of the hover rudder angle can be a quantitative measure to assert the accuracy of the CFD results (shown in table 5-1) in representing the hovering phenomenon, indirectly. It also verifies the applicability of the regression model in modeling the relationship between the rudder angles and the resulting forces, in a small range of angles around the hover rudder angle.

Now, because we are interested only in the rudder angle around the hover rudder angle, the intercept $\mathbf{f}_{\mathrm{itcp}}$ has to be excluded from the regression model in equation (5.2–6). This is done by normalizing the rudder angle with respect to the hover rudder angle. Let us introduce the rudder angle around the hover rudder angle, denoted by $\tilde{\mathbf{u}}_{\delta\mathrm{h}}$, as a new variable that can be expressed as follows:

$$\tilde{\mathbf{u}}_{\delta\mathrm{h}} = \begin{bmatrix} \tilde{\delta}_{\mathrm{p}} \\ \tilde{\delta}_{\mathrm{s}} \end{bmatrix} \equiv \begin{bmatrix} \delta_{\mathrm{p}} - \delta_{\mathrm{ph}} \\ \delta_{\mathrm{s}} - \delta_{\mathrm{sh}} \end{bmatrix}, \tag{5.2–11}$$

which is equivalent to

$$\tilde{\mathbf{u}}_{\delta\mathrm{h}} \equiv \mathbf{u}_{\delta} - \mathbf{u}_{\delta\mathrm{h}}. \tag{5.2–12}$$

The definition of these angles is visualized in figure 5-3, where the rudder angle is positive in the counterclockwise direction. Substituting the expression of $\mathbf{u}_{\delta}$ from equation (5.2–12) to the regression model in equation (5.2–6) gives us,

$$\begin{aligned} \mathbf{f}_{\mathrm{CT}} &= \tilde{\mathbf{V}}\mathbf{u}_{\delta} + \mathbf{f}_{\mathrm{itcp}} \\ &= \tilde{\mathbf{V}}\left(\tilde{\mathbf{u}}_{\delta\mathrm{h}} + \mathbf{u}_{\delta\mathrm{h}}\right) + \mathbf{f}_{\mathrm{itcp}} \\ &= \tilde{\mathbf{V}}\tilde{\mathbf{u}}_{\delta\mathrm{h}} + \tilde{\mathbf{V}}\mathbf{u}_{\delta\mathrm{h}} + \mathbf{f}_{\mathrm{itcp}}. \end{aligned}$$

However, from equation (5.2–8),

$$\mathbf{f}_{\mathrm{CT}} = \tilde{\mathbf{V}}\tilde{\mathbf{u}}_{\delta\mathrm{h}} - \underbrace{\tilde{\mathbf{V}}\tilde{\mathbf{V}}^{-1}}_{\mathbf{I}}\mathbf{f}_{\mathrm{itcp}} + \mathbf{f}_{\mathrm{itcp}}.$$

This gives us the linear transformation that maps the rudder angle to the resulting forces as follows:

$$\mathbf{f}_{\mathrm{CT}} = \tilde{\mathbf{V}}\mathbf{u}_{\delta\mathrm{h}}. \tag{5.2–13}$$

What we have just done is shift the origin of the regression model from the natural origin $(0, 0)$ to the hover rudder angle $\mathbf{u}_{\delta\mathrm{h}}$. This means, whenever the total forces $\mathbf{f}_{\mathrm{CT}} = \mathbf{0}$, the rudders are at the hover rudder angle, that is, $\mathbf{u}_{\delta} = \mathbf{u}_{\delta\mathrm{h}}$, which is indeed what one wants to model. When the forces $\mathbf{f}_{\mathrm{CT}} \neq \mathbf{0}$, the ship is *slowly* moving with its rudder angle $\mathbf{u}_{\delta}$ at around the hover rudder angle, governed by the linear transformation $\tilde{\mathbf{V}}$. The transformation shown in equation (5.2–13) is geometrically visualized in figures 5-4 and 5-5 where $\tilde{\mathbf{V}}$ maps the rudder angles to a point on the plane (spanned by $\tilde{\mathbf{V}}$: any linear combinations of the columns of $\tilde{\mathbf{V}}$). The planes in these figures are the image of the transformation $\tilde{\mathbf{V}}$ in equation (5.2–13).

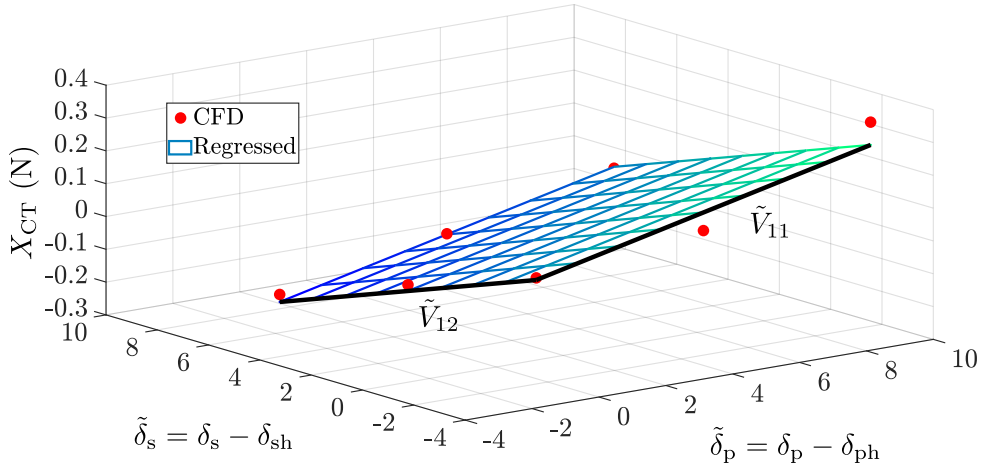


Figure 5-4. Linearity of X_{CT} around the hover rudder angle.

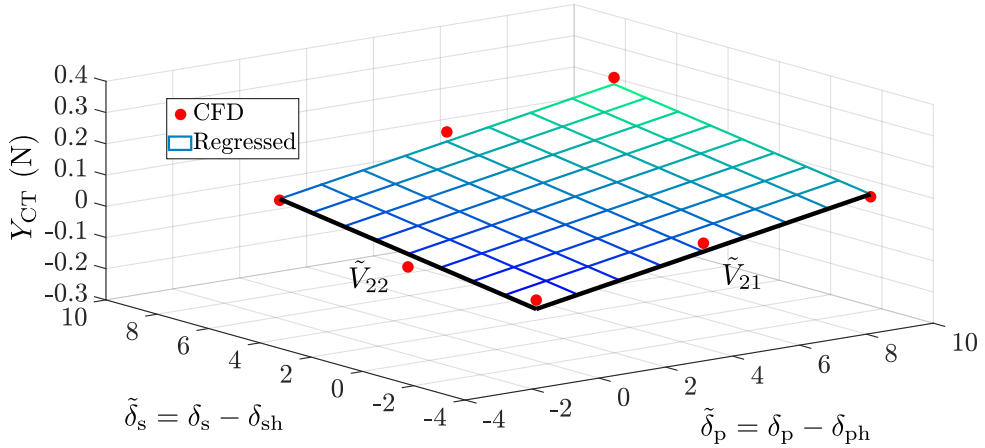


Figure 5-5. Linearity of Y_{CT} around the hover rudder angle.

In addition to assumptions **A.5-1** to **A.5-3**, we establish one more assumption, that is, the relationship in equation (5.2–13) is assumed to hold for a wider range of rudder angle, i.e., $\delta_{\mathrm{p}} \in [-105, -60]$ and $\delta_{\mathrm{s}} \in [60, 105]$. The reason is as follows. The constant

revolution of the propeller results in a constant thrust. At a constant thrust, small rudder angles can result in a lower drag and a higher speed, which consequently accelerates the ship. This is not relevant for the purpose of the VTPS: for low-speed operations. Thus, the rudder angle is limited to not smaller than 60 deg. This assumption is proven acceptable for low-speed operations as demonstrated in several scale model experiments. A brief discussion on this assumption is given in appendix V.A.

Assumption A.5-4: The linearity holds within a wider range

It is assumed that the linear relationship resulting from the linear regression shown in equation (5.2–13) holds for a wider rudder angle. Even if the relationship was obtained for rudder angle $\mathbf{u}_\delta$ between 70 deg and 80 deg, the linearity is assumed true for $\mathbf{u}_\delta$ between 60 deg and 105 deg. This is discussed as an appendix in V.A.

5.2.2 Bow Thruster

The lateral force from the bow thruster, denoted by Y_B, can be approximated as follows:

$$\begin{aligned} Y_\mathrm{B} &= \rho D_\mathrm{B}^2 K_\mathrm{B} n_\mathrm{B}^2 \\ &= C_\mathrm{B} \tilde{n}_\mathrm{B}, \end{aligned} \tag{5.2–14}$$

where D_B is the diameter of the blade of the bow thruster and K_B is the thrust coefficient that was obtained from tank tests. To simplify the expression, one can "force-linearize" the expression by introducing a new constant $C_\mathrm{B} = \rho D_\mathrm{B}^2 K_\mathrm{B}$ and a new variable $\tilde{n}_\mathrm{B} = n_\mathrm{B}^2$. As a consequence, the force Y_B is now proportional to the square of the bow thruster revolution number n_B. This "forced linearization" is intentional such that one can expect a rather aggressive reaction from the bow thruster that should be helpful in several situations.

5.2.3 Complete Commands-Forces Relationship

In the previous subsections, the relationship for each actuator has been established. The rudder angle command was modified such that it is relative to the hover rudder angle (5.2–12). The bow thruster command was also modified as shown in equation (5.2–14). Let us put these modified control commands into a vector $\mathbf{u}$ as follows:

$$\mathbf{u} = \begin{bmatrix} \tilde{\delta}_\mathrm{p} & \tilde{\delta}_\mathrm{s} & \tilde{n}_\mathrm{B} \end{bmatrix}^\top. \tag{5.2–15}$$

Let us also put all the total forces from the actuators into a vector $\mathbf{f}_\mathrm{c}$ as follows:

$$\mathbf{f}_\mathrm{c} = \begin{bmatrix} X_\mathrm{CT} & Y_\mathrm{CT} & Y_\mathrm{B} \end{bmatrix}^\top. \tag{5.2–16}$$

With these, one can combine equation (5.2–13) and equation (5.2–14) as a linear transformation that is, henceforth, referred to as the *complete commands-forces relationship* or just the *commands-forces relationship* as follows:

$$\mathbf{f}_\mathrm{c} = \mathbf{V}\mathbf{u}. \tag{5.2–17}$$

Here the invertible matrix $\mathbf{V}$ is called the *force coefficient matrix* that contains $\tilde{\mathbf{V}}$ (5.2–7) and C_{B} as follows:

$$\mathbf{V} = \begin{bmatrix} \tilde{V}_{11} & \tilde{V}_{12} & 0 \\ \tilde{V}_{21} & \tilde{V}_{22} & 0 \\ 0 & 0 & C_{\mathrm{B}} \end{bmatrix}. \tag{5.2–18}$$

This $\mathbf{V}$ maps the modified control command vector to the resulting actuator forces $\mathbf{f}_{\mathrm{c}}$ and preserves the origin ($\mathbf{f}_{\mathrm{c}} = \mathbf{0}$) at the hover rudder angle $\mathbf{u}_{\delta\mathrm{h}}$ and $n_{\mathrm{B}} = 0$.

Notice that the force coefficient matrix $\mathbf{V}$ is nondiagonal. This is different from the diagonal force coefficient matrix of the usual PID-based DPS [33, 69, 71]. In the usual DPS, each thruster/rudder acts independently. Each actuator has its own command-thrust relationship, e.g., equation (5.2–14). The absence of one or more thrusters should not affect the forces from the others. This is not quite true for the VecTwin rudder where the total forces are due to the interactions between the propeller, both rudders, and the hull. Especially, due to the fact that both rudders interact with each other and act as one system instead of two individual rudders (despite being able to move independently). These interactions were captured and considered when obtaining the regression coefficients $\tilde{\mathbf{V}}$ and consequently the force coefficient matrix $\mathbf{V}$.

5.3 Control Forces Allocation

A ship requires forces to accelerate: to move from one position to another or just to maintain its pose against environmental disturbances. Let $\mathbf{f}_{\mathrm{req}}$ be these required forces as follows:

$$\mathbf{f}_{\mathrm{req}} = \begin{bmatrix} X_{\mathrm{req}} & Y_{\mathrm{req}} & N_{\mathrm{req}} \end{bmatrix}^{\top}, \tag{5.3–1}$$

where X_{req}, Y_{req}, and N_{req} denote the required forces and moment in the surge, the sway, and the yaw directions, respectively. These required forces need to be fulfilled by allocating the forces from the actuators $\mathbf{f}_{\mathrm{c}}$.

From figure 5-3, Y_{B} and Y_{CT} can contribute to fulfilling the required force in the sway direction Y_{req}. They also contribute to satisfying the required moment N_{req}. Meanwhile, only X_{CT} can contribute to the required force in the surge direction X_{req}. The contribution from each actuator to fulfill $\mathbf{f}_{\mathrm{req}}$ can be written as follows:

$$\begin{bmatrix} X_{\mathrm{req}} \\ Y_{\mathrm{req}} \\ N_{\mathrm{req}} \end{bmatrix} = \begin{bmatrix} X_{\mathrm{CT}} \\ Y_{\mathrm{CT}} + Y_{\mathrm{B}} \\ Y_{\mathrm{CT}} x_{\mathrm{fR}} + Y_{\mathrm{B}} x_{\mathrm{B}} \end{bmatrix}, \tag{5.3–2}$$

where x_{B} is the distance of the bow thruster from x_{G} along the x-axis. With equation (5.3–1), one can reformulate equation (5.3–2) into a linear transformation as follows:

$$\begin{bmatrix} X_{\mathrm{req}} \\ Y_{\mathrm{req}} \\ N_{\mathrm{req}} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & x_{\mathrm{fR}} & x_{\mathrm{B}} \end{bmatrix}}_{\mathbf{T}} \begin{bmatrix} X_{\mathrm{CT}} \\ Y_{\mathrm{CT}} \\ Y_{\mathrm{B}} \end{bmatrix}, \tag{5.3–3}$$

where the invertible matrix $\mathbf{T}$ is known as the *configuration matrix* that allocates the control forces to satisfy the required forces. The linear transformation in equation (5.3–3) can be written in a compact form as follows:

$$\mathbf{f}_{\text{req}} = \mathbf{T}\mathbf{f}_{\text{c}}. \tag{5.3–4}$$

Substitution of the commands-forces relationship $\mathbf{f}_{\text{c}}$ shown in equation (5.2–17) into equation (5.3–4) yields a linear transformation that maps the modified control command vector $\mathbf{u}$ (5.2–15) to the required forces $\mathbf{f}_{\text{req}}$. This can be expressed as follows:

$$\mathbf{f}_{\text{req}} = \mathbf{TV}\mathbf{u}. \tag{5.3–5}$$

Since the configuration matrix $\mathbf{T}$ and the force coefficient matrix $\mathbf{V}$ are invertible and constant, their matrix multiplication is also invertible and constant. Thus, the necessary control commands that correspond to the required forces can be computed as follows:

$$\mathbf{u} = (\mathbf{TV})^{-1}\,\mathbf{f}_{\text{req}}$$
$$\mathbf{u} = \mathbf{V}^{-1}\mathbf{T}^{-1}\mathbf{f}_{\text{req}}.$$

Let $\mathbf{Z}$ be the inverse of the matrix multiplication of $\mathbf{T}$ and $\mathbf{V}$, i.e., $\mathbf{Z} = \mathbf{V}^{-1}\mathbf{T}^{-1}$. With this, the above equation can be compactly written as follows:

$$\mathbf{u} = \mathbf{Z}\mathbf{f}_{\text{req}}. \tag{5.3–6}$$

Equation (5.3–6) henceforth is called the *control forces allocation* equation. Note that equation (5.3–6) of course can be solved analytically without the need to calculate $\mathbf{Z}$ numerically. This can be done by first eliminating Y_{B} from equation (5.3–3) to obtain the expressions for Y_{CT} and Y_{B}, and then inserting them to equation (5.2–17). Finally, eliminating either $\tilde{\delta}_{\text{p}}$ or $\tilde{\delta}_{\text{s}}$ from the resulting system of equations yields the expression for the modified control command vector $\mathbf{u}$.

5.4 PID Controller to Regulate the Required Forces

The question is now how to determine the magnitude of the required forces $\mathbf{f}_{\text{req}}$. Regardless of what the control objective is (either positioning or position/station keeping, see subsection 5.1.2), $\mathbf{f}_{\text{req}}$ is regulated by the traditional decoupled PID controller.

Let us now work on a ship-fixed coordinate system where the ship's center of gravity x_{G} is the origin. Let x_0 and y_0 be the position of x_{G} in the earth-fixed coordinate system. Regardless of the origin of the ship-fixed coordinate system, the definition of yaw angle ψ still holds. Also, let $x_{\text{d},0}$ and $y_{\text{d},0}$ be the desired position in the earth-fixed coordinate system. In the same notation, let $\psi_{\text{d},0}$ denote the desired yaw angle.

Consistent with the previous chapter, let $\boldsymbol{\eta}$ denote the instantaneous pose of the ship in the earth-fixed coordinate system as follows (2.1–2):

$$\boldsymbol{\eta} = \begin{bmatrix} x_0 & y_0 & \psi \end{bmatrix}^{\top}, \tag{5.4–1}$$

and let $\boldsymbol{\eta}_{\text{d},0}$ denote the desired pose in the earth-fixed coordinate system as follows:

$$\boldsymbol{\eta}_{\text{d},0} = \begin{bmatrix} x_{\text{d},0} & y_{\text{d},0} & \psi_{\text{d},0} \end{bmatrix}^{\top}. \tag{5.4–2}$$

Moreover, let $\mathbf{v}$ be the vector of the velocities of the ship relative to the earth-fixed coordinate system and evaluated at x_{G} as follows:

$$\mathbf{v} = \begin{bmatrix} u & v & r \end{bmatrix}^{\top}, \tag{5.4–3}$$

where v is the sway velocity at x_{G}.

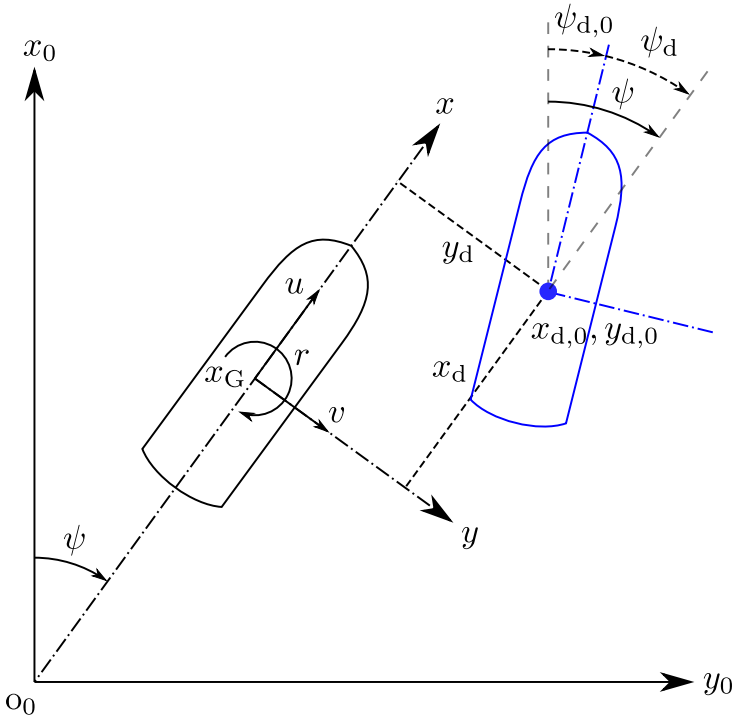


Figure 5-6. Visualization of the desired pose in both coordinate systems.

Since all the forces $\mathbf{f}_{\mathrm{c}}$ are expressed in the ship-fixed coordinate system, it is preferable to explicitly transform the desired pose $\boldsymbol{\eta}_{\mathrm{d},0}$ into the same coordinate system. Take a look at figure 5-6. For both positioning and position/station keeping operations, the control objective is identical, that is, translate the ship-fixed coordinate system $x_{\mathrm{G}} - xy$ (shown in black) and then rotate it such that it is at the desired pose (shown in blue).

Let $\mathbf{e}_0$ be the deviation of the pose $\boldsymbol{\eta}$ from the desired pose $\boldsymbol{\eta}_{\mathrm{d},0}$ in the earth-fixed coordinate system as follows:

$$\mathbf{e}_0 = \boldsymbol{\eta}_{\mathrm{d},0} - \boldsymbol{\eta} = \begin{bmatrix} x_{\mathrm{d},0} - x_0 \\ y_{\mathrm{d},0} - y_0 \\ \psi_{\mathrm{d},0} - \psi \end{bmatrix}. \tag{5.4–4}$$

The vector $\mathbf{e}_0$ is now emanating from x_{G} (x_0, y_0) to $(x_{\mathrm{d},0}, y_{\mathrm{d},0})$, i.e., expressed in an earth-fixed coordinate system that was translated/shifted such that the origin is now at x_{G}. We must now express $\mathbf{e}_0$ in the ship-fixed coordinate system, i.e., change of basis. This can be done by passive rotation of the shifted earth-fixed coordinate system in the clockwise direction by an angle of ψ. Recall from equation (2.1–4) that $\mathbf{R}(\psi)$ is the orthogonal

rotation matrix in the counterclockwise direction as follows:

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

Thus, the rotation matrix in the clockwise direction can be expressed as follows:

$$\mathbf{R}(-\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \equiv \mathbf{R}^{-1}(\psi), \tag{5.4–5}$$

which, in principle, is the inverse of $\mathbf{R}(\psi)$. Notice that the inverse $\mathbf{R}^{-1}(\psi)$ in equation (5.4–5) is just the transpose $\mathbf{R}^{\top}(\psi)$, which proves that $\mathbf{R}(\psi)$ is an indeed an orthogonal transformation. In other words,

$$\mathbf{R}(-\psi) = \mathbf{R}^{-1}(\psi) = \mathbf{R}^{\top}(\psi), \tag{5.4–6}$$

such that,

$$\mathbf{R}\mathbf{R}^{-1} = \mathbf{R}\mathbf{R}^{\top} = \mathbf{I}. \tag{5.4–7}$$

Therefore, to express $\mathbf{e}_0$ in the ship-fixed coordinate system (change the basis), one must rotate the shifted earth-fixed coordinate system by an angle of ψ in the clockwise direction. This can be done via an orthogonal transformation as follows:

$$\mathbf{e} = \mathbf{R}^{\top}(\psi)\ \mathbf{e}_0, \tag{5.4–8}$$

where $\mathbf{e}$ is the deviation from the desired pose but expressed in the ship-fixed coordinate system as follows (see figure 5-6):

$$\mathbf{e} = \begin{bmatrix} x_{\mathrm{d}} & y_{\mathrm{d}} & \psi_{\mathrm{d}} \end{bmatrix}^{\top}. \tag{5.4–9}$$

As a note, due to orthogonality, one can reverse back the transformation (change of basis) in equation (5.4–8) to obtain the following transformation:

$$\mathbf{e}_0 = \mathbf{R}(\psi)\,\mathbf{e},$$

which is similar[18] to the transformation of the kinematics shown in equation (2.1–1) before being differentiated with respect to time.

Now, the deviation $\mathbf{e}$ has to be stabilized at zero by regulating/controlling the required forces $\mathbf{f}_{\mathrm{req}}$. This can be done by the following decoupled PID controller:

$$X_{\mathrm{req}} = K_{\mathrm{PX}}\, x_{\mathrm{d}} + K_{\mathrm{IX}} \int x_{\mathrm{d}}\, \mathrm{d}t - K_{\mathrm{DX}}\, u \tag{5.4–10a}$$

$$Y_{\mathrm{req}} = K_{\mathrm{PY}}\, y_{\mathrm{d}} + K_{\mathrm{IY}} \int y_{\mathrm{d}}\, \mathrm{d}t - K_{\mathrm{DY}}\, v \tag{5.4–10b}$$

$$N_{\mathrm{req}} = K_{\mathrm{PN}}\, \psi_{\mathrm{d}} + K_{\mathrm{IN}} \int \psi_{\mathrm{d}}\, \mathrm{d}t - K_{\mathrm{DN}}\, r. \tag{5.4–10c}$$

[18] Putting $\boldsymbol{\eta} = \mathbf{0}$, i.e., the origin, and $\boldsymbol{\eta}_{\mathrm{d},0} = \boldsymbol{\eta}$ in equation (5.4–4) gives the transformation in equation (2.1–1)

In compact form, equation (5.4–10) can be written as follows:

$$\mathbf{f}_{\text{req}} = \mathbf{K}_{\text{P}}\,\mathbf{e} + \mathbf{K}_{\text{I}} \int \mathbf{e}\,\mathrm{d}t - \mathbf{K}_{\text{D}}\mathbf{v}, \tag{5.4–11}$$

where

$$\mathbf{K}_{\text{P}} = \text{diag}\,(\ K_{\text{PX}},\ K_{\text{PY}},\ K_{\text{PN}}\ ) \tag{5.4–12a}$$

$$\mathbf{K}_{\text{I}} = \text{diag}\,(\ K_{\text{IX}},\ K_{\text{IY}},\ K_{\text{IN}}\ ) \tag{5.4–12b}$$

$$\mathbf{K}_{\text{D}} = \text{diag}\,(\ K_{\text{DX}},\ K_{\text{DY}},\ K_{\text{DN}}\ )\,. \tag{5.4–12c}$$

Here $\mathbf{K}_{\text{P}}, \mathbf{K}_{\text{I}}, \mathbf{K}_{\text{D}} \in \mathbb{R}^{3\times 3}$ are positive diagonal gain matrices for the proportional, the integral, and the derivative term, respectively. Notice that the controller uses velocities $\mathbf{v}$ instead of the time derivative of the error ($\mathrm{d}\mathbf{e}/\mathrm{d}t$). This is to reduce derivative kicks (unnecessary jumps) that could happen in the positioning operation due to the changes in the setpoint.

Once the required forces $\mathbf{f}_{\text{req}}$ are determined by the controller in equation (5.4–11), they are sent to the control forces allocation in equation (5.3–6) to obtain the necessary control command vector $\mathbf{u}$. This $\mathbf{u}$ is then sent to the corresponding ship's actuators.

5.5 Guidance: Selecting the Desired Pose

Now it is a matter of how to select the desired pose $\boldsymbol{\eta}_{\text{d,0}}$. Recall from subsection 1.1.2 that positioning is a path following operation carried out at a low speed, which is exactly one of the two objectives for the VTPS. In this study, several reference trajectories were generated via the Off-TP [10, 14]. However, since the VTPS is intended for a positioning operation instead of a trajectory tracking operation (section 4.2.1), here we only consider the time histories for the pose. Be advised that the term *reference path* also refers to the reference pose. Moreover, the terms *reference* and *desired* are equivalent and will be used interchangeably.

Let $\tilde{\boldsymbol{\eta}}_{\text{d,0}}$ denote the reference pose expressed in the earth-fixed coordinate system. Let $\mathbf{P} \in \mathbb{R}^{q\times 3}$ be an array whose rows are the time-ordered reference pose $\tilde{\boldsymbol{\eta}}_{\text{d,0}}$ extracted from a reference trajectory as follows:

$$\mathbf{P} = \begin{bmatrix} \tilde{\boldsymbol{\eta}}^{1}_{\text{d,0}} & \tilde{\boldsymbol{\eta}}^{2}_{\text{d,0}} & \dots & \tilde{\boldsymbol{\eta}}^{q}_{\text{d,0}} \end{bmatrix}^{\top}. \tag{5.5–1}$$

Here q is the number of the discretizations of the reference trajectory with respect to time (the solution from the Off-TP). Each of these discretizations is called a waypoint (see section 1.1.2). Thus, array $\mathbf{P}$ makes up a database of q waypoints with the corresponding desired pose assigned at each waypoint. One can say that each waypoint defines a location in the earth-fixed coordinate system and is assigned with a yaw angle.

Let $k \in \mathcal{K}$ denote the index of the waypoints, where $\mathcal{K}$ is a list that contains the indices from 1 to q. Let j be the index of the nearest waypoint from the instantaneous position of the ship. This index j can be determined by solving the following problem:

$$j = \underset{k\in\mathcal{K}}{\arg\min}\ L_2\,(k, \boldsymbol{\eta})\,, \tag{5.5–2}$$

where the function L_2 is a weighted norm as follows:

$$\begin{aligned} L_2\left(k, \boldsymbol{\eta}\right) &= \|\boldsymbol{\eta} - \tilde{\boldsymbol{\eta}}^k_{\mathrm{d},0}\|_{\mathbf{W}} \\ &= \left(\boldsymbol{\eta} - \tilde{\boldsymbol{\eta}}^k_{\mathrm{d},0}\right)^\top \mathbf{W} \left(\boldsymbol{\eta} - \tilde{\boldsymbol{\eta}}^k_{\mathrm{d},0}\right). \end{aligned} \tag{5.5–3}$$

Here $\mathbf{W} \in \mathbb{R}^{3\times 3}$ is a positive diagonal weight matrix as follows:

$$\mathbf{W} = \mathrm{diag}\left(\ 1,\ 1,\ 0\ \right), \tag{5.5–4}$$

such that only the positions are considered when selecting the nearest waypoint. It is worth noting that when selecting a new index of the nearest waypoint j, the indices that were used as the nearest waypoints in the past were removed from the list of indices $\mathcal{K}$. This makes sure that the ship does not try to reverse, which is what will happen if the guidance selects the past waypoints.

Finally, having known the index of the nearest waypoint from the current position, the desired waypoint, i.e., the desired pose $\boldsymbol{\eta}_{\mathrm{d},0}$ is selected from the database $\mathbf{P}$ as follows:

$$\boldsymbol{\eta}_{\mathrm{d},0} \longleftarrow \tilde{\boldsymbol{\eta}}^i_{\mathrm{d},0}, \tag{5.5–5}$$

where i is the index of the desired waypoint that is Δk steps ahead from the index of the nearest waypoint j, or

$$i = j + \Delta k. \tag{5.5–6}$$

In other words, take the i-th row of the array $\mathbf{P}$ to be transformed as the setpoints in the controller equation (5.4–11). This Δk is called the *step range*. It should be chosen according to the shape of the reference path (simplified from the reference trajectory). As a rule of thumb, a large Δk is good for a straight path while a small Δk is suitable for a curved path. It is very important to note that Δk does not explicitly specify the geometrical distance to the desired waypoint. This is due to the fact that the waypoints are the discretizations of the reference trajectory with respect to time, not distance.

5.6 Implementation of the VTPS: Experiment Setup

We are now ready to describe the implementation of the system. The preliminary design of VTPS consists of four fundamental modules as follows (see figure 5-7):

- **Full-state measurement module**. This module comprises multiple high-precision GNSS (global navigation satellite system) that measure the ship's position (x_0, y_0) and velocities (u, v), and a fiber optic gyroscope that measures the yaw angle ψ and its rate r. This module is reinforced with selectable linear filters to suppress the noises from the sensors to some extent. These sensors are summarized in table 5-2.
- **Guidance module: desired pose selection**. This module is responsible for deciding which next waypoints from the database of waypoints $\mathbf{P}$ should be selected to be the desired pose $\boldsymbol{\eta}_{\mathrm{d},0}$. This $\boldsymbol{\eta}_{\mathrm{d},0}$ is then transformed to the deviation $\mathbf{e}$ (equation (5.4–8)) and sent to the controller module. This is explained in detail in section 5.5.
- **Controller module**. This module implements the traditional decoupled PID controller equation (5.4–11) that regulates the required forces $\mathbf{f}_{\mathrm{req}}$ based on the given $\mathbf{e}$ from the guidance module. This is described in detail in section 5.4.

- **Control forces allocation module**. This simple and compact module distributes the required forces $\mathbf{f}_{\text{req}}$ to all available actuators and maps them to the corresponding control commands $\mathbf{u}$. This $\mathbf{u}$ is then sent to the ship. This is discussed in section 5.3.

The above four modules are integrated into a ROS (robotic operating system) and MATLAB Simulink environment which will be explained in the following passages.

Table 5-2. State measurements on the VTShip.

States	Onboard sensors
x_0 (m) y_0 (m) u (m/s) v (m/s)	3 GNSS units, Magellan Systems Japan, Inc. MJ-3021-GM4-QZS-EVK and MJ-3008-GM4-QZS
ψ (rad) r (rad/s)	Fiber optic gyro, Japan Aviation Electronics JG-35FD

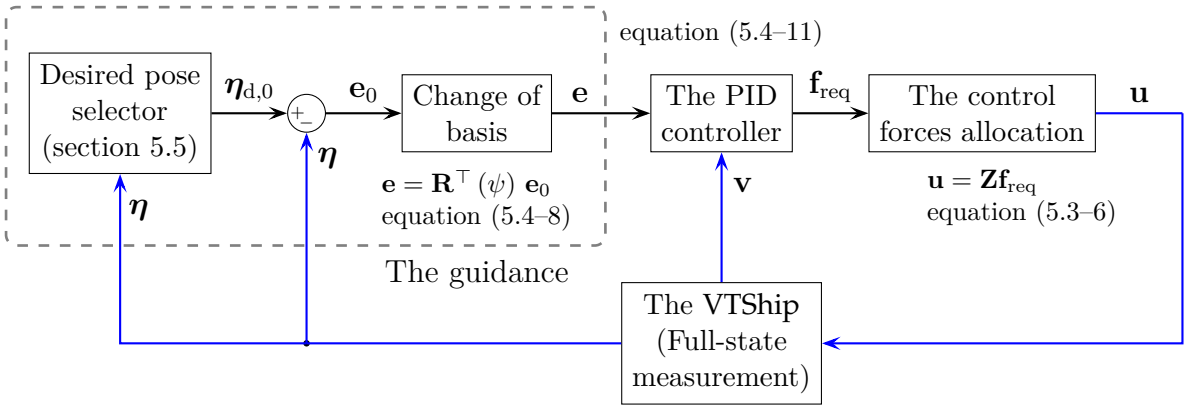


Figure 5-7. Block diagram of the VTPS. Blue lines denote wireless communication.

The VTShip is equipped with a computer that runs on Ubuntu with ROS installed in it. It acts as the master node in the ROS environment. Moreover, the fundamental modules of the VTPS are installed in a land computer as a MATLAB Simulink node with the exception that the full-state measurement module is installed in the VTShip. This full-state measurement in the VTPS publishes the measurement from the onboard sensors to the land computer via a wireless connection. Based on the received state measurement, the remaining modules of the VTPS synthesize the necessary control commands and send them to the VTShip via a wireless connection. This is summarized in figure 5-7 that describes the implementation of VTPS with the VTShip.

Together, the VTShip and the MATLAB Simulink in the land computer create a ROS network that provides communication and interaction between the onboard sensors, the actuators, and the controller. The update frequency of this communication is 10 Hz, i.e., updated every 0.1 s.

5.7 Combined Automatic Docking and Position/Station keeping Experiments

This section is dedicated to discussing the performance of the VTPS in several experiments with the VTShip. These experiments are intended to test hypothesis **H.V** directly and to show that the VTPS is a working prototype (implementable design). Recall from section 5.1.2, the VTPS is intended for operations in a calm harbor area. Thus, these experiments were done when the wind in the experiment pond was calm with no/weak gusts. In a single run, two experiments were conducted in a continuous sequence as follows:

1. Automatic docking operation that can be divided into two stages: transition stage and docking stage. The transition stage is when the ship moves from the open sea to a *receiving point* at the harbor's entrance (the initial location in section 4.2.1). Maneuver from this receiving point to the final docking point is the docking stage.
2. Position/station keeping operation, that is, when the ship has to stabilize its pose at the final docking point.

The following subsections explain each of the experiments in more detail.

5.7.1 Automatic Docking Experiments: Positioning Operation

At the beginning, the ship is manually controlled to an initial point. At this point, the command to start the experiment is sent to the ship. At the same time, an array $\mathbf{P}$ containing the reference pose (database of waypoints) is selected based on the ship's initial point[19] (see figure 5-24 in appendix V.B). This array $\mathbf{P}$ (5.5–1) contains a sequence of waypoints that defines a path with a yaw angle assigned at each waypoint.

In the first stage: the transition stage, the VTPS is not activated. Instead, the rudders are controlled simultaneously by a simple PID controller with line-of-sight (LOS) guidance such that the ship is at the desired path. In this stage, the propeller revolution number is constant in forward mode and the rudders are not constrained (allowed to move at their full range). This is summarized in table 5-3.

Table 5-3. Ranges of the actuators in the transition stage: LOS guidance + PID rudder controller.

Actuator	Range
Port rudder (deg)	$\delta_{\mathrm{p}} \in [-105, 35]$
Starboard rudder (deg)	$\delta_{\mathrm{s}} \in [-35, 105]$
Bow thruster (rps)	$n_{\mathrm{B}} = 0$ (Off)
Propeller (rps)	$n = 10$ or $n = 7$

The ship proceeds to the next stage: the docking stage when it is in the proximity of a predefined switching point, i.e., the receiving point. In this stage, the VTPS is activated.

[19]Prior to the experiments, at least forty reference trajectories were prepared via the Off-TP [10, 14]. Each of these trajectories has a distinct initial point distributed on the pond. The algorithm in the VTPS selects one that has an initial location closest to the ship's actual location.

The VTPS is responsible to synthesize the required control commands to slow the ship down such that it follows the remaining path until finally stops at the final docking point. The propeller revolution number is still constant in forward mode. On the other hand, the rudders are limited based on assumption **A.5-4** established in appendix V.A. The summary of the ranges of the actuators in this docking stage is presented in table 5-4.

The PID gains were determined from extensive trials with the VTShip at the experiment pond. The selected gains are those that balance the performance in both the docking stage and the position/station keeping operation. The proportional gains led to excessive oscillations in the position/station keeping operation but were very helpful in the docking stage. On the other hand, the derivative gains were very crucial in the position/station keeping stage but reduced the performance in the docking stage. The integral gains were set very low, considering that the gusts at the pond blew within a very short period. In contrast to the gains, the saturation values of the required forces $\mathbf{f}_{\text{req}}$ were determined based on the maximum values of these forces within the allowed ranges of the actuators (assumption **A.5-4**). The PID gains and the saturation values of the required forces used in the VTPS are shown in tables 5-5 and 5-6, respectively.

Table 5-4. Ranges of the actuators for the VTPS in the docking stage and the position/station keeping experiment.

Actuator	Range
Port rudder (deg)	$\delta_{\text{p}} \in [-105, -60]$
Starboard rudder (deg)	$\delta_{\text{s}} \in [60, 105]$
Rudder rate (deg/s)	≈ 23
Bow thruster (rps)	$n_{\text{B}} \in [-27, 27]$
Propeller (rps)	$n = 10$

Table 5-5. PID gains in the VTPS (docking stage and position/station keeping experiment).

Terms	Values
Proportional	$\mathbf{K}_{\text{P}} = \text{diag}\,(\ 4,\ 4,\ 4\ )$
Integral	$\mathbf{K}_{\text{I}} = \text{diag}\,(\ 0.01,\ 0.01,\ 0.001\ )$
Derivative	$\mathbf{K}_{\text{D}} = \text{diag}\,(\ 25,\ 25,\ 30\ )$

Table 5-6. Saturation values for the required forces $\mathbf{f}_{\text{req}}$ from the PID controller in the VTPS (docking stage and position/station keeping experiment).

Required forces	Range
Surge (N)	$-1.5 \leq X_{\text{req}} \leq 0.8$
Sway (N)	$-1.0 \leq Y_{\text{req}} \leq 1.0$
Yaw moment (N $\cdot$ m)	$-1.7 \leq N_{\text{req}} \leq 1.5$

5.7.2 Position/Station keeping Experiments

Once the ship arrives and stops at the final docking point, the position/station keeping operation starts. The VTPS has to maintain and stabilize the pose of the ship, that is, the desired pose is constant. This desired pose is the last row of the array $\mathbf{P}$ (see section 5.5), i.e., $\boldsymbol{\eta}_{\mathrm{d},0} = \tilde{\boldsymbol{\eta}}^{q}_{\mathrm{d},0}$ until the end of the experiment (that lasts for an extended period). The ship must remain stationary at a constant propeller revolution number in forward mode. When the gust is strong, the VTPS should be able to return the ship back to the desired pose. In this experiment, the VTPS still uses the same parameters and settings as in the docking stage (see tables 5-4 to 5-6).

5.7.3 Results of the Experiments

In total, there were at least three experiments conducted to test the VTPS. These experiments were conducted during the period between March 2022 and April 2022. Each of these experiments is referred to as experiment A, experiment B, and experiment C. In experiment A, the propeller revolution number was 10 rps in forward mode at all times. In experiment B, the propeller revolution number was set to a lower value: 7 rps in the transition stage, and increased to 10 rps for the rest of the experiment; all in forward mode. The reason for this treatment will be explained later in the dedicated sub-subsection. Lastly, experiment C was carried out to test the performance of VTPS for a different type of path: 180 deg/U-turn docking operation. In experiment C, the propeller revolution was 10 rps in forward mode all the time. Note that experiment C is dedicated only to the docking stage (positioning operation). Moreover, the step range Δk was 25 in experiment A and experiment B, while in experiment C was 5. These values were determined based on several trials with the VTShip.

5.7.3.1 Experiment A

Experiment A was conducted when the wind was quite calm with 0.3 m/s to 2.0 m/s gusts. The visualization of the experiment and the time histories of the control commands are shown in figure 5-8. Meanwhile, the time histories of the states are shown in figure 5-9. In the figures, each of the stages is shaded with a different color. For example, the transition stage, the docking stage, and the position/station keeping stage are shaded with purple, cyan, and yellow, respectively.

In the first stage (shaded in purple), the VTShip was accelerating (u was increasing) when the VTShip was at the desired path. This was because the rudders were zero when there is no deviation from the path. At a constant propeller revolution number n, the acceleration was fast as shown in figure 5-9. The VTShip was accelerating from $u \approx 0.10$ m/s to $u \approx 0.37$ m/s within about two minutes.

When the VTShip arrived at the switching (receiving) point, the VTPS was activated; thus the docking stage began. This stage is shaded with cyan. One can observe that the VTPS performed really well in decelerating the VTShip from $u \approx 0.37$ m/s to $u \approx 0.03$ m/s in a very short time: 40 s; equivalent to a crash-stopping operation. This shows the excellent performance of VecTwin rudder in executing a crash-stopping operation at a constant propeller revolution number in forward mode as reported in [79].

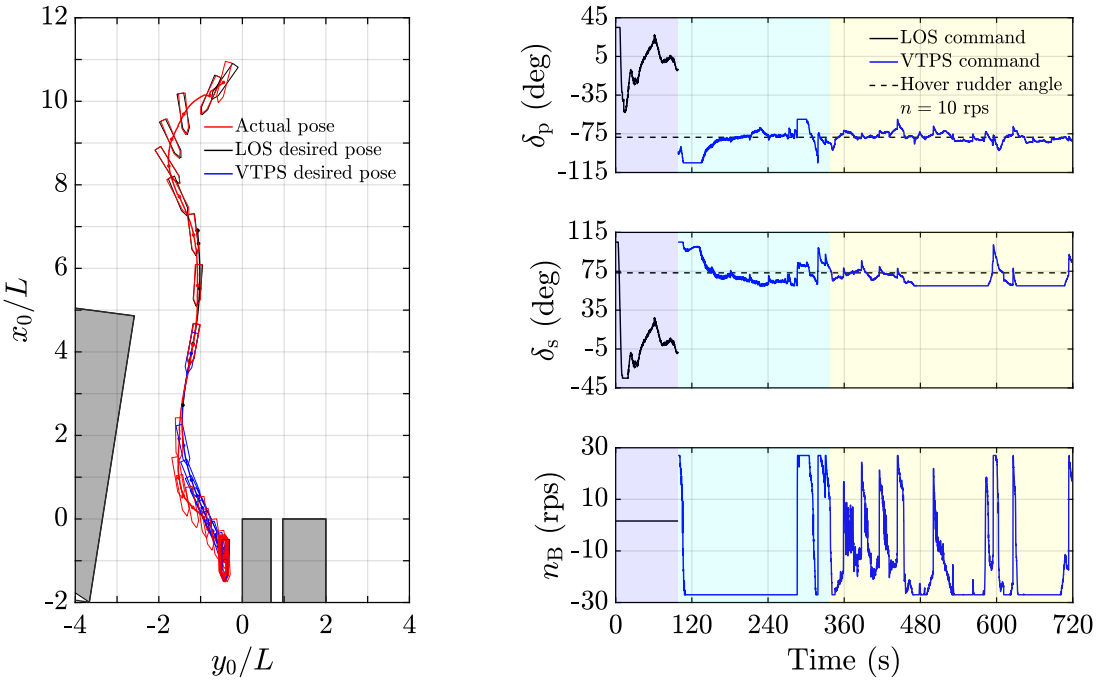


Figure 5-8. Visualization and time histories of the control commands in experiment A.

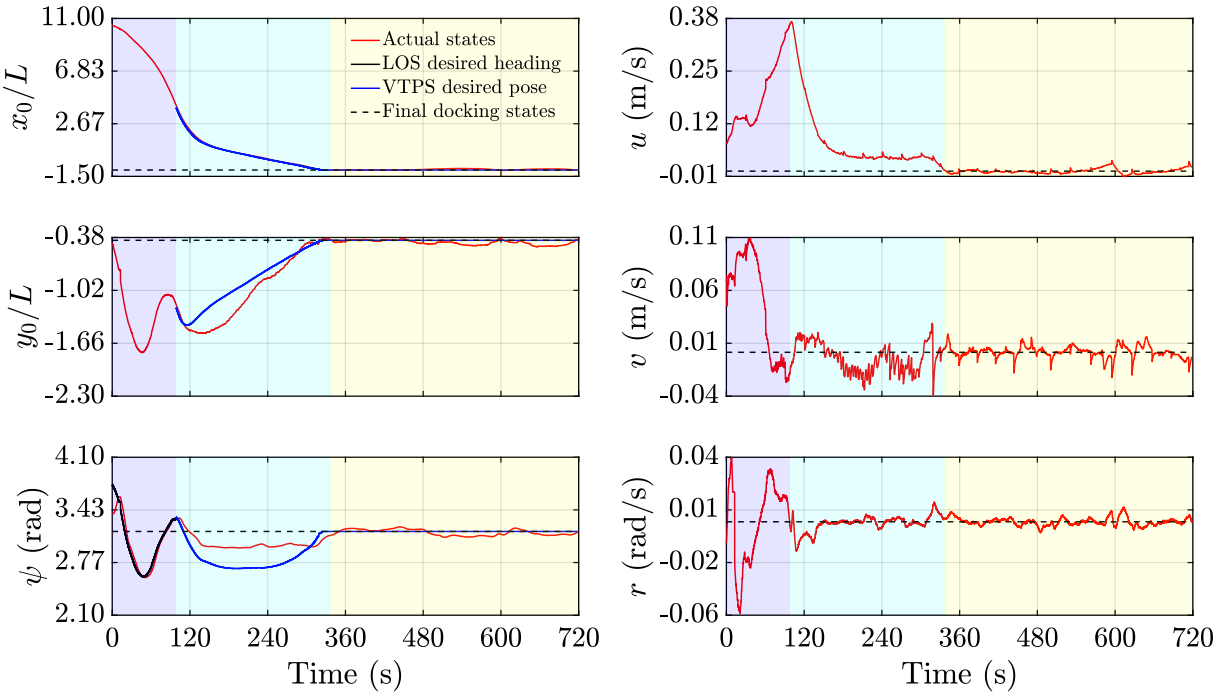


Figure 5-9. Time histories of the states of the VTShip in experiment A.

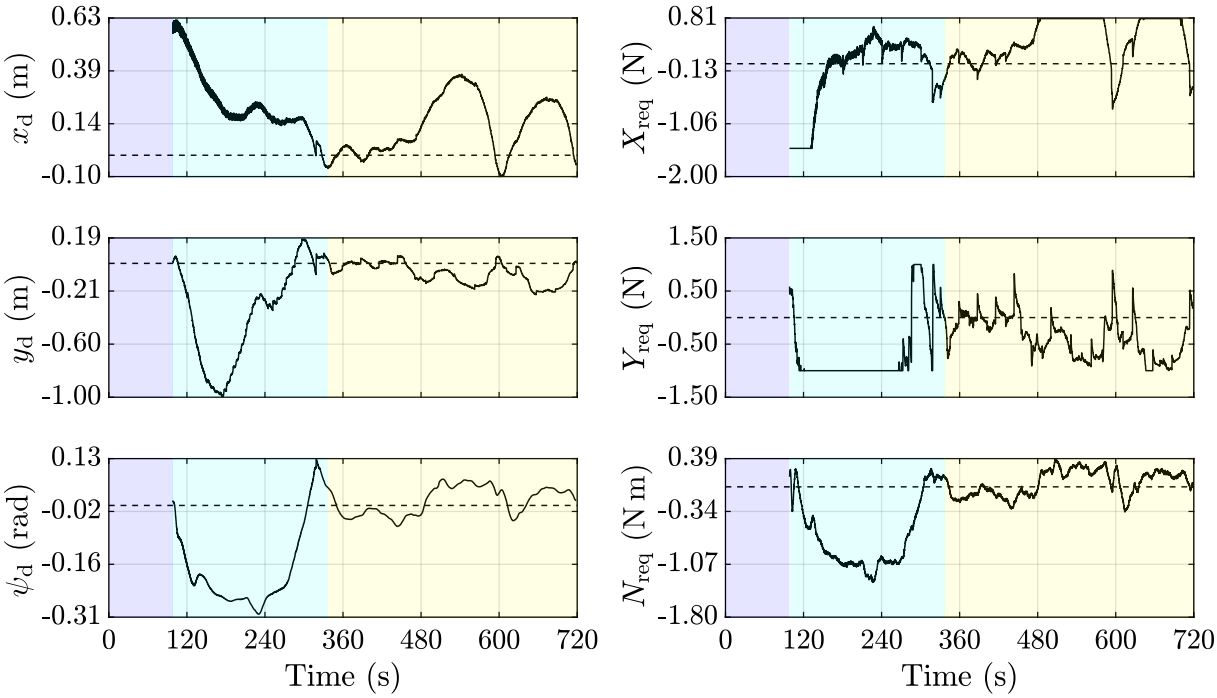


Figure 5-10. Desired pose in the ship-fixed coordinate system: the transformed error $\mathbf{e}$, and the required forces $\mathbf{f}_{req}$ in experiment A.

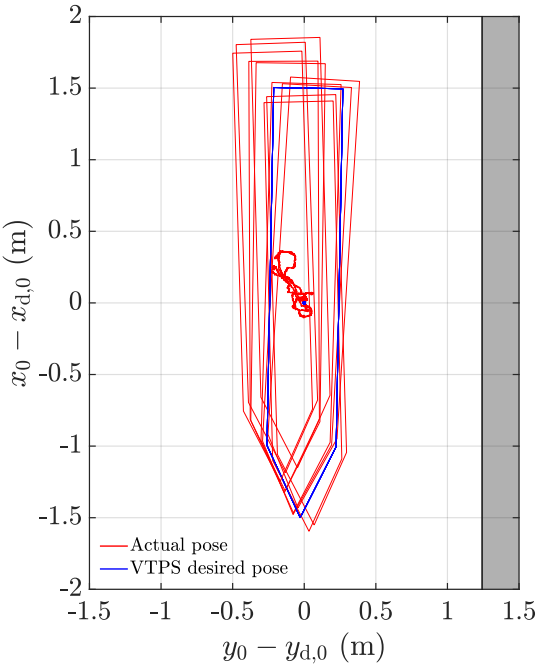


Figure 5-11. Deviation of the pose of $\boldsymbol{\eta}$ from the desired pose $\boldsymbol{\eta}_{d,0}$ during position/station keeping experiment in experiment A.

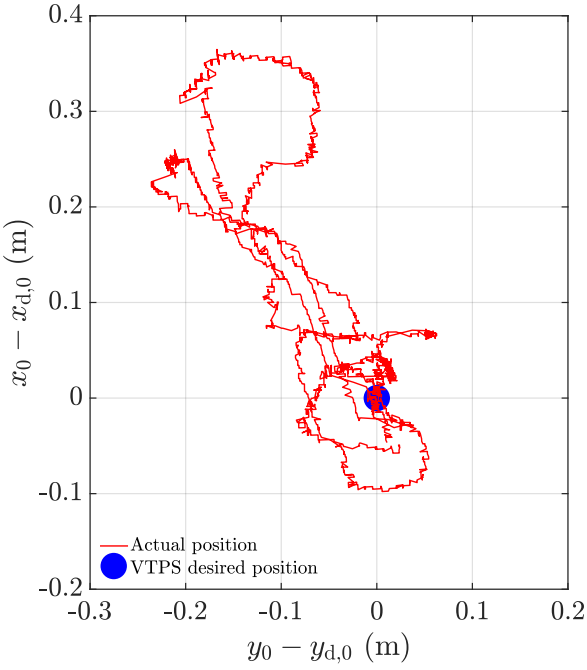


Figure 5-12. Deviation of the position (x_0, y_0) from the desired position $(x_{d,0}, y_{d,0})$ during position/station keeping experiment in experiment A.

Observe that during the crash-stopping operation (or sudden/rapid deceleration in general), the VTShip deviated from the desired path. Nonetheless, the VTPS was able to return the VTShip back to the desired pose. Continuing, the VTPS synthesized the necessary control commands such that the ship was at the remaining desired poses and finally stopped at the final docking point. At this point, the VTShip was almost stopped: at very small velocities, i.e., $u, v \approx 0\,\mathrm{m/s}$ and $r \approx 0\,\mathrm{rad/s}$.

The almost-zero condition of the velocities marked the beginning of the position/station keeping experiment; shaded with red. The VTPS gave satisfactory performances in stabilizing the deviation/error of the pose of the VTPS at the final docking point for about six minutes straight. This is shown in figure 5-9 where the VTShip's states (solid red lines) are stabilized around the desired values (dashed black lines). This can also be observed from figure 5-10 that shows the deviation of the pose in the ship-fixed coordinate system $\mathbf{e} = \begin{bmatrix} x_\mathrm{d} & y_\mathrm{d} & \psi_\mathrm{d} \end{bmatrix}^\top$ and the corresponding required forces $\mathbf{f}_\mathrm{req} = \begin{bmatrix} X_\mathrm{req} & Y_\mathrm{req} & N_\mathrm{req} \end{bmatrix}^\top$ (regulated by the PID controller in equation (5.4–11)).

More clearly, figures 5-11 and 5-12 show the deviation of the pose $\boldsymbol{\eta}$ from the desired final docking pose $\boldsymbol{\eta}_{\mathrm{d},0}$ in the earth-fixed coordinate system ($\mathbf{e}_0$) during the position/station keeping experiment. The gusts during this experiment tend to push the VTShip to the stern-starboard direction. Nevertheless, the VTPS performed well to stabilize the ship back to the desired final docking pose.

5.7.3.2 Experiment B

Experiment B was conducted at the pond when the wind was calm with 0.1 m/s to 2.0 m/s gusts. In the experiment, the propeller revolution number in the first stage was reduced to 7 rps instead of 10 rps. It was returned back to 10 rps in the remaining stages (see table 5-4). Moreover, the switching point was moved farther away from the final docking point. This resulted in the first stage (transition stage) that was shorter than that in experiment A. Doing this also gave more time to see the VTPS in action during the docking stage.

The motivation behind the above treatments is to reduce the unnecessary acceleration that is likely to happen in the transition stage (see figure 5-9 of experiment A). These treatments are also expected to minimize the deviation from the desired path that was observed in experiment A when the VTShip was decelerating (when switching to the docking stage, see figure 5-8).

The summary of experiment B was visualized and presented in figures 5-13 to 5-15. As one can observe in figure 5-14, as expected, the acceleration in the transition stage was minimal. The surge velocity u was at around 0.30 m/s during the short period of this stage (shaded with purple). When entering the docking stage (shaded with cyan), the surge velocity was also gradually decreasing. More importantly, the tendency of the VTShip to rotate when decelerating was attenuated. Thus, as shown in figure 5-8, the deviation from the desired path was also minimized that the VTShip was almost always at the desired path during the docking stage.

When approaching the final docking point (the end of the docking stage), the VTShip overshot for about 1.15 m; approximately $2L/5$. Nevertheless, the VTPS managed to control the ship to reverse slowly until finally stopped it at the final docking point. This overshoot is directly related to the guidance module (desired pose selection in section 5.5). This issue will be dealt with and improved in the future.

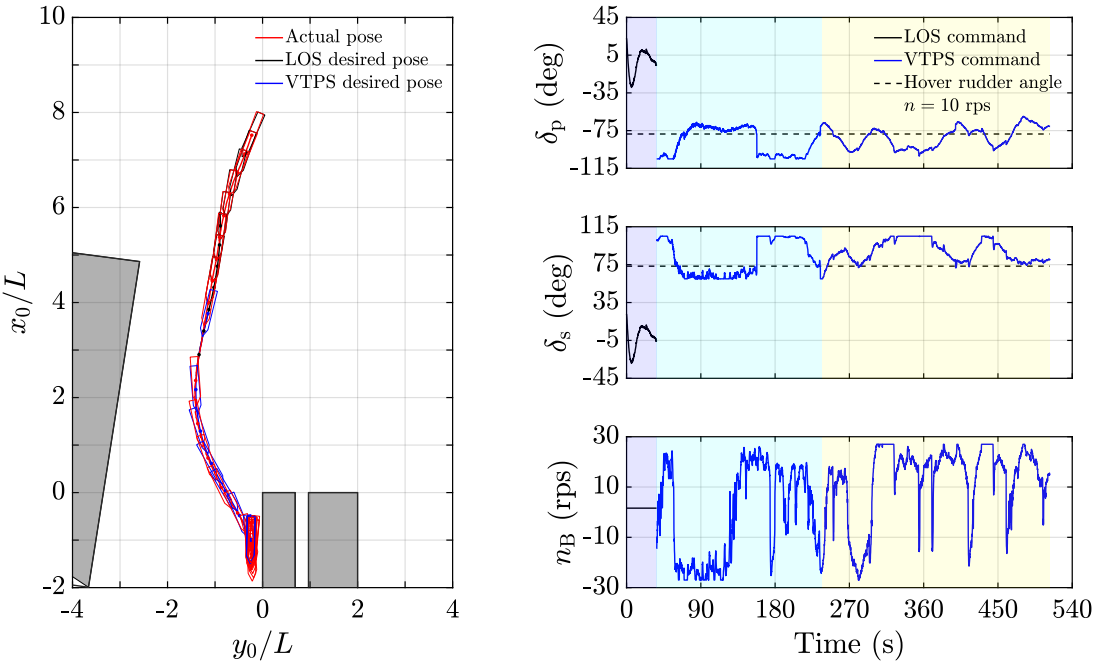


Figure 5-13. Visualization and time histories of the control commands in experiment B.

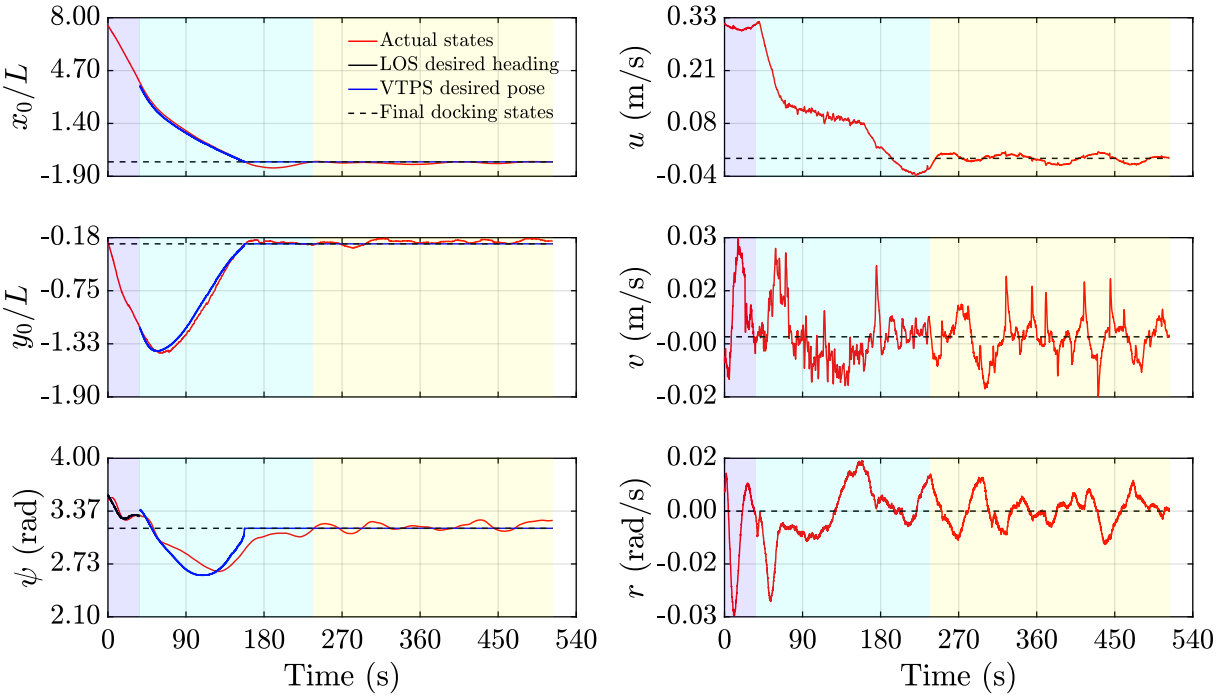


Figure 5-14. Time histories of the states of the VTShip in experiment B.

大阪大学 OSAKA UNIVERSITY

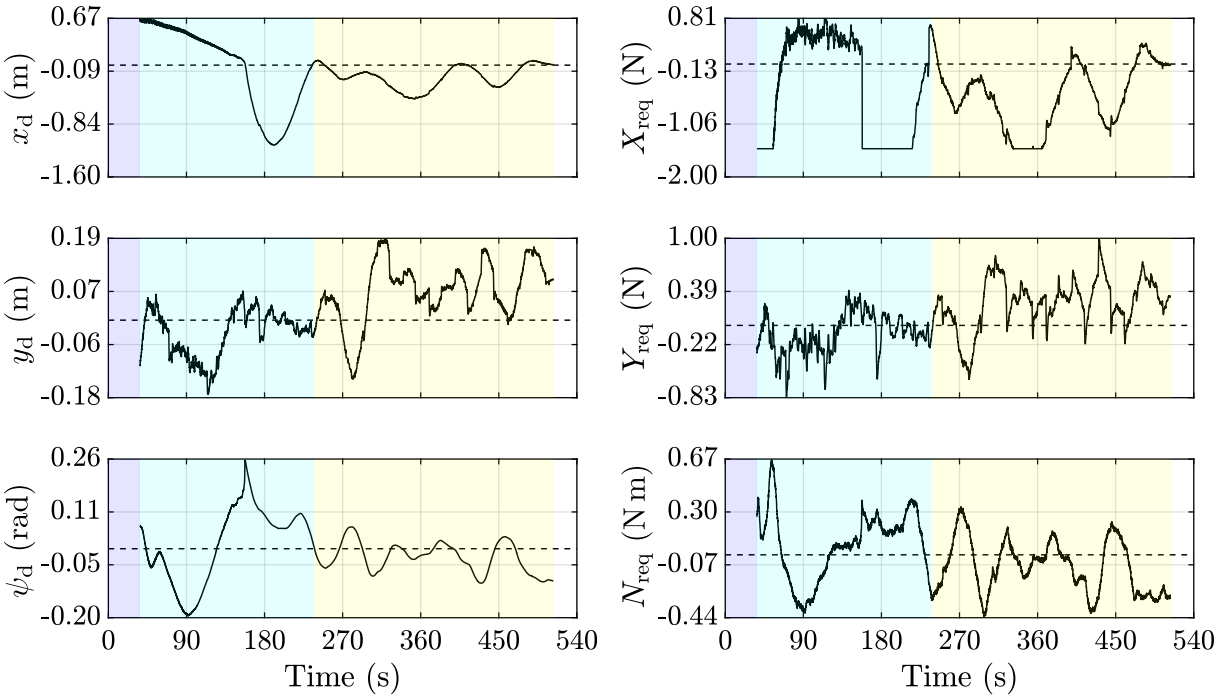


Figure 5-15. Desired pose in the ship-fixed coordinate system: the transformed error $\mathbf{e}$, and the required forces $\mathbf{f}_{\text{req}}$ in experiment B.

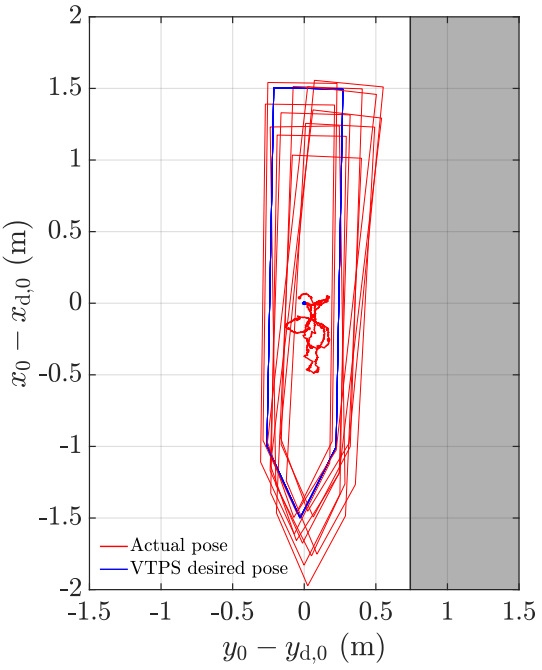


Figure 5-16. Deviation of the pose of $\boldsymbol{\eta}$ from the desired pose $\boldsymbol{\eta}_{d,0}$ during position/station keeping experiment in experiment B.

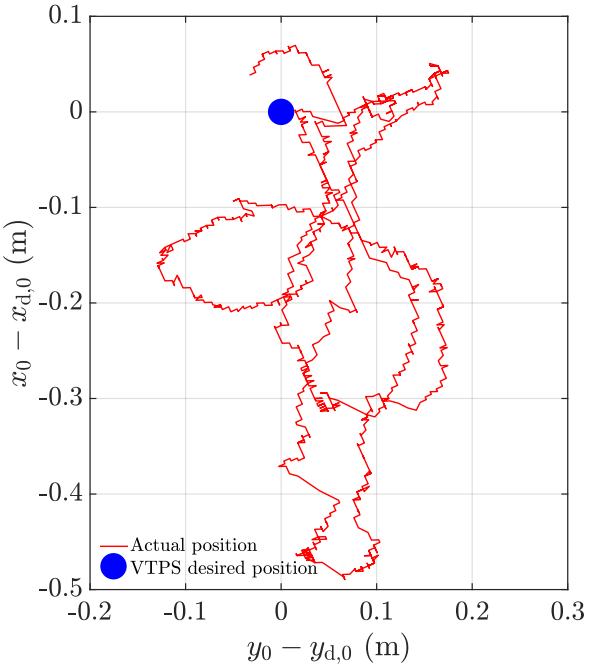


Figure 5-17. Deviation of the position (x_0, y_0) from the desired position $(x_{d,0}, y_{d,0})$ during position/station keeping experiment in experiment B.

In the position/station keeping experiment, the VTPS consistently gave a satisfactory performance in stabilizing the pose $\boldsymbol{\eta}$ at the final docking point for about five minutes straight. This can be seen in figure 5-9 (shaded with yellow) and especially figure 5-15 that presents the time histories of the deviation/transformed error $\mathbf{e}$ in the ship-fixed coordinate system. The position/station keeping experiment is more clearly visualized in figures 5-16 and 5-17.

5.7.3.3 Experiment C

As mentioned earlier at the beginning of section 5.7.3, the purpose of experiment C is to test the performance of the VTPS for a 180 deg/U-turn docking path. Thus, only relevant results of the docking stage are presented and discussed here. In this experiment, the wind condition at the pond was calm with 0.1 m/s to 2.0 m/s gusts. The visualization of this experiment is presented in figure 5-18. On the other hand, the time histories of the states and the transformed deviation $\mathbf{e}$ are shown in figures 5-19 and 5-20, respectively. One can see that the VTPS proved itself able to synthesize the necessary control commands such that the VTShip followed the desired U-turn docking poses.

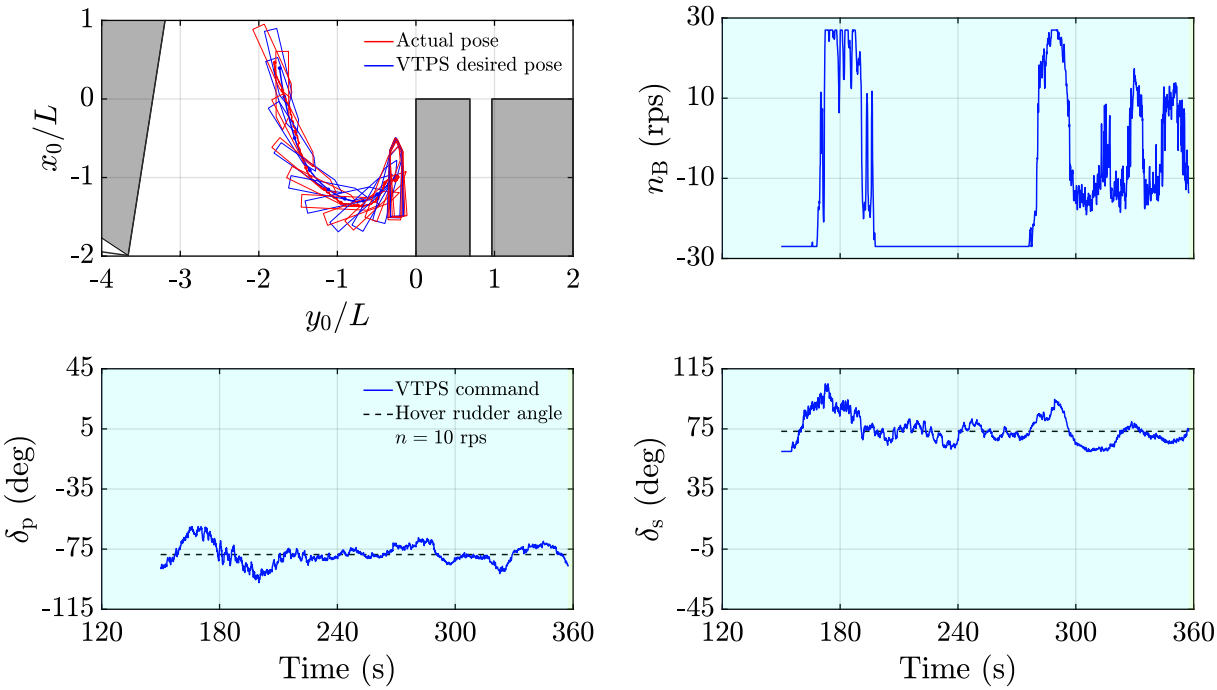


Figure 5-18. Visualization and time histories of the control commands in experiment C.

Notice that during the turning, the surge velocity u was kept very low; almost equal to the sway velocity v as observed in figure 5-19. This made the VTShip vulnerable to gusts that could have easily drifted it from the desired path. One can also observe from figure 5-18 that the bow thruster was used heavily to turn/rotate the ship while the rudders were used solely to maintain the low speed. This implies that the current design of VTPS relies primarily on the bow thruster for operations that require a hard turn.

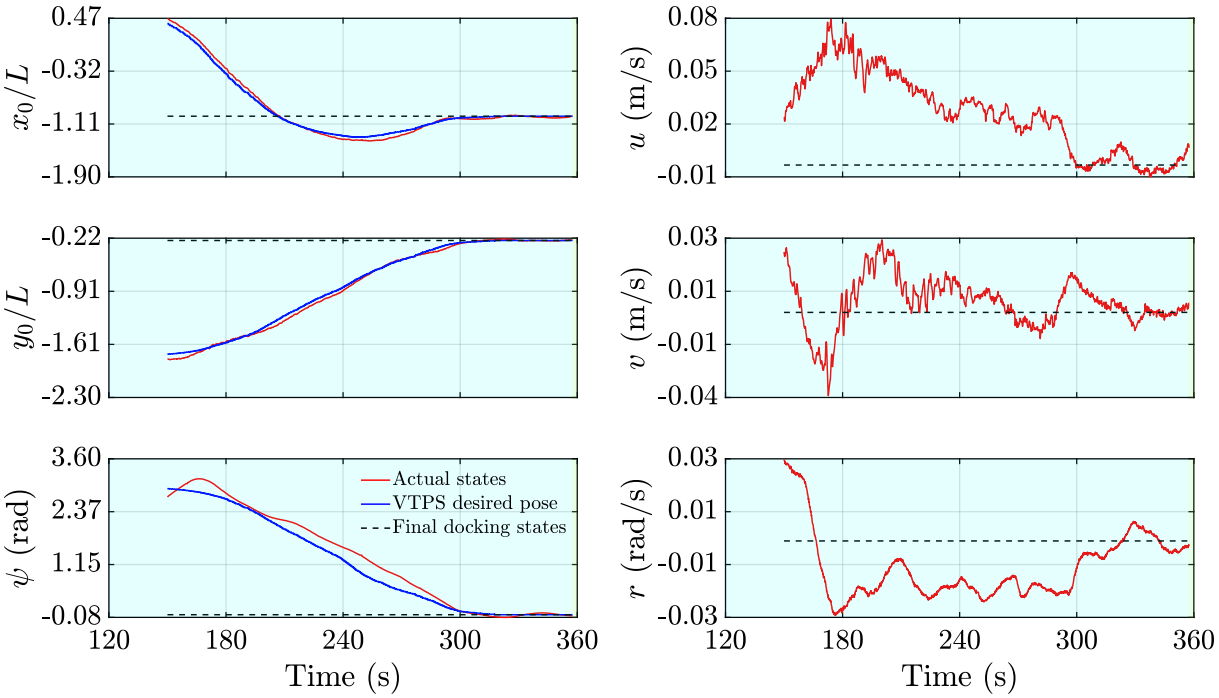


Figure 5-19. Time histories of the states of the VTShip in experiment C.

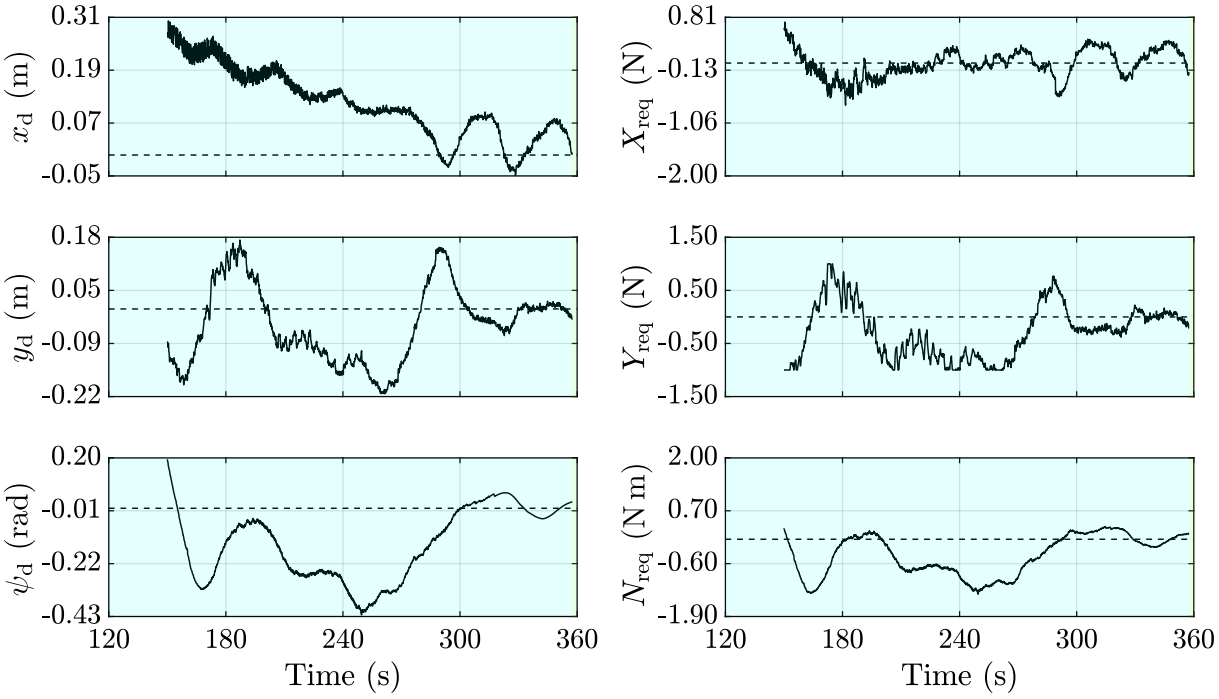


Figure 5-20. Desired pose in the ship-fixed coordinate system: the transformed error $\mathbf{e}$, and the required forces $\mathbf{f}_{\text{req}}$ in experiment C.

There are two possible reasons for this. The first is that the established assumptions **A.5-2** and **A.5-3** do not capture the true capability of the VecTwin rudder. The second is that the range of the rudders is significantly limited to just 60 deg; in contrast to the actual range of 140 deg. It is also worth mentioning that the presence of strong

gusts or other disturbances could have deteriorated the performance, particularly during the turning maneuver at a very low speed. Nonetheless, at the moment, one has observed and can consider that the potential of the VTPS is quite promising.

5.8 Concluding Remark

This chapter introduces a practical approach to design a novel low-speed positioning system that utilizes twin high-performance fishtail rudders: VecTwin rudder, and a bow thruster. This system is called VTPS which is an acronym for VecTwin positioning system. The design is based on a hypothesis that the ability of the VecTwin rudder to make a ship hovers indeed opens up the possibility to design a low-speed positioning system (see hypothesis **H.V**). In the VTPS, the propeller revolution is kept constant in forward mode for all operations; rendering it unavailable as a free/active actuator. This kind of arrangement does not necessarily make a ship fully actuated. Moreover, the commands-forces relationship that is the core of the VTPS was obtained via several CFD simulations of the bollard pull test. This approach is not uncommon when the resources and opportunities to conduct bollard pull tests with the scale model ship are not available. This approach does not require a mathematical model or dynamics of the ship to be fully defined (in contrast to the previous chapters). However, a geometrical model is necessary to perform the CFD simulations.

To test the overall approach and the VTPS itself, several experiments with the VTShip were conducted. These experiments serve as concrete evidence of the practical implementation of the system; demonstrating that the system is a working prototype (in contrast to numerical simulations). In other words, the hypothesis **H.V** is directly verified and validated in the experiments instead of going through numerical simulations. Moreover, the important objective of control, i.e., stabilization of the ship's pose at the desired pose against disturbances was also achieved.

The current design of the VTPS relies on the rudders to actively distribute the constant thrust from the propeller (constant propeller revolution in forward mode) to balance the forces such that the ship hovers. This is an advantage over the usual DPS arrangement where it is the thrusters that actively change their revolution number and mode to balance the forces. The rudders also transfer the thrust to lateral force and moment to support the bow thruster. However, the fact that VTPS relies on the bow thruster in an operation that requires a hard turn, e.g., a U-turn docking, is due to several assumptions imposed on the forces (see assumptions **A.5-1** to **A.5-3**) and the limitation of the rudder range. This is a disadvantage over the usual DPS that can generate stronger lateral forces and turning moments. This will be discussed and improved in the future.

Nevertheless, this arrangement: a VecTwin rudder, a bow thruster, and a single-screw propeller (constant revolution in forward mode) in the current design of the VTPS performed very well for low-speed operations like positioning (path following at a low speed) and position/station keeping, in a calm wind waterway with no/weak gusts. With further improvement, the system can be a potential addition to the usual DPS. In addition, the experiments gave a direct demonstration of the excellent ability of the VecTwin rudder to execute hovering and an emergency crash-stopping operation; all at a constant propeller revolution number in forward mode.

The low-speed positioning system introduced in this chapter is relevant to realize

the idea of a general MASS (all degrees). For example, the shipmaster can always take over and override the VTPS during an emergency. The VTPS is the main product of **contribution III**: to design a simple and practical positioning system that is suitable for low-speed operations in a calm waterway. For reference, this study was presented at a conference and is publicly available as part of the proceeding as **publication [E]**. It is also published as a journal paper in **publication [C]**.

Appendices

V.A On the linearity assumption for a wider range of rudder angle

The linear transformation $\tilde{\mathbf{V}}$ in equation (5.2–13) was obtained by multiple linear regression of the results from nine CFD simulations as shown in table 5-1. These simulations were done with rudder angles $\mathbf{u}_\delta$ between 70 deg and 80 deg. The purpose of these simulations is to linearize the forces around the hover rudder angle $\mathbf{u}_{\delta \mathrm{h}}$ (equation (5.2–10)). This linearity is assumed to be true for a wider range of rudder angle: from 60 deg to 105 deg (assumption **A.5-4**). Let us denote the set of all $\mathbf{u}_\delta$ within this wider range as $\mathcal{R}$ as follows:

$$\mathcal{R} = \left\{ \text{all } \begin{bmatrix} \delta_\mathrm{p} \\ \delta_\mathrm{s} \end{bmatrix} : \delta_\mathrm{p} \in [-105, -60] \text{ and } \delta_\mathrm{s} \in [60, 105] \right\}.$$

This assumption **A.5-4** is verified by performing additional CFD simulations with rudder angles $\mathbf{u}_\delta$ between 40 deg and 105 deg. Let $\mathcal{A}$ be the finite set of these additional $\mathbf{u}_\delta$. One can plot the results from these additional simulations and linearly interpolate the values between two adjacent results to form an interpolated surface. This is visualized in figures 5-21 and 5-22 where the interpolated surface (shown in black) is compared with the regression plane (shown in blue): the image of the linear transformation $\tilde{\mathbf{V}}$ (5.2–13) given the set $\mathcal{R}$ as the domain.

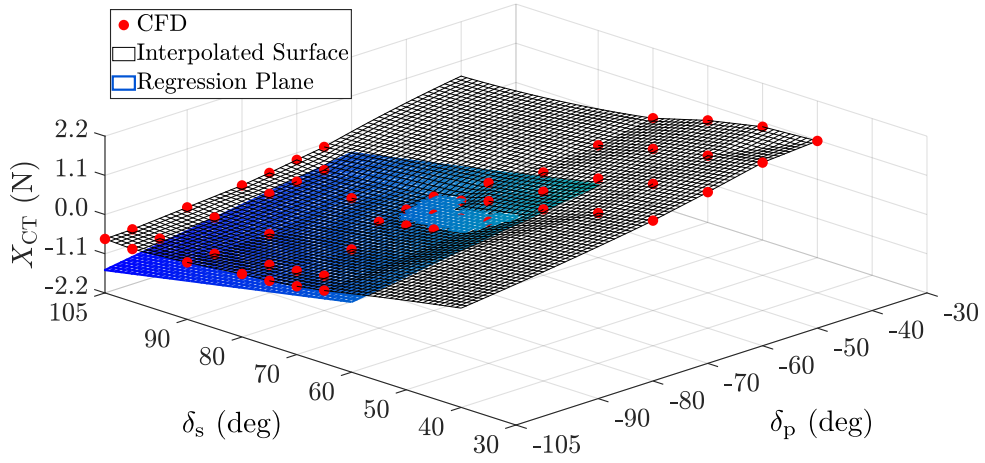


Figure 5-21. Interpolated surface from the results of additional CFD simulations and the regression plane (first row of $\tilde{\mathbf{V}}$ for rudder angle between 60 deg to 105 deg).

One can visually observe from figure 5-21 that the regression plane is a good estimation of X_{CT} for $\mathbf{u}_\delta$ around the hover rudder angle: between 60 deg to 90 deg. The regression overestimates X_{CT} for $\mathbf{u}_\delta$ bigger than 90 deg, i.e., the first row of $\tilde{\mathbf{V}}$ should be smaller in magnitude (smaller slope) within this region. However, conservatively, this is desirable in low-speed operations because it is expected to give aggressive action to regulate the low speed. For $\mathbf{u}_\delta$ smaller than 60 deg, the regression underestimates X_{CT}, i.e.,

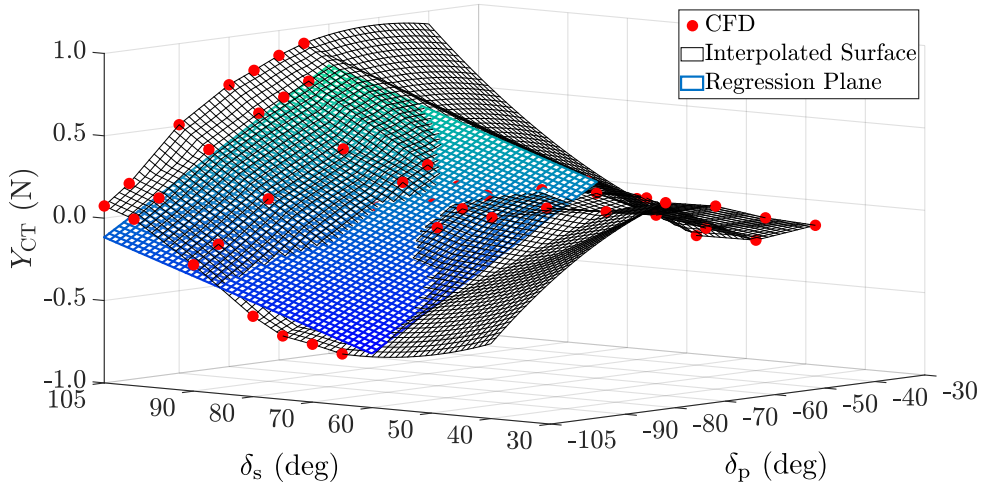


Figure 5-22. Interpolated surface from the results of additional CFD simulations and the regression plane (second row of $\tilde{\mathbf{V}}$ for rudder angle between 60 deg to 105 deg).

the magnitude of the first row of $\tilde{\mathbf{V}}$ should be bigger (steeper slope). This is as expected, because earlier in section 5.2.1.3 it is mentioned that small rudder angles give a total force X_{CT} that causes the ship to accelerate in the surge direction. These observations imply that for operations at normal speed and/or with smaller $\mathbf{u}_\delta$, one or more different $\tilde{\mathbf{V}}$ should be formulated. From figure 5-22, it can be observed that the regression plane does give a good estimation of Y_{CT} for all $\mathbf{u}_\delta \in \mathcal{R}$. One can also see that the slopes of Y_{CT} change sign when $\mathbf{u}_\delta$ is lower than 60 deg.

From these observations, it is safe to say that the linearity assumption is acceptable for all $\mathbf{u}_\delta \in \mathcal{R}$, as verified by the CFD simulations and validated in the experiments (tested in a closed-loop feedback control system). To conclude this appendix, with the additional data points, one might be tempted to obtain a single linear transformation $\tilde{\mathbf{V}}$ that covers the whole range of rudder angle. This is not recommended as the intercept may not be at the hover rudder angle $\mathbf{u}_{\delta\mathrm{h}}$; effectively contradicts the fact that at $\mathbf{u}_{\delta\mathrm{h}}$ the total forces should be zero such that the ship is hovering.

V.B User Interface of the VTPS in Matlab Simulink

During the study, the author developed many user-friendly interfaces for different kinds of scale model experiments in MATLAB Simulink. One of them is the interface for the VTPS. This interface shows the real-time status of the sensors and the state measurements; all summarized in a dashboard as shown in figure 5-23. It is also able to visualize the pose of the ship on the pond (real-time monitor) as shown in figure 5-24. These figures also show the sequence of the experiments.

大阪大学 OSAKA UNIVERSITY

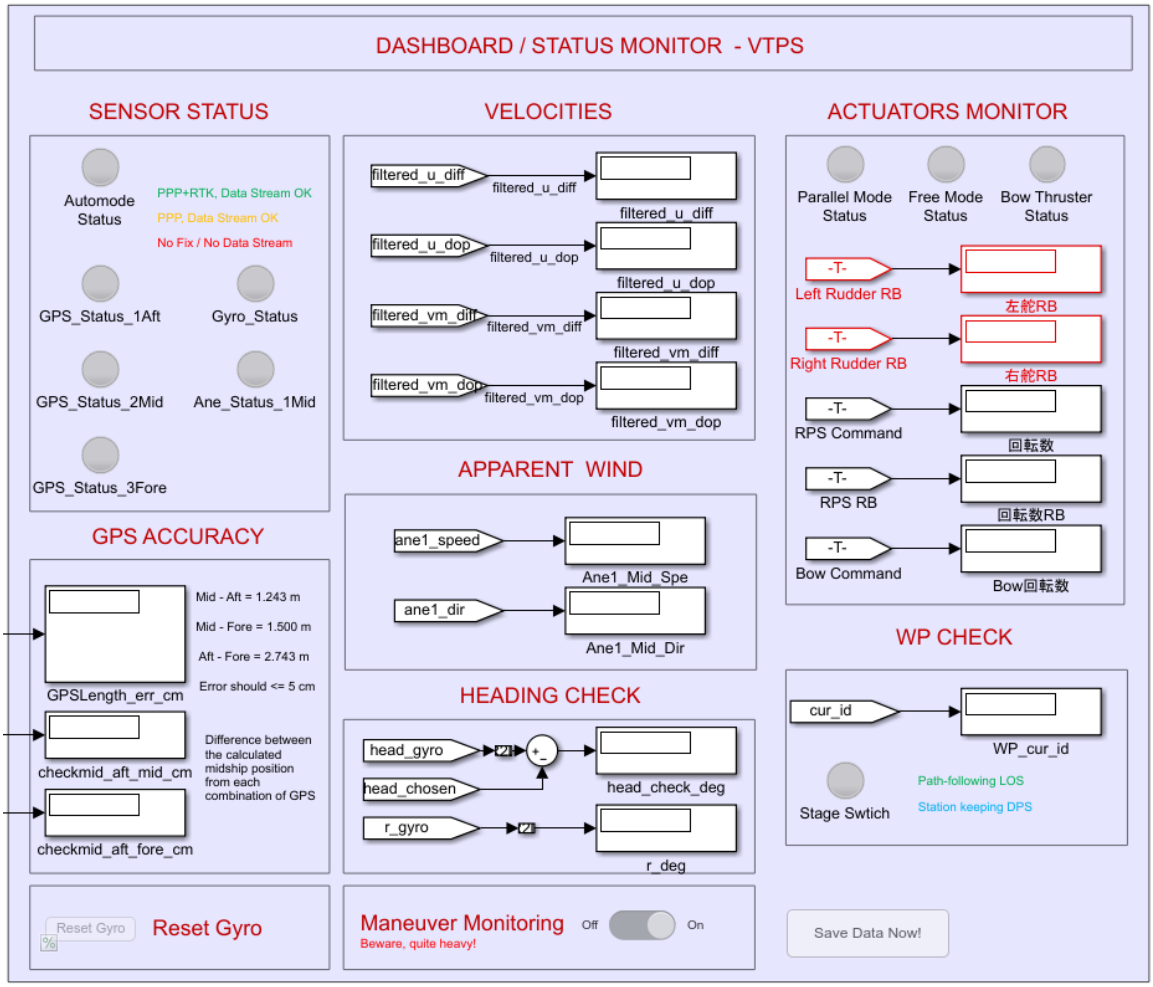
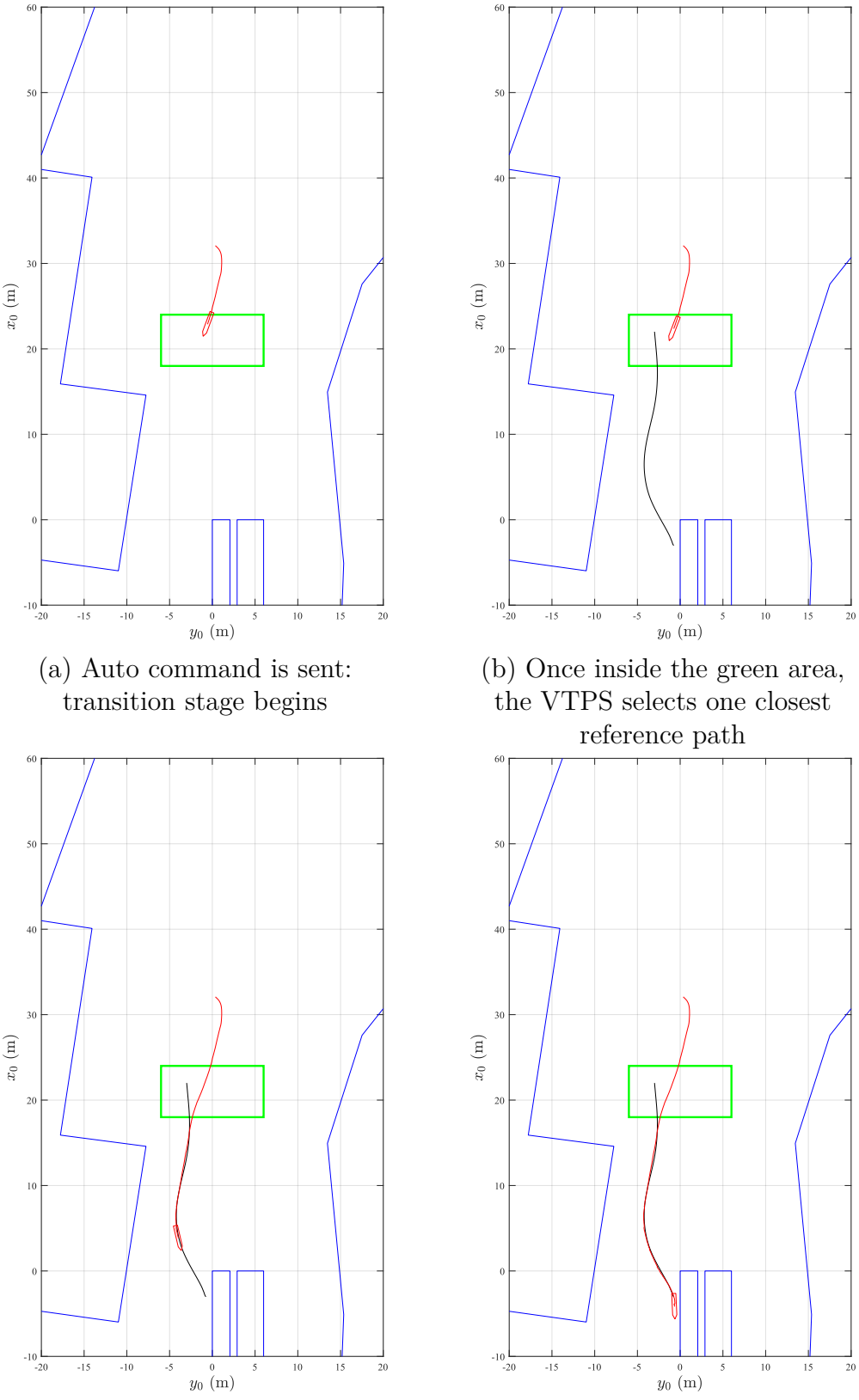


Figure 5-23. Dashboard of the VTPS in MATLAB Simulink: the user interface.



(a) Auto command is sent: transition stage begins

(b) Once inside the green area, the VTPS selects one closest reference path

(c) Real-time monitoring: docking stage

(d) Real-time monitoring: position/station keeping

Figure 5-24. Real-time monitoring of the pose during the stages of the experiments.

VI

General Conclusion

6.1 Summary

This thesis discusses the two practical problems in automatic docking of a MASS. The first problem is how to plan a safe trajectory or a safe path for a docking operation efficiently. The second problem is how to maneuver the ship slowly and carefully according to the planned trajectory or path until finally reaches the desired final docking point. The main contributions of this thesis evolve around these two problems. Each contribution attacks the problems with a different approach that has its own advantages and disadvantages. However, their main objective is the same, to give some insight and different perspectives to realize automatic docking of a ship (particularly a conventional underactuated ship); or in general, towards the realization of a MASS.

On the first problem, this thesis introduces a semionline trajectory planner (SO-TP) that takes into account a predefined geometry of the harbor and the dynamics of the wind at the instant the planner is executed. The purpose of the planner is to modify/replan a new trajectory when the current trajectory is deemed not feasible anymore. This kind of planner somehow lies between a high-level planner and a middle-level planner because it only takes into account the static obstacles but then modifies the trajectory based on the instantaneous wind dynamics and the states of the ship. Such a planner should be fast enough to be implemented in practice.

One of the ways to plan a trajectory is via an optimization where the mathematical model or the dynamics of the ship's maneuvering is needed. However, in most cases, a conventional underactuated ship has an intricate and nonlinear dynamics at a low speed. Coupled with the underactuation and the estimation of the wind dynamics, it increases the difficulty of the optimization problem, hence the long computational time. To circumvent this, the SO-TP is warm-started (using a good initial guess) with an almost-globally optimal solution of an identical problem obtained by a high-level trajectory planner termed as offline trajectory planner (Off-TP). In the simulation, this warm start proved to have a significant effect on reducing the computational time compared to without a warm start (cold-started).

Actually, the preliminary study in **publication [D]** does not consider any geometric constraints. However, in the published study in **publication [B]**, a safe trajectory is guaranteed by an allision constraints generator that implements a point-in-polygon method that is applicable to any shape of the harbor; convex or nonconvex. This method opens up the possibility to extend the SO-TP to take into account unmapped objects (at that instant, not included in the predefined harbor's geometry). However, this advantage comes at the cost of nonlinear constraints that may increase the computational

time. Fortunately, the form of the constraint is kind of analytically differentiable. Thus, it is possible to supply a gradient/slope (search direction) to the solver to speed up the computational time.

On the second problem: slowly maneuvering the ship to the final docking point, this thesis implements two different approaches. The first approach is by implementing a model-based controller: NMPC where the NLP inside it is solved by a nonlinear gradient-free global optimization solver known as CMA-ES. This first approach is tested for a conventional underactuated ship that is equipped with a single rudder and a single propeller. The next approach is by designing a novel positioning system that utilizes VecTwin rudder and one bow thruster. This system is called VTPS, which is an acronym for VecTwin positioning system.

Usually, the NLP in an NMPC (controller) is solved by a gradient-based solver since it is fast enough to be implemented in real time. However, the nonlinearity of the dynamics of a conventional underactuated ship at a low speed can result in the optimization having a nonsmooth and rugged (with many local optima) search space. Such a search space can cause a gradient-based solver to get stuck at a local optimum or even fail to converge. This is not a big issue when using CMA-ES, which is powerful in solving a complicated and ill-conditioned problem in exchange for computational time. So, it is interesting to investigate the implementation of CMA-ES to solve optimization problems in the controller in real time. This is simulated and discussed in **publication** [**A**] where CMA-ES has proved itself capable of solving the corresponding NLPs in real time such that the ship was controlled to the final docking point under a constant disturbance. In the controller, the OCP is modified based on some conditions to attenuate unnecessary mode switch (forward/reverse) of the propeller revolution. This is to partially consider the mechanical limitation of the actuators to some extent. To the limit of the author's knowledge, this study is the first to demonstrate the real-time implementation of evolution strategy in solving OCPs related to maneuvering.

The second approach is preferred when the mathematical model or the dynamics of the ship is unavailable or yet to be described. The main contribution is a novel low-speed positioning system termed VecTwin positioning system and abbreviated as VTPS. The system features VecTwin rudder (twin rudders) and a bow thruster; which does not make the ship fully actuated. At a constant propeller revolution in forward mode, a ship equipped with a VecTwin rudder can execute special low-speed maneuvers like hover, sway/crabbing, rotation, and reverse. The ability of the VecTwin rudder to execute hovering underlines the hypothesis **H.V**, i.e., makes it suitable to design a positioning system for docking operations or any other operation that has to be executed at a low speed.

This second approach starts with the assumption (see assumption **A.5-1**) that there is a linear relationship between the rudder angle and the resulting forces due to the interactions between the propeller, the rudder, and the hull. Preferably, this relationship should be obtained empirically (regression/curve fitting) from the results of bollard pull experiments. However, when the resources and opportunity to conduct such experiments are limited, one can simulate the bollard pull tests in CFD simulations; provided that the geometrical model of the ship is available. From these results, a control forces allocation method is designed and finally implemented as a whole in the VTPS. The control forces allocation distributes the forces from the available actuators to satisfy the required forces

so that the control objective can be achieved. The required forces are regulated by a decoupled PID controller that offers robustness and guarantees stability with correct gains.

The VTPS was tested in multiple experiments that combine automatic docking operation (as a positioning operation) and position/station keeping operation. It saw action in controlling the scale model ship to follow the desired poses under random wind/gusts disturbance. It also performed very well in stabilizing the ship's pose against gusts in the position/station keeping experiments. With several improvements, this system shows promising potential for more practical implementation. It can also be an addition to the usual DPS. The study on this approach was presented in **publication** [**E**] and made public in **publication** [**C**]).

6.2 Future Work

Let us start with the SO-TP. The semionline nature of the computational time makes it possible for the SO-TP to be reformulated as an NMPC. In this approach, the SO-TP is used both as a trajectory planner and as a controller where the first control signal in the whole time histories of the optimal control is sent to the ship. This can be tested in both simulation and experiments. When sensors like LiDAR (light detection and ranging) or proximity sensors are integrated, dynamic obstacles and unmapped objects (not the predefined geometry of the harbor) can also be taken into account. The resulting product can be very promising and worth to be investigated in the future. Moreover, like many model-based approaches, it is desired that the mathematical model or the dynamics of the ship is made as simple as possible to achieve real-time performance. This creates the urgency to do a system identification of a less intricate nonlinear model or linear parameter-varying (LPV) model. Having such a less complicated mathematical model enables us to use a broader range of mathematical tools.

For the real-time implementation of CMA-ES, it showed promising potential in the simulation. However, the heavy computational cost has prevented us to implement it in experiments. Matters like signal processing and delays in the communication between the controller and the ship may further degrade the performance of the controller which was promising in the simulation. Like the previous passage, the computational cost can be reduced when the mathematical model is less complicated. Of course, using a more powerful machine can also be another option. Nevertheless, we have demonstrated that it is indeed possible to implement CMA-ES in real time. Thus, there may be other tasks related to control that the CMA-ES can solve in real time.

It has been demonstrated in the experiments that the linearity assumption (assumption **A.5-1**) is indeed applicable for positioning operations and position/station keeping operations. This statement is also applicable to all of the established assumptions (assumptions **A.5-2** to **A.5-4**). However, these assumptions significantly prevent the VecTwin rudder to be at its peak performance. Take an example when the VTPS controlled the ship in a 180 deg/U-turn docking operation. The current VTPS relies heavily on the bow thruster to turn and rotate the ship. On the other hand, the rudders are utilized mainly to transfer the constant thrust such that the ship is maintained at a low speed (recall that the propeller revolution is constant in forward mode). In practice, the VecTwin rudder increases the maneuverability of the ship significantly. Thus, the biggest

task is to find a way such that one can utilize the VecTwin rudder to its full capacity while still guaranteeing the stability of the controller. One of the ways is to create multiple commands-forces relationship for different ranges of rudder angles such that they cover the whole possible range of the rudder angle. Therefore, whenever the required forces are too much for a certain range, the VTPS should select the appropriate commands-forces relationship. This is somehow identical to gain scheduling or linear parameter-varying (LPV) model, which is common when dealing with a nonlinear problem.

Moreover, mechanical limitations like rudder rate and how fast the thruster changes its mode can also be taken into account in the VTPS. This is very important to prevent the wear of the actuators, especially when implementing a derivative term in the PID controller. The derivative term is important for the position/station keeping operation but it is very sensitive to high-frequency noise from the sensors that can cause unnecessary switches of the actuators. This can cause wear of the actuators. Therefore, in addition to restricting the rate of the actuators, the filtering technique of the state measurements should be improved.

In addition, it is also interesting to employ both the SO-TP and the VTPS together in an integrated system. The SO-TP serves as the middle-level trajectory planner that generates the reference trajectory or path. Meanwhile, the VTPS will be the low-level controller that generates the necessary actuator command/control commands to follow the path generated by the SO-TP. In this approach, with LiDAR, the system can modify the path based on the actual condition of the harbor (dynamic obstacles).

Bibliography

[1] Maritime Safety Committee, "Outcome of the regulatory Scoping Exercise for the use of Maritime Autonomous Surface Ships (MASS)," International Maritime Organization, London, Tech. Rep., 2021, MSC.1/Circ.1638. [Online]. Available: https://www.imo.org/en/MediaCentre/HotTopics/Pages/Autonomous-shipping.aspx

[2] Lloyd's Register: Cyber-enabled ships, "Cyber-enabled ships: ShipRight procedure assignment for cyber descriptive notes for autonomous & remote access ships," Lloyd's Register, Tech. Rep., 2017, a Lloyd's Register guidance document. [Online]. Available: https://www.lr.org/en/cyber-safe-for-marine/

[3] Ø. J. Rødseth and H. Nordahl, "Definitions for autonomous merchant ships," Norwegian Forum for Unmanned Ships, Tech. Rep., 2017.

[4] N. Im and K. Hasegawa, "A study on automatic ship berthing using parallel neural controller," *Journal of the Kansai Society of Naval Architects, Japan*, vol. 2001, no. 236, pp. 65–70, 2001.

[5] L. Yang and G. Chen, "Automatic berthing control of underactuated surface ships in restricted waters based on nonlinear adaptive control method," in *Proceedings of the 31st Chinese Control Conference*, 2012, pp. 939–944.

[6] Y. A. Ahmed and K. Hasegawa, "Implementation of automatic ship berthing using artificial neural network for free running experiment," *IFAC Proceedings Volumes*, vol. 46, no. 33, pp. 25–30, 2013.

[7] Y. A. Ahmed and K. Hasegawa, "Experiment results for automatic ship berthing using artificial neural network based controller," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 2658–2663, 2014.

[8] N. Mizuno, Y. Uchida, and T. Okazaki, "Quasi real-time optimal control scheme for automatic berthing," *IFAC-PapersOnLine*, vol. 48, no. 16, pp. 305–312, 2015, 10th IFAC Conference on Manoeuvring and Control of Marine Craft MCMC 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896315021886

[9] Y. Liao, Z. Jia, W. Zhang, Q. Jia, and Y. Li, "Layered berthing method and experiment of unmanned surface vehicle based on multiple constraints analysis," *Applied Ocean Research*, vol. 86, pp. 47–60, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141118718304759

[10] A. Maki, N. Sakamoto, Y. Akimoto, H. Nishikawa, and N. Umeda, "Application of optimal control theory based on the evolution strategy (CMA-ES) to automatic berthing," *Journal of Marine Science and Technology*, vol. 25, no. 1, pp. 221–233, 2020.

[11] S. Li, J. Liu, R. R. Negenborn, and Q. Wu, "Automatic docking for underactuated ships based on multi-objective nonlinear model predictive control," *IEEE Access*, vol. 8, pp. 70 044–70 057, 2020.

[12] S. Han, Y. Wang, L. Wang, and H. He, "Automatic berthing for an underactuated unmanned surface vehicle: A real-time motion planning approach," *Ocean Engineering*, vol. 235, p. 109352, 2021. [Online]. Available: https:

//www.sciencedirect.com/science/article/pii/S002980182100768X
[13] R. Sawada, K. Hirata, Y. Kitagawa, E. Saito, M. Ueno, K. Tanizawa, and J. Fukuto, "Path following algorithm application to automatic berthing control," *Journal of Marine Science and Technology*, vol. 26, pp. 541–554, 2021. [Online]. Available: https://doi.org/10.1007/s00773-020-00758-x
[14] Y. Miyauchi, R. Sawada, Y. Akimoto, N. Umeda, and A. Maki, "Optimization on planning of trajectory and control of autonomous berthing and unberthing for the realistic port geometry," *Ocean Engineering*, vol. 245, p. 110390, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801821016826
[15] K. Shouji, K. Ohtsu, and S. Mizoguchi, "An automatic berthing study by optimal control techniques," *IFAC Proceedings Volumes*, vol. 25, no. 3, pp. 185–194, 1992.
[16] N. Mizuno, H. Kakami, and T. Okazaki, "Parallel simulation based predictive control scheme with application to approaching control for automatic berthing," *IFAC Proceedings Volumes*, vol. 45, no. 27, pp. 19–24, 2012, 9th IFAC Conference on Manoeuvring and Control of Marine Craft. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667016311983
[17] N. K. Im and V. S. Nguyen, "Artificial neural network controller for automatic ship berthing using head-up coordinate system," *International Journal of Naval Architecture and Ocean Engineering*, vol. 10, no. 3, pp. 235–249, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2092678216304368
[18] A. B. Martinsen, A. M. Lekkas, and S. Gros, "Autonomous docking using direct optimal control," *IFAC-PapersOnLine*, vol. 52, no. 21, pp. 97–102, 2019, 12th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles CAMS 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896319321755
[19] G. Bitar, A. B. Martinsen, A. M. Lekkas, and M. Breivik, "Trajectory planning and control for automatic docking of ASVs with full-scale experiments," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 14 488–14 494, 2020, 21st IFAC World Congress. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896320318632
[20] Y. Zhang, M. Zhang, and Q. Zhang, "Auto-berthing control of marine surface vehicle based on concise backstepping," *IEEE Access*, vol. 8, pp. 197 059–197 067, 2020.
[21] A. B. Martinsen, G. Bitar, A. M. Lekkas, and S. Gros, "Optimization-based automatic docking and berthing of ASVs using exteroceptive sensors: Theory and experiments," *IEEE Access*, vol. 8, pp. 204 974–204 986, 2020.
[22] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, *Path Planning and Trajectory Planning Algorithms: A General Overview*. Cham: Springer International Publishing, 2015, pp. 3–27. [Online]. Available: https://doi.org/10.1007/978-3-319-14705-5_1
[23] C. Zhou, S. Gu, Y. Wen, Z. Du, C. Xiao, L. Huang, and M. Zhu, "The review unmanned surface vehicle path planning: based on multi-modality constraint," *Ocean Engineering*, vol. 200, p. 107043, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801820301177
[24] A. Vagale, R. Oucheikh, R. T. Bye, O. L. Osen, and T. I. Fossen, "Path planning and collision avoidance for autonomous surface vehicles I: a review," *Journal of Marine Science and Technology*, pp. 1–15, 1 2021. [Online]. Available:

https://link.springer.com/article/10.1007/s00773-020-00787-6

[25] A. Ravankar, A. A. Ravankar, Y. Kobayashi, Y. Hoshino, and C.-C. Peng, "Path smoothing techniques in robot navigation: State-of-the-art, current and future challenges," *Sensors (Basel)*, vol. 18, no. 9, Sep. 2018.

[26] R. Song, Y. Liu, and R. Bucknall, "Smoothed A* algorithm for practical unmanned surface vehicle path planning," *Applied Ocean Research*, vol. 83, pp. 9–20, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141118718302621

[27] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, 1998. [Online]. Available: https://doi.org/10.2514/2.4231

[28] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2nd ed. Society for Industrial and Applied Mathematics, 2010. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9780898718577

[29] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Review*, vol. 59, no. 4, pp. 849–904, 2017. [Online]. Available: https://doi.org/10.1137/16M1062569

[30] D. E. Kirk, *Optimal control theory: an introduction*. Courier Corporation, 2004.

[31] L. Grüne, J. Pannek, L. Grüne, and J. Pannek, *Nonlinear model predictive control*. Springer, 2017.

[32] Y. Yoshimura, I. Nakao, and A. Ishibashi, "Unified mathematical model for ocean and harbour manoeuvring," in *International Conference on Marine Simulation and Ship Maneuverability*, 2009, pp. 116–124. [Online]. Available: http://hdl.handle.net/2115/42969

[33] T. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. Wiley, 2011. [Online]. Available: https://books.google.co.jp/books?id=vCAzd3DaZCgC

[34] L. Wang, Q. Wu, J. Liu, S. Li, and R. R. Negenborn, "State-of-the-art research on motion control of maritime autonomous surface ships," *Journal of Marine Science and Engineering*, vol. 7, no. 12, 2019. [Online]. Available: https://www.mdpi.com/2077-1312/7/12/438

[35] R. S. Anna Witkowska, "Designing a ship course controller by applying the adaptive backstepping method," *International Journal of Applied Mathematics and Computer Science*, vol. 22, no. 4, pp. 985–997, 2012. [Online]. Available: http://eudml.org/doc/244525

[36] Q. Zhang, X. ku Zhang, and N. kyun Im, "Ship nonlinear-feedback course keeping algorithm based on MMG model driven by bipolar sigmoid function for berthing," *International Journal of Naval Architecture and Ocean Engineering*, vol. 9, no. 5, pp. 525–536, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2092678216305945

[37] Z. Liu, X. Lu, and G. Diju, "Ship heading control with speed keeping via a nonlinear disturbance observer," *Journal of Navigation*, vol. 72, pp. 1–18, 01 2019.

[38] A. J. Sørensen, S. I. Sagatun, and T. I. Fossen, "Design of a dynamic positioning system using model-based control," *Control Engineering Practice*, vol. 4, no. 3, pp. 359–368, 1996.

[39] A. J. Sørensen, "A survey of dynamic positioning control systems," *Annual reviews in control*, vol. 35, no. 1, pp. 123–136, 2011.

[40] T. I. Fossen, M. Breivik, and R. Skjetne, "Line-of-sight path following of underactuated marine craft," *IFAC Proceedings Volumes*, vol. 36, no. 21, pp. 211–216, 2003, 6th IFAC Conference on Manoeuvring and Control of Marine Craft (MCMC 2003), Girona, Spain, 17-19 September, 1997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667017378096

[41] E. Borhaug, A. Pavlov, and K. Y. Pettersen, "Integral LOS control for path following of underactuated marine surface vessels in the presence of constant ocean currents," in *2008 47th IEEE Conference on Decision and Control*, 2008, pp. 4984–4991.

[42] C. Paliotta, E. Lefeber, K. Y. Pettersen, J. Pinto, M. Costa, and J. T. de Figueiredo Borges de Sousa, "Trajectory tracking and path following for underactuated marine vehicles," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 4, pp. 1423–1437, 2019.

[43] J. Cheng, J. Yi, and D. Zhao, "Design of a sliding mode controller for trajectory tracking problem of marine vessels," *IET Control Theory & Applications*, vol. 1, pp. 233–237(4), January 2007. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/iet-cta_20050357

[44] K. D. von Ellenrieder, "Stable backstepping control of marine vehicles with actuator rate limits and saturation," *IFAC-PapersOnLine*, vol. 51, no. 29, pp. 262–267, 2018, 11th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles CAMS 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S240589631832202X

[45] J.-Y. Park and N. Kim, "Design of an adaptive backstepping controller for auto-berthing a cruise ship under wind loads," *International Journal of Naval Architecture and Ocean Engineering*, vol. 6, no. 2, pp. 347–360, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2092678216303065

[46] N. Hansen, "The CMA evolution strategy: A tutorial," *arXiv preprint arXiv:1604.00772*, 2016.

[47] Y. Shi, C. Shen, H. Fang, and H. Li, "Advanced control in marine mechatronic systems: A survey," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 3, pp. 1121–1131, 2017.

[48] Japan Hamworthy & Co., Ltd., "Steering systems: New VecTwin system," https://www.japanham.com/en/service/new_vectwin.html, 2022, accessed on 9 May 2023.

[49] H. Yasukawa and Y. Yoshimura, "Introduction of MMG standard method for ship maneuvering predictions," *Journal of Marine Science and Technology*, vol. 20, pp. 37–52, 2014.

[50] Y. Miyauchi, A. Maki, N. Umeda, D. M. Rachman, and Y. Akimoto, "System parameter exploration of ship maneuvering model for automatic docking/berthing using cma-es," *Journal of Marine Science and Technology*, vol. 27, no. 2, pp. 1065–1083, 2022.

[51] T. Hachii, "The prediction of maneuvering motion on ships at low speed using standard MMG model," M.Eng., Osaka University, Osaka, Japan, February 2004.

[52] H. Fujii and T. Tsuda, "Experimental research on rudder performance (2)," *Journal of the Society of Naval Architects of Japan*, vol. 1961, no. 110, pp. 31–42, 1961.

[53] Y. Kitagawa, Y. Tsukada, and H. Miyazaki, "A study on mathematical models of propeller and rudder under maneuvering with propeller reverse rotation," *Conference proceedings, the Japan Society of Naval Architects and Ocean Engineers*, vol. 20, pp.

117–120, 2015.

[54] H. Kobayashi, A. Ishibasi, K. Shimokawa, and Y. Shimura, "A study on mathematical model for the maneuvering motions of twin-propeller twin-rudder ship : In reference to the maneuvering motion from ordinary speed range to low speed range," *The Journal of Japan Institute of Navigation*, vol. 91, pp. 263–270, 1994.

[55] T. Fujiwara, M. Ueno, and T. Nimura, "Estimation of wind forces and moments acting on ships," *Journal of the Society of Naval Architects of Japan*, vol. 1998, pp. 77–90, 1998.

[56] D. M. Rachman, "Study of automatic berthing in real time by CMA-ES," M.Eng., Osaka University, Osaka, Japan, September 2020.

[57] A. Forsgren, "On warm starts for interior methods," in *System Modeling and Optimization*, F. Ceragioli, A. Dontchev, H. Futura, K. Marti, and L. Pandolfi, Eds. Boston, MA: Springer US, 2006, pp. 51–66.

[58] K. Hormann and A. Agathos, "The point in polygon problem for arbitrary polygons," *Computational Geometry*, vol. 20, no. 3, pp. 131–144, 2001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925772101000128

[59] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta Numerica*, vol. 4, p. 1–51, 1995.

[60] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evolutionary computation*, vol. 11, no. 1, pp. 1–18, 2003.

[61] A. Auger and N. Hansen, "A restart CMA evolution strategy with increasing population size," in *2005 IEEE congress on evolutionary computation*, vol. 2. IEEE, 2005, pp. 1769–1776.

[62] N. Sakamoto and Y. Akimoto, "Modified box constraint handling for the covariance matrix adaptation evolution strategy," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2017, pp. 183–184.

[63] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, 2nd ed. Cambridge, USA: Cambridge University Press, 1992.

[64] A. Roubos, L. Groenewegen, and D. J. Peters, "Berthing velocity of large seagoing vessels in the port of Rotterdam," *Marine Structures*, vol. 51, pp. 202–219, 2017.

[65] A. Alessandretti, A. P. Aguiar, and C. N. Jones, "Trajectory-tracking and path-following controllers for constrained underactuated vehicles using model predictive control," in *2013 European Control Conference (ECC)*, 2013, pp. 1371–1376.

[66] M. Abdelaal, M. Fränzle, and A. Hahn, "NMPC-based trajectory tracking and collison avoidance of underactuated vessels with elliptical ship domain," *IFAC-PapersOnLine*, vol. 49, no. 23, pp. 22–27, 2016, 10th IFAC Conference on Control Applications in Marine SystemsCAMS 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896316319036

[67] M. Diehl, H. J. Ferreau, and N. Haverbeke, *Efficient Numerical Methods for Nonlinear MPC and Moving Horizon Estimation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 391–417. [Online]. Available: https://doi.org/10.1007/978-3-642-01094-1_32

[68] J. G. Balchen, N. A. Jenssen, E. Mathisen, and S. Sælid, "A dynamic positioning system based on Kalman filtering and optimal control," *Modeling, Identification and Control*, vol. 1, no. 3, pp. 135–163, 1980.

[69] E. Tannuri, A. Agostinho, H. Morishita, and L. Moratelli, "Dynamic positioning systems: An experimental analysis of sliding mode control," *Control Engineering Practice*, vol. 18, no. 10, pp. 1121–1132, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0967066110001486

[70] H. Yasukawa, N. Hirata, S. Tanaka, and H. Hata, "Tank tests on low speed maneuvering of a ship with VecTwin rudder," *The Journal of Japan Institute of Navigation*, vol. 124, pp. 265–271, 03 2011, in Japanese.

[71] E. Tannuri and H. Morishita, "Experimental and numerical evaluation of a typical dynamic positioning system," *Applied Ocean Research*, vol. 28, no. 2, pp. 133–146, 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141118706000587

[72] H. L. Alfheim, K. Muggerud, M. Breivik, E. F. Brekke, E. Eide, and Øystein Engelhardtsen, "Development of a dynamic positioning system for the ReVolt model ship," *IFAC-PapersOnLine*, vol. 51, no. 29, pp. 116–121, 2018, 11th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles CAMS 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896318321682

[73] E. A. Tannuri, T. T. Bravin, and C. P. Pesce, "Dynamic positioning systems: Comparison between wave filtering algorithms and their influence on performance," *Proceedings of the ASME 2003 22nd International Conference on Offshore Mechanics and Arctic Engineering. Volume 1: Offshore Technology; Ocean Space Utilization*, pp. 109–117, 06 2003.

[74] V. Hassani, A. J. Sørensen, A. M. Pascoal, and A. P. Aguiar, "Multiple model adaptive wave filtering for dynamic positioning of marine vessels," *Proceedings of 2012 American Control Conference (ACC)*, pp. 6222–6228, 2012.

[75] K. Hasegawa, D. Kang, M. Sano, and K. Nabeshima, "Study on the maneuverability of a large vessel installed with a mariner type super VecTwin rudder," *Journal of Marine Science and Technology*, vol. 11, pp. 88–99, 06 2006.

[76] Y. Aoki, "Study of the interaction between propeller and rudder of VecTwin rudder ship using CFD free-running stopping simulations," Master Thesis, Osaka University, Osaka, Japan, March 2022.

[77] R. G. Rajagopalan and S. R. Mathur, "Three dimensional analysis of a rotor in forward flight," *Journal of the American Helicopter Society*, vol. 38, no. 3, pp. 14–25, 1993.

[78] F. R. Menter, M. Kuntz, and R. Langtry, "Ten years of industrial experience with the SST turbulence model," *Heat and Mass Transfer*, vol. 4, 01 2003.

[79] M. Hamamoto and T. Enomoto, "Maneuvering performance of a ship with VecTwin rudder system," *Journal of the Society of Naval Architects of Japan*, vol. 1997, no. 181, pp. 197–204, 1997.