NTNU
Norwegian University of Science and Technology

Dynamic Positioning for Unmanned Surface Vehicles

Håvard Halvorsen

Master of Science in Engineering Cybernetics

Submission date:	June 2008
Supervisor:	Morten Breivik, ITK
Co-supervisor:	Vegard Evjen Hovstein, Maritime Robotics AS

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem Description

Description: The candidate will consider the problem of dynamic positioning (DP) for unmanned surface vehicles
(USVs). The following elements must be considered:

1. Set up and present a progress plan for your thesis work so that the due date can be met.
2. Review state-of-the-art concepts for dynamic positioning of marine surface vessels, and assess their applicability for
implementation with regard to the Maritime Robotics Viknes 830 vessel (actuation, sensor, and computation issues).
3. Suggest control strategies that realize DP capability for the current Viknes 830 configuration, and perform a
simulation study in Matlab/Simulink to support your claims.
4. Implement the suggested control strategies onboard the Viknes USV and perform full-scale experiments.
5. Suggest upgrades to the existing USV configuration that facilitate additional DP functionality, and evaluate the
suggestions with regard to performance-price criteria.

Assignment given: 09. January 2008
Supervisor: Morten Breivik, ITK

Abstract

This thesis develops a Dynamic Positioning (DP) system for small marine craft by using the LQR controller approach. Development has been done with a 'Viknes 830' vessel in mind, which is operated by the company 'Maritime Robotics AS' and will be equipped for DP operation the during summer of 2008. A Matlab-based simulator designed for DP simulations has been developed, and is used throughout the thesis. Furthermore, a Hardware-In-the-Loop (HIL) simulator has been used in order to localize and resolve as many implementation issues as possible prior to full-scale installation. A discussion on the general use of a HIL simulator for DP is included.

Three variations of a feedback LQR station-keeping controller have been implemented and compared; a simple LQR controller, an LQR controller with modeled actuator dynamics, and finally an LQR controller with actuator dynamics and integral action.

A feedforward controller has been added in order to provide enhanced station-keeping performance, as well as bumpless transfer from station keeping to low-speed maneuvering. A reference model has been created for smooth transfer in-between station-keeping reference points, and as input for the feedforward controller. A passive Luenberger DP observer has been applied in order to filter out high-frequency wave loads.

Simulation results reveal that the LQR controller with actuator dynamics and integral action is most likely to perform well in real-life application. The largest performance enhancement is gained from the inclusion of actuator dynamics in the controller. It is discovered that the performance turns out better if the actuator dynamics is modeled faster in the controller due to unmodeled actuator saturation limits.

Preface

This thesis represents the finalization of my time as a student at NTNU. Little did I know of that being a student in Trondheim could be so much fun, thanks to all the great people that I've been so lucky to become friends and colleagues with during these last five years.

Dynamic positioning is a subject that I have enjoyed working with. Not only because I find the subject highly interesting, but it has also allowed me to apply a wide range of what I have learned in the past years. Finally, choosing this thesis was a simple decision when I was allowed to work on a project that can be applied in a real-life applications within just a short time horizon.

I've had the privilege of performing this work for *Maritime Robotics AS*, which is a newly founded technology company developing unmanned surface vehicles (USVs) for the offshore industry. This thesis is performed in the hope that it can be of use in their ongoing USV project.

I would like to thank the following people for their help and appreciated effort throughout the thesis period:

Morten Breivik	My thesis advisor
Vegard Evjen Hovstein	Managing director at Maritime Robotics AS
Aasmund Fannemel	Fellow student
Øivind A. Loe	Fellow student

In particular, I want to express my gratitude to Morten Breivik for the much appreciated feedback he has given me throughout the last year.

Håvard Halvorsen
June 13, 2008

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Up until now, DP has been mainly used for large offshore vessels, and lately also cruise ships, minehunters, etc. DP for small craft has traditionally seen smaller usage. Nowadays, as software and hardware expenses related to DP for smaller vessels are decreasing, DP technology is becoming more available to the recreational boater. Unmanned surface vehicles (USV) technology is starting to gain popularity [24]. Station keeping and low-speed maneuvering are qualities that are important to many USV applications. Even though recreational DP and DP for USVs is based on the same technology, the demand for robustness in the USV market is larger than for the leisure boat market.

The increasing interest in DP for small craft is the main motivator for this study on the LQR DP approach. Furthermore, the thesis includes the development of an LQR controller for the Viknes 830 vessel (see Section 7), using a HIL simulator. This implementation is motivated by the enlightenment of practical issues that is normally not discovered through normal simulation.

1.2 Contribution

Aside of being a literature study on DP for small marine craft, the contribution from this thesis can be summarized as:

- A comparison and discussion on three different implementations of the LQR DP algorithm.

- A discussion on the use of a HIL simulator in a practical DP implementation.

- A new proposed method on how to manipulate the integral action in order to obtain better stability in the case of rapidly varying yaw setpoints.
- A Matlab simulator with many features applicable to DP simulations.

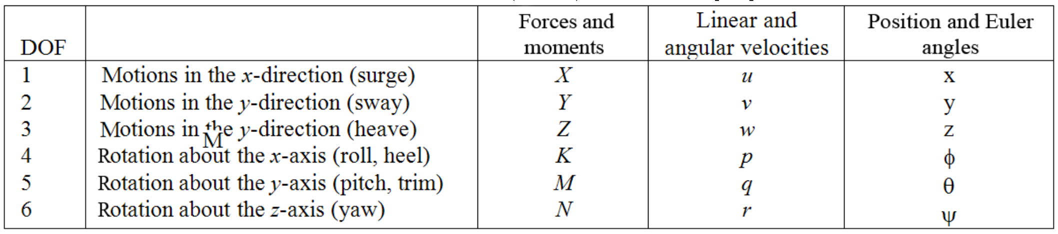
1.3 Outline

- **Chapter 2** introduces the concept of DP. Some DP principles are discussed, and a brief history of DP is also provided. Existing commercially available DP solutions for small marine craft are introduced.
- **Chapters 3-4** covers the mathematical background for the modeling of vessels and environmental disturbances. Chapter 4 applies the theory presented in Chapter 3 to present the mathematics applied in the DP controllers presented in this thesis.
- **Chapter 5** presents simulation results gained from applying the LQR approaches described in Chapter 4. The Matlab simulator developed in order to perform DP simulations and model identification is also presented.
- **Chapter 6** introduces the concept of Hardware-In-the-Loop(HIL) simulation, and its usage in DP application. Furthermore, the HIL simulator available at Maritime Robotics is presented, and results from HIL simulations are presented.
- **Chapter 7** introduces the 'Viknes 830' vessel available at Maritime Robotics. Hardware and hardware modifications are discussed, and a section on model identification is provided.
- **Chapters 8** Conclusion

1.4 Notation

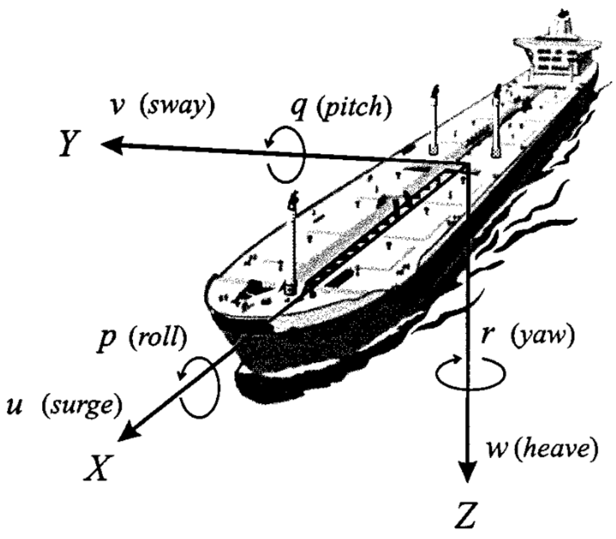
When applicable, the SNAME notation for marine vessels is applied. As stated in [14]:

Table 1.1: The SNAME (1950) notation [14].



DOF		Forces and moments	Linear and angular velocities	Position and Euler angles
1	Motions in the x-direction (surge)	X	u	x
2	Motions in the y-direction (sway)	Y	v	y
3	Motions in the y-direction (heave)	Z	w	z
4	Rotation about the x-axis (roll, heel)	K	p	ϕ
5	Rotation about the y-axis (pitch, trim)	M	q	θ
6	Rotation about the z-axis (yaw)	N	r	ψ

Figure 1.1: Motion variables for a marine vessel[14].



$\boldsymbol{M}$	- system inertia matrix (including added mass)
$\boldsymbol{C}(\boldsymbol{\nu})$	- coriolis-centripetal matrix (including added mass)
$\boldsymbol{D}(\boldsymbol{\nu}_r)$	- damping matrix
$\boldsymbol{\eta}$	- $\begin{bmatrix} x & y & z & \phi & \theta & \psi \end{bmatrix}^\top$
$\boldsymbol{\nu}$	- $\begin{bmatrix} u & v & w & p & q & r \end{bmatrix}^\top$
$\boldsymbol{x}$	- $\begin{bmatrix} \boldsymbol{\eta} & \boldsymbol{\nu} \end{bmatrix}^\top$
$\boldsymbol{\nu}_c$	- current velocity vector
$\boldsymbol{\nu}_r$	- vessel velocity relative to current velocity; $\boldsymbol{\nu} - \boldsymbol{\nu}_c$
$\boldsymbol{g}(\boldsymbol{\eta})$	- vector of gravitational/buoyancy forces and moments
$\boldsymbol{\tau}$	- vector of control inputs
$\boldsymbol{\tau}_{com}$	- vector of commanded forces from the DP controller
$\boldsymbol{g}_0$	- local gravity force
$\boldsymbol{w}$	- vector of environmental disturbances (wind and waves)

1.5 Abbreviations

CD	- Center of Dissipative forces
CMD	- Cummins Mercruiser Diesel
CG	- Center of Gravity
DOF	- Degree Of Freedom
DP	- Dynamic Positioning
ECEF	- Earth Centered Earth Fixed
GNC	- Guidance Navigation and Control
GPS	- Global Positioning System
HIL	- Hardware In the Loop
INS	- Inertial Navigation System
NED	- North East Down
NFA	- Norsk Forening for Automatisering
MPC	- Model Predictive Control
MPM	- Modified Pierson-Moskowitz
OBC	- On Board Computer
ODE	- Ordinary Differential Equation
RAO	- Response Amplitude Operator
RPM	- Revolutions Per Minute
SNAME	- Society of Naval Architects and Marine Engineers
USV	- Unmanned Surface Vehicle
WF	- Wave Frequency
WOPC	- Weather Optimal Positioning Control
NMEA	- National Marine Electronics Association

Chapter 2

DP fundamentals

Fay [13] defines dynamic positioning of a floating vessels as follows:

"A process involving the action of thrusters which, commanded by a controller and opposing the environmental forces, maintain a ship or any other floating vessel in the vicinity of a reference point and stabilize its heading. The position is known at all times from the data transmitted by a position reference system."

This section covers DP history as well as an overview of the most common DP technologies. Existing DP technology for small marine craft are also discussed. Dynamic positioning also include stating keeping for moored vessels. This is often done in order to save fuel on larger craft that are to remain stationary for a longer period of time. However, position mooring is not discussed in this report.

2.1 DP history

The very first offshore operation by using a moored ship, the *Submarex*, was done at a depth of 120 meters outside the shore of California in 1953. The positioning problem was solved using several anchors. As requirements for operations at greater water depths rose, so did problems with anchor-based positioning. Factors like vessel mass, elasticity of the anchoring system and weak hydrodynamic damping could give rise to strong oscillating movements, depending on the environmental conditions (see Section 1.2 in [13]). These problems grew as water depths became larger. In 1957, the American project *Mohole* involved drilling at a water depth of 4500 meters. It became clear that anchors had to be replaced by an alternative system that could solve the problem of position holding regardless of water depth [13].

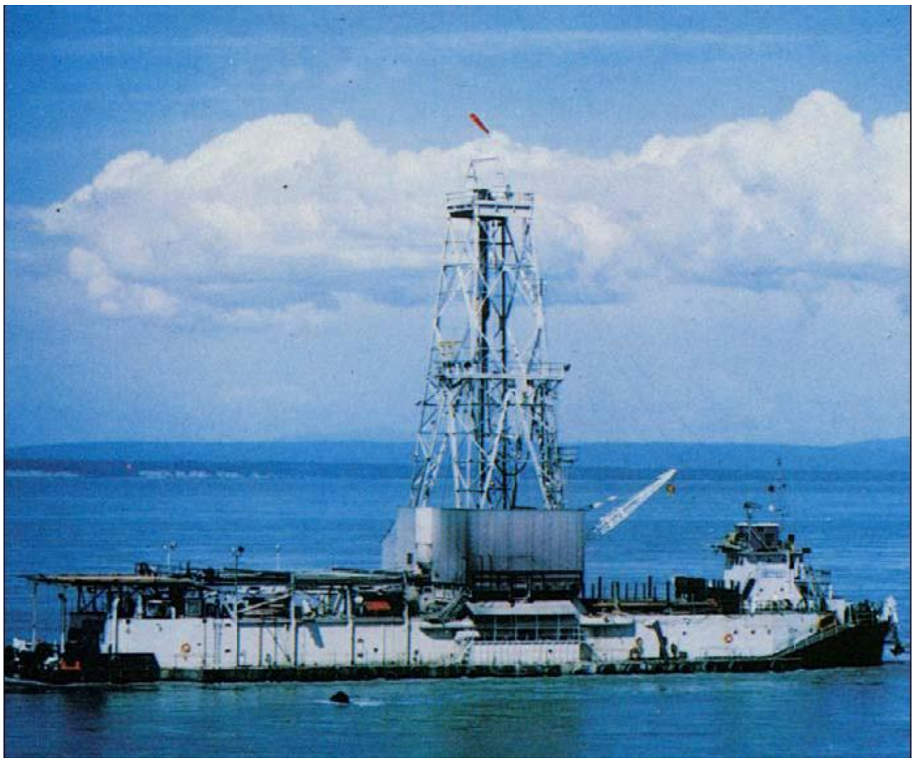


Figure 2.1: *Cuss 1*, The first dynamically positioned vessel,equipped with four steerable thrusters. Courtesy of Friede & Goldman, LTD [5].

In 1958, the vessel *Cuss 1* was used in the Mohole project. The positioning problem was solved by using four manually-controlled thrusters (steerable, with2 00 hp each), one in each corner of the vessel. As one can imagine, the simultaneous manual control of the thrusters was both tedious and difficult. The idea of a central controller was then developed. In 1961, the vessel *Eureka*, launched on behalf of the *Shell Oil Company* became the first ship equipped with a controller for automatic position and heading maintenance. These first implementations were mostly analogue, and there were no redundancy in any of the systems [11].

The first dynamically positioned vessels used radar and taut wire[1] in order to measure positions. Since 1961, there has been much development within the field of DP. The area of usage has grown from the first drillships applications to other commercial applications such as crane vessels, cable layers, accommodation vessels, fire fighting vessels and more [20].

[1]Taut wire: A tensioned wire between the vessel and a seabed weight. The wire is held in constant tension, and is used for position measurement [20].

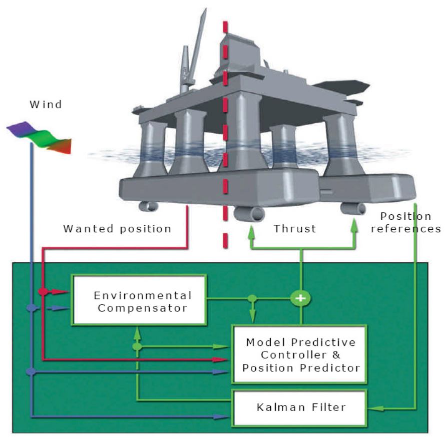

Wind
Wanted position
Thrust
Position references
Environmental Compensator
Model Predictive Controller & Position Predictor
Kalman Filter

Figure 2.2: Modern DP Technology: Schematic view of the *Green DP*® controller. Courtesy of Kongsberg Simrad [16].

A large vessel can have a displacement of thousands of tonnes. Wave movements still lifts these large ships up and down without problems. These are enormous forces that thrusters cannot handle. Consequently, there was needed some sort of motion filtering. The first DP systems used notch filters to remove the high-frequency component from the position measurement. However, these filters inevitably introduced an undesirable feedback lag. Subsequently, new techniques such as a model-based concept utilizing stochastic theory and Kalman filtering was introduced, and solved the problem of feedback lag [20].

By the late 1970s, DP had become an established technique. In 1980, the number of DP capable vessels was about 65, and increased to 150 within 1985. As of 2003, the number passed a thousand DP-capable ships [11].

The latest development in the DP area is the *Green DP*, provided by Kongsberg Maritime AS, which utilizes non-linear model predictive control (MPC) in order to perform DP action with focus on fuel-economy. We are also seeing that DP technology is starting to become available to recreational boaters, through new products such as The *Volvo Penta IPS* system (see Section 2.2.2).

There are also existing DP technologies for USVs available. A throughout discussion is available in [12].

2.2 DP technologies for small marine craft

As discussed in Section 1.1, the number of applications for small craft DP has been relatively low. However, as the request is growing, a few products have been released to the consumer market. This section covers the ones that was found available at the time of writing.

2.2.1 The Mercury Marine Zeus/Axius System

Introduction The soon-to-be consumer available "Zeus" and "Axius" systems are advanced marine propulsion systems delivered by Mercury Marine and Cummins Mercruiser Diesel (CMD). The system is designed for smaller yachts. Mercury Marine claims to deliver both better high- and low-speed maneuvering, and also encompasses a DP system called "Skyhook electronic anchor".

"Zeus" is a complete system that comes with so-called "pod drives", while "Axius" encompasses the same technology as "Zeus", but is designed to be installed in existing hulls. The limitation being that it must be either a MerCruiser gasoline engine or a Cummins Mercruiser Diesel engine in the range 260 - 425 HP.

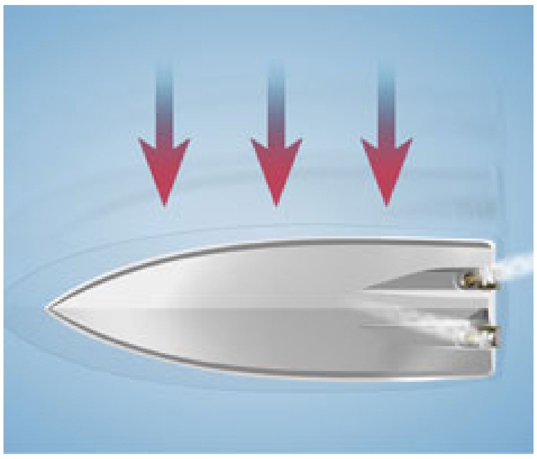


Figure 2.3: The Mercruiser Zeus system in action, moving the vessel in the sway direction. Courtesy of Mercury Marine [2].

Technology "Zeus" and "Axius" utilizes two individually controllable rear-mounted propellers. These are referred to as "pod drives" by CMD Marine, but appears similar to azimuth thrusters. These are used to render the vessel fully actuated. The user interface is an intuitive joystick that

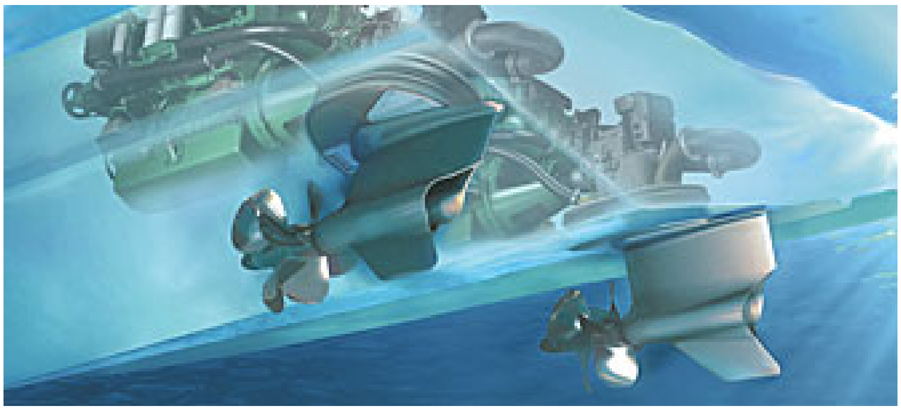


Figure 2.4: The Volvo Penta IPS System. With permission from Volvo Penta AS.

enables the user to maneuver the vessel in all horizontal directions, and spin around the center vertical axis.

DP capabilities The system includes a DP mode presented as ”Skyhook electronic anchor”. By using GPS as a position reference [10], it enables the vessel to maintain a set position and heading. CMD Marine claims the system is able to operate in ”(...) strong currents and windy conditions” [3].

Industrial application While these systems are designed and developed for the consumer market, their appliance in the industrial field is an important consideration. Due to the fact that the system has just been released, information on reliability standards, etc., are not currently available, and will need attention when released. A strength of the system is its use of two main engines, which leaves room for a main engine failure. This of course depends on the system routines for such failure handling.

2.2.2 Volvo Penta IPS

Released in 2005, the Volvo Penta IPS system is similar to ”Mercury Marine Zeus”, the main difference being that main propellers are directed forward instead of backwards [8].

An interview with Frank Finnkroken from Volvo Penta reveals that the Volvo Penta IPS systems is not factory ready to be computer controlled, and would need modifications. Seen from an unmanned operation point of view, reliability is of importance. The Volvo Penta IPS system is designed to handle a single-engine breakdown by providing the same controllers to the operator, but with severely reduced efficiency. Using two engines augments the safety level of an unmanned vessel significantly. Finnkroken states that the precision of the Skyhook function is designed for recreational use, like fishing. Though dependent on engine and vessel configuration, the performance of the DP mode should handle ”pretty rough conditions”.

2.3 Underactuated DP for small marine craft

The LQR control approach for DP requires a fully actuated vessel[2]. However, underactuated DP is included because of its growing actuality, even for vessels that are fully actuated. [18] or [19] provides a thorough discussion on the problem of underactuated maneuvering.

The winnings of mastering underactuated control for small marine craft is twofold. The first and most obvious winning is the ability to perform DP operations with vessels not equipped with extra thrusters. The second reason is that one will sometimes want to operate a fully actuated craft using underactuated DP, due to better fuel economy and reduced equipment wear. As electrical thrusters tends to represent a significant battery drain when operating, one can not use electrical thrusters in extended periods of time, given a standard setup. The generator in a vessel such as the Viknes 830 described in Chapter 7 is able to deliver about 100 Amps (12 voltage system), while an adequately sized bow thruster can drain as much as 700 Amps from the battery, effectively reducing the available operating time to about 20 minutes, depending on battery capacity.

It is clear that this is not a solution that can be used for, e.g., offshore DP applications that may require hours of station keeping. However, the problem is solved if one can apply a method that manages to control the vessel in an underactuated mode, like the WOPC approach (see Section 2.3.1).
The ability to maneuver a craft using only the main propeller will result in greatly reduced DP implementation costs. Thus the ability to solve problems using underactuated control can give winnings, especially in the consumer market, where pricing level is crucial.

2.3.1 Weather Optimal Positioning Control (WOPC)

Weather optimal positioning control can be applied to an underactuated vessel, and a description of the principle is included here because of its applicability both to underactuated and fully actuated vessels. Given a USV

[2]A fully actuated vessel is a vessel that is capable of moving in the surge, sway, and yaw direction without interfering with eachother. E.g., a vessel that cannot more only in the surge direction is *underactuated.*

that is operating offshore, and is required to stay in the same position for a prolonged period of time, fuel economy and minimized equipment wear is crucial. As this controller makes the vessel always face the sum of the environmental forces, it will keep lateral thruster usage to a minimum while also reducing stress from the main engine, as the most energy economic heading for a vessel is facing the environmental load.

The principle of WOPC was introduces Strand and Fossen in [15]. The concept of weather optimal position control is a way of making environmental disturbances produce zero yaw momentum without measuring the environmental disturbances. The basic idea is to picture the resulting environmental forces as a force field similar to gravity, and then let the vessel act as a pendulum in this field. The control goal is to make the vessel stay on the cicle, while always facing the circle centre. Whenever the vessel if facing mean environmenal load from somewhere other than straight ahead, the vessel will be forced to travel along the circle until the mean load is facing the vessel from stright ahead. The idea is illustrated in Figure 2.5.

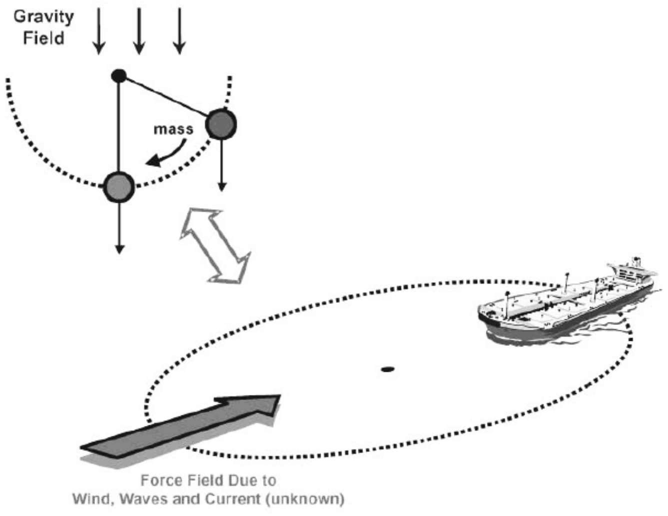


Figure 2.5: The concept of Weather Optimal Positioning Control. Courtesy of [15].

A controller is used to make the vessel stay on the circle. The sum of environmental disturbances will make the vessel automatically come to rest at the equilibrium point. Furthermore, the controller can be augmented with moving the center of rotation, so that the vessel remains in the same position, while the circle is moving. A thorough description can be found in the original publication [15].

Note that the principle of this approach is patented, which can result in restricted usage.

Chapter 3

Mathematical modeling

This chapter gives the mathematical background for the modeling of vessels, thrusters, and environmental disturbances. These models are utilized and further elaborated throughout the thesis, specifically in:

- Chapter 4.1 - DP controllers
- Chapter 5.1 - Matlab simulator
- Chapter 6 - HIL simulator

3.1 Vessel dynamics

As introduced by Fossen in 1991, the following vectorial convention conveniently expresses the 6-DOF nonlinear dynamic equation of motion, as stated in [14]:

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} + \boldsymbol{g}_0 + \boldsymbol{w}. \tag{3.1}$$

This setup allows for advanced modeling of vessels. Simplifications has been done in order to make the system appropriate for DP simulation. As DP operation in surge, sway, and yaw only requires a 3-DOF model, this significantly reduces the complexity of (3.1). The following sections summarizes the meaning of the varying terms in (3.1), and explains how one can simplify these in order to meet an adequate level of precision.

3.1.1 Inertia matrix

The inertia matrix $\boldsymbol{M}$ from (3.1) includes both rigid-body inertia and added-mass inertia, defined as:

$$\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_A. \tag{3.2}$$

The added mass $\boldsymbol{M}_A$ is due to the inertia of the surrounding fluid, and will only contribute whenever the vessel body is accelerating relative to the water masses. As dynamic positioning is limited to low-speed maneuvering and station keeping, acceleration in calm waters can be considered small enough to be neglected. Also, a constant (or slowly varying) current will not impose any acceleration of the vessel relative to the water masses. However, when waves are present, the vessel will frequently accelerate relative to the water masses in both the horizontal and vertical direction. Thus, the effect of added mass has the biggest impact with respect to waves. Also, wave motion tends to accelerate the vessel more in the vertical than the horizontal direction, thus the effect can be neglected in the 3-DOF case.

As stated in [14], the generalized 6-DOF $\boldsymbol{M}_{RB}$ is defined as follows:

$$\boldsymbol{M}_{RB} = \begin{bmatrix} m\boldsymbol{I}_{3\times3} & -m\boldsymbol{S}(\boldsymbol{r}_g^b) \\ m\boldsymbol{S}(\boldsymbol{r}_g^b) & \boldsymbol{I}_0 \end{bmatrix} \tag{3.3}$$

$$= \begin{bmatrix} m & 0 & 0 & 0 & mz_g & -my_g \\ 0 & m & 0 & -mz_g & 0 & mx_g \\ 0 & 0 & m & my_g & -mx_g & 0 \\ 0 & -mz_g & my_g & I_x & -I_{xy} & -I_{xz} \\ mz_g & 0 & -mx_g & -I_{yx} & I_y & -I_{yz} \\ -my_g & mx_g & 0 & -I_{zx} & -I_{zy} & I_z \end{bmatrix}, \tag{3.4}$$

where I is the moment of inertia around the respective axes, and m is body mass.

Simplifications As further stated in [14], (3.4) can be greatly simplified by carefully choosing the body-fixed coordinate system. First, by choosing the origin of the body-fixed coordinate system such that it equals the center of gravity of the vessel, and moreover by choosing the body axes to coincide with the principal axes of inertia. Furthermore, the conversion from 6-DOF to 3-DOF results in the following $\boldsymbol{M}_{RB}$:

$$\boldsymbol{M}_{RB} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & \boldsymbol{I}_z \end{bmatrix}, \tag{3.5}$$

where m is the vessel mass, and $\boldsymbol{I}_z$ is the moment of inertia around the vertical axis.

The simplification presented in (3.5) also reduces computational load during simulation, because of the higher number of zero-terms. This approach has been used in the simulator described in Section 5.1.

3.1.2 Coriolis-centripetal matrix

The coriolis effect describes the change of required force to achieve the same acceleration in a rotating frame of reference as in an inertial frame of reference. Thus, the coriolis term $\boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu}$ in (3.1) is dependent on both mass m and velocity ν. In contrast to the inertia matrix $\boldsymbol{M}$, it is possible to find a large number of representations for $\boldsymbol{C}$ [14]. This leaves one with the option to choose an appropriate representation. From a computational point of view, one would prefer a coriolis matrix with as many zero and or equal terms as possible. The definition of $\boldsymbol{C}(\boldsymbol{\nu})$ presented in Section 3.1 of [14] includes both many zero terms, as well as being symmetric. This representation is also chosen in the coriolis matrix tool provided in GNC Toolbox[1], which is used in the simulator described in Chapter 5.1. This representation is described as follows:

$$\boldsymbol{C}(\boldsymbol{\nu}) = \left[\begin{array}{cc} \boldsymbol{0}_{3\times 3} & -\boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{\nu}_1 + \boldsymbol{M}_{12}\boldsymbol{\nu}_2) \\ -\boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{\nu}_1 + \boldsymbol{M}_{12}\boldsymbol{\nu}_2) & -\boldsymbol{S}(\boldsymbol{M}_{21}\boldsymbol{\nu}_1 + \boldsymbol{M}_{22}\boldsymbol{\nu}_2) \end{array} \right], \tag{3.6}$$

where $\boldsymbol{\nu}_1 = [u, v, w]^\top$, $\boldsymbol{\nu}_2 = [p, q, r]^\top$, and $\boldsymbol{S}$ is the cross product operator[2].

Note that the coriolis effect also is present for the added mass described in Section 3.1.1. This is included in $\boldsymbol{C}$ as follows:

$$\boldsymbol{C}(\boldsymbol{\nu}) = \boldsymbol{C}_{RB}(\boldsymbol{\nu}) + \boldsymbol{C}_A(\boldsymbol{\nu}). \tag{3.7}$$

As DP includes only low speed maneuvering or station keeping, the velocity is zero or very small, such that the coriolis term can be removed from the equation without much loss of precision. The coriolis term is left out in the vessel model utilized in the model-based LQR controllers. However, because of completion, the Matlab simulator described in Chapter 5.1, includes the coriolis effect.

[1]GNC Toolbox is a "Guidance Navigation and Control" Matlab® toolbox provided by Marine Cybernetics [1].

[2]The Cross-product operator $\boldsymbol{S}$: $a \times b = \boldsymbol{S}(a)b$.

3.1.3 Damping matrix

The damping matrix represents forces from skin friction, wave-drift damping and damping due to vortex shedding [14]. However, these are complex phenomena that are hard to model correctly. For DP operations, a linear damping matrix can be used [14]. While a quadratic damping matrix can be appropriate for high speed maneuvering. Note that the damping matrix is relative to $\boldsymbol{\nu}_r$, not $\boldsymbol{\nu}$, making this term also compensate for current effects. More on current modeling in Section 3.3.1.

The determination of both the linear and quadratic part of a damping matrix for a small craft should be possible with a systematically experimental approach, where one knows the engine output and the vessel speed. Linear and quadratic terms can then be estimated using regression analysis. Such an approach would also include aerial damping, which is otherwise not modeled. This experiment should be performed in calm water and calm winds.

Is is important to realize that the *center of dissipative forces* (CD)[3] not necessarily is located in the COG. From a DP point of view, this effectively means that the vessel will encounter a yaw momentum whenever encountering movement in the sway direction relative to the water masses, and likewise the other way around. This effect can be included in the damping matrix by introducing off-diagonal elements. As stated in [14], a 3-DOF damping-matrix will yield:

$$\boldsymbol{D} = \begin{bmatrix} -X_u & 0 & 0 \\ 0 & -Y_v & -Y_r \\ 0 & -N_v & -N_r \end{bmatrix}, \tag{3.8}$$

where, e.g., X_u represents the force generated in the X-direction by velocity in the u-direction. See SNAME definitions in Table 1.4 for more details. The values given in (3.8) is sufficient for 3-DOF DP. However, 6-DOF DP involves a more complicated damping matrix. A description of the 6-DOF DP can be found in [14].

3.1.4 Additional terms

A description of the additional terms in (3.1), being $\boldsymbol{g}(\boldsymbol{\eta})$, $\boldsymbol{\tau}$, $\boldsymbol{g}_0$ and $\boldsymbol{w}$.

[3]The *center of dissipative forces* is the point where the sum of all dissipative forces act, i.e., a surge movement will not induce any yaw acceleration if the CD is in the CG.

$\boldsymbol{\tau}$ - Actuator forces All forces that are due to actuator action. See Chapter 7.3 for a general discussion on thrusters. See Section 3.2 for a discussion on thruster dynamics.

$\boldsymbol{w}$ - Environmental forces In this case being the sum of wind and wave forces acting on the body. (Currents is handled in the damping matrix). See Chapter 3.3 for more on modeling of environmental forces.

$\boldsymbol{g}_0$ - Gravity forces Gravity is always acting in the CG. Defined by $\boldsymbol{g}_0 = m\boldsymbol{g}$, where $\boldsymbol{g}$ is the local gravity acceleration vector.

$\boldsymbol{g}(\boldsymbol{\eta})$ - Buoyancy forces Forces acting on the vessel from the water due to buoyancy. This force will get larger when the vessel displaces a larger volume of fluid, according to Archimedes. It will also set up a torque on the body according to the vessel angular configuration. When the vessel is at rest in calm waters, this force is equal and opposite to the gravitational force $\boldsymbol{g}_0$.

3.2 Actuator dynamics

3.2.1 Electrical thrusters

For small marine craft, a typical bow or stern thruster is an electrically-powered propeller positioned in a transverse hull tunnel. Upon demanding an actuator setpoint, it will not be reached before a certain amount of time has passed. This is due to that the motor rotor, gears and the propeller itself has mass, and will need to be accelerated. Backlash is present, but is typically minimal with electrical thrusters. Furthermore, the water initially present in the thruster tunnel also has mass, and must be accelerated by the propeller before the change in propeller RPM has any effect on the force acting on the vessel.

3.2.2 Combustion engine thruster

The main propulsion system of a small boat usually consist of one or more combustion engines. In addition to be more complex than electrical motors, they also in most cases have an extra degree of freedom, namely the option to change the direction in which the force is applied. Rudder actuators have their own dynamics, which will need to be determined in order to utilize the extra degree of freedom. Note that the control algorithm also must take into consideration that a rotatable (azimuth) thruster can be rotated.

The propulsion dynamics of a combustion engine includes throttle dynamics, gear backlash, and optionally the operation of a trolling gear. As frequent change of gears will represent a significant wear on clutch and gearbox, this should be considered when designing the controller. One can not simply treat a combustion engine the same way as an electrically powered thruster.

3.2.3 Dynamics approximation

The behavior of both electrical and combustion engines are nonlinear. This dynamics can however be estimated in order to achieve an adequate model precision. Section 11.1.1 in [14] suggest a simple approximation, that is defining three time constants in *surge, sway and yaw*, such that

$$\boldsymbol{A}_{thr} = -diag\{1/T_{surge}, 1/T_{sway}, 1/T_{yaw}\} \tag{3.9}$$

$$\dot{\boldsymbol{\tau}} = \boldsymbol{A}_{thr}(\boldsymbol{\tau} - \boldsymbol{\tau}_{com}), \tag{3.10}$$

where $\boldsymbol{\tau}_{com}$ is the commanded thrust.
This model has been augmented in order to apply the time-constant approach on each individual thruster. Furthermore, thruster saturation has been included in order to limit maximum and minimum force, such that the individual thruster dynamics can be described by:

$$\dot{\tau} = -\frac{1}{t_{tc}}(\tau - \tau_{com}), \tag{3.11}$$

where τ is thruster force, t_{tc} is thruster time constant and τ_{com} is thruster force setpoint.

For the combustion engine, that involves gearing, one could introduce a deadband when the sign of the actuator force changes, in order to simulate gearbox dynamics.

3.3 Environmental modeling

Counteracting environmental disturbances is more or less what DP is all about. This chapter describes methods for modeling of *current*, *waves* and *wind*. Modeling of environmental disturbances includes both modeling of the disturbance itself, and the interaction between the vessel and the disturbance. It is important to differentiate these problems. Both subjects are treated under each disturbance commented.

Vessel interaction is treated in two different ways; currents affect the vessel movements through the damping matrix $\boldsymbol{D}$, while waves and wind affect the vessel through $\boldsymbol{\tau}$, see (3.1). When developing control systems, the *principle of superposition* is commonly used as a fair approximation when dealing with wind and wave disturbances [14], such that:

$$\boldsymbol{w} = \boldsymbol{w}_{wind} + \boldsymbol{w}_{wave}. \tag{3.12}$$

The simulator discussed in Chapter 5.1 applies some of the methods discussed here.

3.3.1 Currents

Ocean currents can be very strong and have a significant effect on the vessel. However, they tend to vary slowly, such that one can assume that the change of current state is zero when working with DP applications. This study is on general DP, thus also including DP operations that can take place in a river, where current is very likely to be the most dominant source of disturbance. Moreover, it is important to realize that a vessel moving at constant speed in calm water is equal to a vessel performing station keeping when a current is present (when neglecting other disturbances such as wind).

Modeling

The modeling of a current can simply be done by assigning a velocity and a direction, as currents tend to be constant. Currents can also be modeled more complex, for example does [14] suggest a current model similar to the method of modeling wind described in Section 3.3.2, which uses a Gauss-Markov process.

Vessel interaction

The current effect on the vessel is a complex phenomena. As current in reality is water moving relative to the vessel, it makes sense to incorporate current forces in the damping matrix $\boldsymbol{D}$ (see (3.1)), rather than an extra force acting on the vessel, which is the method used when implementing wind and wave forces. Note that the damping term $\boldsymbol{D}$ is dependent on $\boldsymbol{\nu}_r$, instead of just $\boldsymbol{\nu}$. This is what makes the vessel automatically follow the current. The challenge of correctly modeling current effects on a vessel is thus a correct setup of the $\boldsymbol{D}$ matrix. More on this in Section 3.1.3.

3.3.2 Wind

Wind forces can have a dominant effect on the resulting environment force vector. Everyone that has tried to stay in the same position with a small boat knows that wind can give even small boats a significant speed due to wind forces. This is probably due to that small objects by nature will have a larger surface/weight ratio than a bigger object. These forces will naturally need to be compensated for in a proper DP system. The current method for handling this problem on larger vessels is by using feedforward control on the mean wind direction and moment. It is important that measurements from a wind sensor are properly filtered such that the DP algorithm only is asked to compensate for movements within its available bandwidth.

Wind is a complex 3D phenomenon [14]. However, modeling for DP applications can be simplified to 2D. Real wind rapidly varies both direction and momentum, and can be modeled in a proper way by, e.g., the NORSOK wind spectrum [23]. However, modeling only the mean value of the wind will not affect the simulation result much, since the vessel dynamics is much slower than the fluctuating component of the wind. Furthermore, an optional wind sensor will have to filter its measurements down to the mean value when used.

The same feedforward technology can be applied on small vessels. However, an alternative solution is to consider the wind force as a contributor to the resulting environmental force, and leave it for the controller to estimate.

It is generally considered unnecessary for the control system to compensate for wind gust because of the slower vessel dynamics. However, when working with smaller vessels, one cannot filter out as low frequencies as for bigger vessels. One can imagine a wind gust lasting for say 5 seconds. This is not enough to have any effect of interest on larger vessels, but could have a severe effect on smaller vessels.

The effective momentum acting on the vessel as a result of wind is dependent on the wind attack angle, due to the exposed area normal to the wind angle.

Modeling

Slowly varying variations in the mean wind velocity may be implemented by a 1st order Gauss-Markov Process [14]. Variation in wind direction can be implemented similarly:

$$\dot{\psi} + \mu\psi = w, \tag{3.13}$$

where ψ is wind direction [23]. This method is implemented in the simulator, and the effect is visualized by the fluctuating movement of the wind vector.

Vessel interaction

In a 3-DOF environment, wind force acting on the vessel can be expressed as

$$\boldsymbol{w}_{wind} = \left[X_{wind}, Y_{wind}, N_{wind}\right]^{\top}, \tag{3.14}$$

where $\boldsymbol{X}_{wind}$ and $\boldsymbol{Y}_{wind}$ is surge and sway force, while $\boldsymbol{N}_{wind}$ is yaw torque.

There are several models for wind-vessel interaction. However, they tend to be employed for larger vessels. An approach where the attack point of the wind vector relative to the vessel CG has been developed. The wind is modeled as a force affecting the vessel in this point, thus creating a momentum, due to that this point is offset from the CG. This has been implemented in the simulator, and yielded a result that appeared natural when comparing to experience with a small boat moving in a windy environment. The wind center of attach has been set 1 meter in front of CG in the default vessel implemented in the simulator.

Note: The example simulation 'EXAMPLE_windModel.m' shows a simple example where the vessel is initiated and affected by a constant wind in a constant direction. The wind point of attack can be adjusted by setting the property 'windAttackVector'.

In addition to yaw effect, the wind will have a varying effect in surge and sway, as a function of attack angle. This has not been implemented in the simulator.

Note: The wind only has effect relative to the vessels velocity, such that one should subtract vessel velocity when dealing with wind forces. However, one can assume that the wind velocity is much larger than the vessel velocity when working with DP operations, thus one can approximate the real wind speed to be equal to vessel wind speed [23].

3.3.3 Waves

Waves and wave-vessel interaction are very complicated phenomena, that play a crucial role in DP systems. Currents tend to be slowly varying, and thus easily predictable, while wind can easily be compensated for by feedforward control. Waves, on the other hand, is causing the toughest challenges within the field of DP. This consideration is even more important in respect to small-craft DP, as waves naturally have a bigger impact on the movement of a smaller craft than a larger vessel.

Waves can be described by their frequency spectra. It turns out that the wave spectra tends to vary according to the state of the sea. According to [14], it is observed that a *developing sea* starts with a spectrum with a peak at a relative high frequency. After the wind has stopped, the waves become longer and form a wave spectrum with a low peak frequency.

Modeling

Waves can be, e.g., modeled after the *Pierson-Moskowitz Spectrum*, which is a two parameter wave spectral formulation for fully developed wind-generated seas, as stated in [14]. The spectrum is described by:

$$S(\omega) = A\omega^{-5}exp(-B\omega^{-4}), (m^2s), \tag{3.15}$$

where the parameters A and B are described by:

$$A = 8.1 \cdot 10^{-3}g^2 = constant \tag{3.16}$$

$$B = 0.74\left(\frac{g}{V_19.4}\right)^4 = \frac{3.11}{H_s^2}, \tag{3.17}$$

where $V_19.4$ is the wind speed at a height of 19.4 meters above sea level and H_s is the significant wave height(mean of the one-third highest waves).

The emphModified Pierson-Moskowitz (MPM) Spectrum is recommended for simulation of open sea waves [14]. Values A and B are given by:

$$A = \frac{4\pi^3 H_s^2}{T_z^4}, B = \frac{16\pi^3}{T_z^4}, \tag{3.18}$$

where $T_z = 0.710T_0 = 0921T_1$

Vessel interaction

Forces induced by waves, $\boldsymbol{w}_{wave}$ (see (3.12)), can according to [14] be generated using either a force transfer function in a state-space formulation, or alternatively the *wave-frequency* (WF) motion can be simply added to the vessel motion after the principle of linear superposition, such that the total motion becomes:

$$\boldsymbol{y} = \boldsymbol{\eta} + \boldsymbol{\eta}_w. \tag{3.19}$$

As further stated in [14], *"linear wave response approximations are usually preferred by ship control system engineers, because of their simplicity and applicability"*.

Transforming a wave spectrum to vessel forces or force movement can be done by *Response Amplitude Operators* (RAO). E.g, a emphForce RAOs, which is applied in the Matlab Simulator uses the sea surface elevation to generate the 3-DOF force vector, which acts on the vessel from the waves.

Chapter 4

DP controllers

The DP controllers output is a *control vector* $\boldsymbol{\tau}$, that is calculated as follows:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{LQ} + \boldsymbol{\tau}_{FF}, \tag{4.1}$$

where $\boldsymbol{\tau}_{LQ}$ is generated by the model-based LQR controller and $\boldsymbol{\tau}_{FF}$ is generated by the feedforward control, which is based on data from the reference model. After calculation of the control vector $\boldsymbol{\tau}$, it is distributed to the available actuators as force setpoints (see Section 4.9). Three different LQR approaches has been discussed for feedback control. As the LQR feedback control is only applicable to station-keeping, a reference model with feed forward control has been included to provide low speed maneuvering functionality. A comparison on controller performance is provided in Chapter 5.

4.1 DP observer

The extra challenges related to state estimation that comes with smaller craft because they are small, is an important issue. A thorough discussion on general sensor setup and considerations for general DP vessels can be found in [11], but is also covered in [14] and [23], among others.

As smaller craft naturally have higher pitched dynamics than larger craft, we can expect an augmented level of estimation challenges. Picture a small lightweight boat present at sea, facing waves with a significant wave height of 3 meters. The movement of the boat will be significant in all 6 degrees of freedom. However, a larger vessel is less likely to notice much of the extra movement caused by such waves. This has at least two consequences that will need attention:

- The measurement and estimation system will need to operate faster that what is required on a larger vessel with slower dynamics. This is because a small vessel with its lower mass per area than its larger sister will be able to accelerate faster in all six degrees of freedom.

- The vessel is likely to encounter significantly larger roll and pitch movements than its bigger sister. This will result in the need for sensors located far away from the center of rotation to be accurately compensated. Picture a GPS antenna located on about 3 meters higher up than the axis of the experienced roll rotation. It will indicate that the vessel moves around 4 meters from its extreme port point to its extreme starboard point given that the vessel experiences a roll movement of 45°to each side. If this is not compensated, the sensor will assume that the vessel surge position is oscillating, when in reality its stationary with an oscillating roll movement. This problem is also present on larger vessels, but with higher amplitude and lower frequency.

Counteracting first order wave loads is neither possible nor desirable when in DP operation. Such high frequency movements can be filtered out with a low-pass and/or notch-filters. This was done in the first DP implementations in the sixties (see 2.1). However, using such filters involves introducing a phase-shift, such that the phase-margin of the controller is reduced, and possibly rendered unstable.

As a result, model-based observers have been introduced. A model-based observer utilizes a dynamical model of both the vessel and the disturbances. Using the control vector $\boldsymbol{\tau}$ from the controller, the observer is able to estimate vessel movement caused by first order wave loads, and can thus provide a properly filtered controller input. Furthermore, measurements of vessel velocities are not always available. As the observer includes a vessel model, it can estimate the full state vector of the vessel, including velocities.

The observer used in this thesis is the nonlinear passive observer described in [14]. This observer utilizes a nonlinear-3-DOF vessel model with a first-order damping term:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\phi)\boldsymbol{\nu} \tag{4.2}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{D}\boldsymbol{\nu} = \boldsymbol{\tau} + \boldsymbol{R}^T(\phi)\boldsymbol{b} + \boldsymbol{w}_3, \tag{4.3}$$

where $\boldsymbol{\eta} = [x, y, \phi]^T$, $\boldsymbol{\nu} = [u, v, r]^T$, $\boldsymbol{b}$ is a vector of bias terms, and $\boldsymbol{w}_3$ is a vector of zero-mean Gaussian white noise processes.

The first-order wave response model is modeled as:

$$\dot{\boldsymbol{\xi}} = \boldsymbol{A}_w\boldsymbol{\xi} + \boldsymbol{E}_\omega\boldsymbol{w}_1 \tag{4.4}$$

$$\boldsymbol{\eta}_\omega = \boldsymbol{C}_\omega\boldsymbol{\xi}, \tag{4.5}$$

where $\boldsymbol{\xi}$ is the state vector, $\boldsymbol{w}_1$ is a vector of zero-mean Gaussian white noise, $\boldsymbol{A}_\omega$, $\boldsymbol{E}_\omega$, and $\boldsymbol{C}_\omega$ are constant matrices. The first-order wave-induced motion is modeled using linear superposition; $\boldsymbol{\eta}_\omega$ is the first-order contribution to the vessel position and heading.

The observer also encompasses a model for slowly-varying environmental disturbances caused by second-order mean and slowly varying wave, current, and wind loads. This is modeled as a first-order Markov model:

$$\dot{\boldsymbol{b}} = -\boldsymbol{T}^{-1}\boldsymbol{b} + \boldsymbol{w}_2, \tag{4.6}$$

where $\boldsymbol{T}$ is a user specified diagonal matrix of positive bias timeconstants, and $\boldsymbol{b}$ is the force acting on the vessel from the slowly varying environmental disturbances

When the above statements are combined, one gets the following model of the vessel and the disturbances:

$$\dot{\boldsymbol{\xi}} = \boldsymbol{A}_w\boldsymbol{\xi} \tag{4.7}$$
$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(y_3)\boldsymbol{\nu} \tag{4.8}$$
$$\dot{\boldsymbol{b}} = -\boldsymbol{T}^{-1}\boldsymbol{b} \tag{4.9}$$
$$\boldsymbol{M}\dot{\boldsymbol{\nu}} = -\boldsymbol{D}\boldsymbol{\nu} + \boldsymbol{R}^T(y_3)\boldsymbol{b} + \tau \tag{4.10}$$
$$\boldsymbol{y} = \boldsymbol{\eta} + \boldsymbol{C}_w\boldsymbol{\xi}. \tag{4.11}$$

The resulting observer equations are described by:

$$\dot{\hat{\boldsymbol{\xi}}} = \boldsymbol{A}_\omega\hat{\boldsymbol{\xi}} + \boldsymbol{K_1}(\boldsymbol{\omega}_0)\tilde{\boldsymbol{y}} \tag{4.12}$$
$$\dot{\hat{\boldsymbol{\eta}}} = \boldsymbol{R}(\hat{y}_3)\hat{\boldsymbol{\nu}} + \boldsymbol{K}_2\tilde{\boldsymbol{y}} \tag{4.13}$$
$$\dot{\hat{\boldsymbol{b}}} = \boldsymbol{T}^{-1}\hat{\boldsymbol{b}} + \boldsymbol{K}_3\tilde{\boldsymbol{y}} \tag{4.14}$$
$$\boldsymbol{M}\dot{\hat{\boldsymbol{\nu}}} = -\boldsymbol{D}\hat{\boldsymbol{\nu}} + \boldsymbol{R}^T(y_3)\hat{\boldsymbol{b}} + \tau + \boldsymbol{R}^T(y_3)\boldsymbol{K}_4\tilde{\boldsymbol{y}} \tag{4.15}$$
$$\hat{\boldsymbol{y}} = \hat{\boldsymbol{\eta}} + \boldsymbol{C}_w\hat{\boldsymbol{\xi}}, \tag{4.16}$$

where $\tilde{\boldsymbol{y}} = \boldsymbol{y} - \hat{\boldsymbol{y}}$ is the estimation error, hence $\boldsymbol{y}$ is the measured vessel position and heading. This data is fetched from the GPS in conjuction with the INS unit. Furthermore, $\boldsymbol{K}_1(\omega_0) \in \Re^{6x3}$ and $\boldsymbol{K}_{2,3,4} \in \Re^{3x3}$ are observer gain matrices, and $\hat{\boldsymbol{\eta}}$ is the estimated low-frequency motion, which is used as input to the DP controller.

The observer implementation utilized in the thesis is the Simulink® - based DP observer available in [1] (see Appendix B for the Simulink® implementation).

4.1.1 Observer tuning

The observer requires tuning for vessel and environmental model dynamics. The inertia matrix and the first order damping matrix in 3-DOF is sufficient for the vessel model in order to obtain acceptable performance. Values are obtained from model-identification in Section 7.4.

In order to verify the integrity of the vessel model used in the observer, a comparison with the vessel model implemented in the DP simulator described in Section 5 was performed. The observer estimated vessel position was fed back to the observer input as the measured position, hence $\tilde{\boldsymbol{y}} = 0$, such that the observer vessel dynamics were isolated. Furthermore, two vessels with the same control vector $\boldsymbol{\tau}$ was created, where one of the vessels used the observer vessel model, and the other used the simulator vessel model. This experiment confirmed that the observer vessel model was acting sufficiently equal to the simulator vessel model at lower vessel velocities.

4.2 DP reference model

When under DP operation, steps in position and attitude reference are undesired, due to rapidly changing actuator setpoints. Such behavior tears the actuator, is more energy consuming, and renders the system less robust. The lack of robustness is due to that actuator saturation is unmodeled in the linear vessel model applied in the LQR controller. Furthermore, the use of a reference model makes the vessel movement predictable. The rate of movement can be adjusted with controlling the natural frequency of the reference model.

A vessel reference model is introduced in order to make sure the controller is always facing a smooth-changing reference. A third order reference model for filtering of the desired reference position and attitude $\boldsymbol{r}_d$ [14].

A second order filter can also be applied, however, a third order filter includes the acceleration as part of the state-vector, thus making it easily available for vessel feed forward control. Furthermore, a second order filter introduces steps in the reference model acceleration whenever facing a change in position and attitude reference, while the third order approach guarantees a smooth acceleration.

As stated in [14], the state-space representation of third order reference model can be described as follows:

$$\boldsymbol{A}_d = \begin{bmatrix} \mathbf{0} & \boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{I} \\ -\boldsymbol{\Omega}^3 & -(2\boldsymbol{\Delta} + \boldsymbol{I})\boldsymbol{\Omega}^2 & -(2\boldsymbol{\Delta} + \boldsymbol{I})\boldsymbol{\Omega} \end{bmatrix}, \tag{4.17}$$

$$\boldsymbol{B}_d = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \boldsymbol{\Omega}^3 \end{bmatrix}, \tag{4.18}$$

where $\boldsymbol{\Delta}$ is a diagonal matrix of relative damping ratios, $\boldsymbol{\Omega}$ is a diagonal matrix of natural frequencies. A critically damped reference model is a good choice, as one typically wants a fast response that does not overshoot. Thus $\boldsymbol{\Delta} = \boldsymbol{I}$ is a good setting. In a more compact form, the filter can now be described by:

$$\boldsymbol{\eta}_d^{(3)} + 3\boldsymbol{\Omega}\ddot{\boldsymbol{\eta}}_d + 3\boldsymbol{\Omega}^2\dot{\boldsymbol{\eta}}_d + \boldsymbol{\Omega}^3\boldsymbol{\eta}_d = \boldsymbol{\Omega}^3\boldsymbol{r}^n, \tag{4.19}$$

where $\boldsymbol{r}^n$ is the target reference.

This reference model is implemented in the simulator described in Chapter 5. Figure 4.1 illustrates the reference model response in respect to arbitrary change in setpoint.

4.2.1 Reference model saturation

The reference model must stay within the vessel bandwidth, such that normal operation leaves actuators within their normal operating limits. As this is a linear model, a sufficiently large step in reference position and attitude will result in accelerations that the vessel can not handle without falling behind. This is addressed by implementing saturation limits on the velocity and the acceleration vector [14].

However, when implementing saturation limits on the state vector of the reference model, one must take into account that the state-vector is used for vessel feed forward control. Hence one must take into consideration that, e.g., saturating vessel velocity while not saturating acceleration will result in an incorrectly applied feed forward control, as the feed forward term will take both velocity and acceleration into account in order to calculate the correct feed forward control vector.

As the reference step in vessel position can be of arbitrary size, the reference step in yaw can never exceed π radians, given a proper implementation.

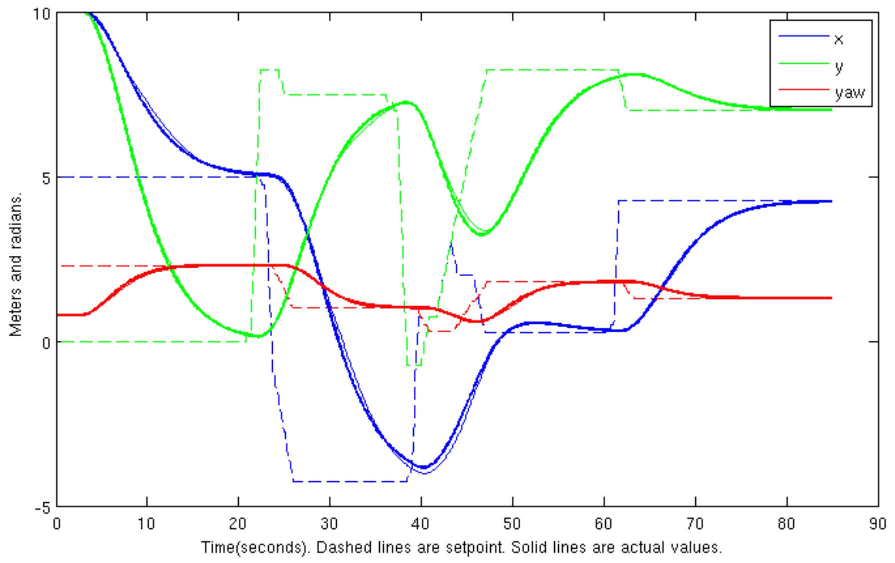


Figure 4.1: Thick lines represents the reference model, thin line represents the vessel model, and dashed lines represents the operator reference input. Plotted using the CyberShip2 model-parameters. The reference model is critically damped, and natural frequencies are: [0.4 0.4 0.6] rad/sec in x, y and yaw.

This fact can be utilized to leave out saturation of yaw velocity and acceleration, as the natural frequency of the reference model yaw dynamics can be designed to handle a full π radians change in setpoint.

4.2.2 Reference model initialization

The reference model dynamics must be enabled only when the DP controller is enabled. This ensures that the reference model does not drift away from the actual vessel configuration when not in DP operation. Hence the reference model should be reset to match the vessel model whenever the DP controller is enabled.

Note that it is important to include velocities when initializing the reference model to the vessel model. When using a third degree reference model, one should also set the reference model initial acceleration equal to the vessel acceleration. Leaving the reference model initial velocity and acceleration to zero can result in unnecessary large controller reference steps in situations where the reference model is turned on while the vessel is in motion. This has the most impact in situations where the reference model target position is located in the opposite direction of the initial vessel velocity vector.

4.3 Feedforward control

When the operator demands a step in vessel position and heading, the reference model generates a smooth trajectory to this new configuration. As both velocity and acceleration of the reference model is known, this can be utilized as input to the feedforward controller. The feedforward controller utilizes the vessel model to calculate the actuator setpoints that are most likely to generate the same motion as the reference model.

This reversed usage of (3.1) results in that the feedback controller only needs to control the deviation caused by model error and external forces, instead of performing the actual transfer, which it is not intended to do.

A nonlinear feedforward controller has been implemented in the Matlab Simulator. Vessel acceleration, first, second, and third order damping is used to calculate $\boldsymbol{\tau}_{FF}$ (see (4.1)). In order to properly show the effect of the feedforward controller, a simulation has been done with the feedback controller switched off. See figure 4.2.

Notice how the feedforward controller is diverging from the reference model in Figure 4.2. This is due to that actuator dynamics are unmodeled in the feedforward control. Furthermore, whenever an actuator is saturated, the vessel will fall behind, as feed forward by nature has no knowledge of the actual vessel position. However, when in situations with no currents, waves or wind, the reference model can be tuned and saturated such that the feedforward control never will return a $\boldsymbol{\tau}_{FF}$ which results in actuator saturation. Note that as this is a simulation, the feedforward uses a perfect vessel model, which never will be the case in a real-world situation.

Simulation reveals that though deviations due to actuator dynamics makes the vessel fall behind the reference model, the vessel will still end up close to the final position. This is due to that the lag in initial acceleration is canceled by the lag in deceleration when the vessel is about to reach its final position. This reduces the need for actuator dynamics compensation in the feedforward controller. This is especially true when performing station keeping, as the reference model is not designed to follow a trajectory, but rather just give a smooth transfer from one vessel configuration to the next.

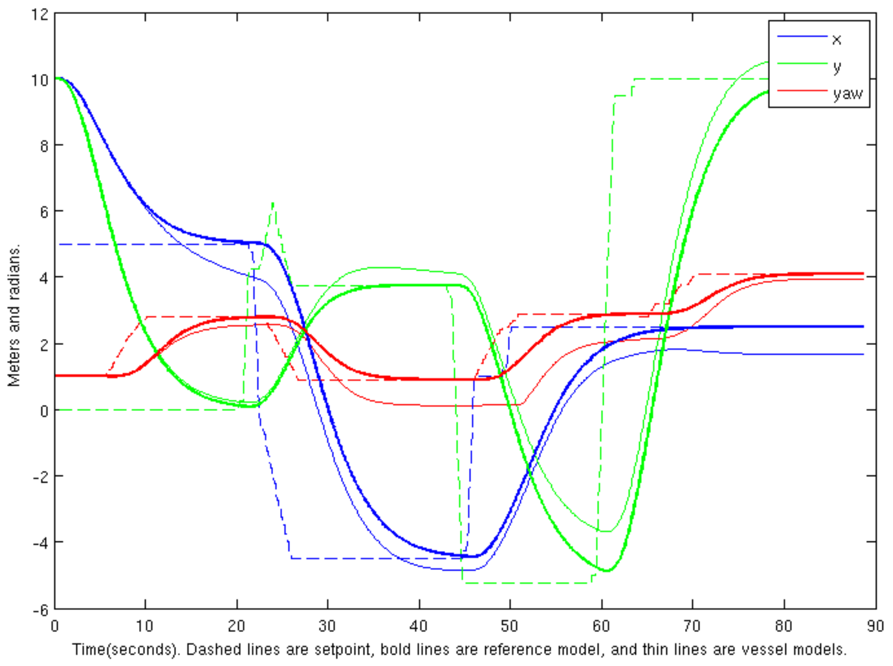


Figure 4.2: Feedforward control without external disturbances. Note that feedback control has been removed from this simulation. Plotted using the CyberShip2 model parameters. The reference model is critically damped, and natural frequencies are: [0.4 0.4 0.6] rad/sec in x, y and yaw.

4.4 Linear quadratic optimal control

The Linear Quadratic Regulator (LQR) is a feedback controller that can compute the optimal control signal, given a linear dynamic system, and a cost, expressed as a quadratic polynomial. The cost function includes the matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$, that are weighting matrices for state deviation and control action. An LQR controller has many similarities to a PID controller, however, the difference being that the method of tuning is abstracted to the level of tuning the weight matrices instead of tuning the actual gain matrix. The LQR controller assumes one has access to the complete state vector. However, the state vector can be estimated by an *observer* (see Section 4.1).

In conjunction with feedforward control(see Section 4.3 , LQR is the chosen feedback controller treated in this thesis. LQR is the feedback controller of choice due to the combination of simplicity and applicability.

This section first includes a section on the theory behind the LQR controller. Furthermore, three different versions of the controller have been implemented, and are presented here. Firstly, the simple LQR controller that considers vessel attitude and velocity as states. The following two extends this controller with adding actuator dynamics, and integral action. The motivation for implementing all three instead of the most complicated controller is to see what one can gain from these augmentations. The theoretical background for these augmentations are presented where they are first applied.

4.4.1 Mathematical background

The Linear Quadratic Control (LQR) approach solves the positioning problem by minimizing a cost-function dependent on the deviation from the given reference state. This method is described in [14]. Given a linear time-invariant(LTI) vessel model expressed as:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\underbrace{(\boldsymbol{\tau}_{LQ} + \boldsymbol{\tau}_{FF})}_{\boldsymbol{\tau}_{com}}, \tag{4.20}$$

where $\boldsymbol{\tau}_{FF}$ is the commanded force vector from wind feedforward calculations, and $\boldsymbol{\tau}_{LQ}$ is the commanded force from the optimal feedback, calculated using LQR..

In order to apply the LQR approach to DP, a linearized version of 3.1 must be developed. In order to remove the non-linearities imposed in the vessel model, the vessel state is rotated to a vessel parallel frame, such that the relation

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{4.21}$$

can be approximated by

$$\dot{\boldsymbol{\eta}}_p = \boldsymbol{J}^T\boldsymbol{\nu} \Rightarrow \dot{\boldsymbol{\eta}}_p \approx \boldsymbol{\nu}, \tag{4.22}$$

as the rotational matrix $\boldsymbol{J}$ can be approximated as the identity matrix.

The vessel can now be modeled according to the following linear state-space model:

$$\dot{x} = Ax + B\boldsymbol{\tau} + \Gamma w \tag{4.23}$$
$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} + \boldsymbol{v}, \tag{4.24}$$

where:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{\eta}^T & \boldsymbol{\nu}^T \end{bmatrix}^T \tag{4.25}$$

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{0}_{3x3} & \boldsymbol{I}_{3x3} \\ \mathbf{0}_{3x3} & -\boldsymbol{M}^{-\mathbf{1}}\boldsymbol{D}_L \end{bmatrix} \tag{4.26}$$

$$\boldsymbol{B} = \begin{bmatrix} \mathbf{0}_{3x3} \\ -\boldsymbol{M}^{-1} \end{bmatrix} \tag{4.27}$$

$$\boldsymbol{\Gamma} = \begin{bmatrix} \mathbf{0}_{3x3} \\ -\boldsymbol{M}^{-1} \end{bmatrix}, \tag{4.28}$$

The control objective is to stabilize $\boldsymbol{x}$ to zero. Thus, $\boldsymbol{\tau}_{LQ}$ can be computed by minimizing:

$$J = min\left\{\frac{1}{2}\int_0^T (\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{\tau}_{LQ}^\top \boldsymbol{R}\boldsymbol{\tau}_{LQ})d\tau\right\}, \tag{4.29}$$

where $\boldsymbol{R}$ and $\boldsymbol{Q}$ are cost matrices for control action and state deviation, respectively.

The steady-state solution to this problem is [?]:

$$\boldsymbol{u} = \underbrace{-\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P}_\infty}_{\boldsymbol{G}}\boldsymbol{x} \tag{4.30}$$

$$\boldsymbol{P}_\infty\boldsymbol{A} + \boldsymbol{A}^T\boldsymbol{P}_\infty\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P}_\infty + \boldsymbol{C}^T\boldsymbol{Q}\boldsymbol{C} = 0, \tag{4.31}$$

where $\boldsymbol{P}_\infty = \lim_{t\to\infty}\boldsymbol{P}(t)$.

The LQR controller theory is based on that the reference position is fixed. Whenever the reference position is moving, one can not guarantee that the LQR solution is the optimal solution for the linear system. Hence, using LQR as an isolated controller should only be used when performing station keeping. In order to use the LQR controller to low-speed maneuvering, the concept of feedforward control is used in addition to the feedback LQR controller. The feedforward control produces the control vector $\boldsymbol{\tau}_F F$ that will perform most of the control action. The LQR is left to control the deviation error. Even though the application of a feedforward control significantly reduces the error gained from erroneous usage of the LQR controller, it is still not theoretically feasible due to that low-speed maneuvering implies that vessel acceleration and speed can change. However, the LQR controller is still applied to low-speed maneuvering, as the weighting matrices of the LQR controller can be configured such that the controller becomes robust enough to handle the small deviations when performing low-speed maneuvering. Note that when using both a feedforward and a feedback controller for low-speed maneuvering, it is important to make sure that the the LQR controller does not try to approach zero speed, but rather the speed of the reference model (which is also used by the feedforward control). Using the speed of the reference model will reduce the error gained from performing low speed maneuvering.

State design

The LQR controller is designed to calculate feedback gains from the system state, thus every quantity subject to weighting should be represented as a state or an input. For example, using a 3-DOF vessel state vector, one can only perform control in respect to vessel position, attitude, and velocity in surge, sway and yaw. In order to perform more advanced control, the state vector must be augmented.

4.5 Simple LQR control

The simple LQR control uses vessel position and speed as reference. This approach is comparable to a nonlinear PID-controller. The equations for this controller is equal to the one presented in Section 4.4.1. The weighting matrix (R) is now weighting the thrust setpoints in vessel surge, sway and yaw directions. The state vector is here of size 6.

4.6 LQR with actuator dynamics

In order to include actuator dynamics, the state vector must be augmented to 9 states, where the first six states are the same states as presented in the simple LQR controller, and the new three states represents the current force acting in the vessel surge sway and yaw directions. This augmentation allows the controller to weight the actual force tat is applied by the thrusters, instead of the setpoint that is given to them. Furthermore, the controller becomes aware of the dynamics presented in the thrusters, such that one can design a more aggressive controller while still remain stable.

When designing the system matrix $\boldsymbol{A}$, one has to use the time constant of the actuators in surge, sway and yaw directions. Simulation tests reveals that setting the time constant to the correct value, e.g., setting the value equal to the value used in the vessel model, make the controller less stable. This artifact is due to that the LQR controller is not aware of the saturation limits of the actuators. A high time constant will naturally make the controller set an even higher setpoint in order for the response to be quicker. However, setting a setpoint that is much larger than the saturation limit of the actuators makes the dynamics behave significantly different. Simulation revels that setting the time constant of the thrusters in the controller vessel model to a quicker response than the actual response gives the best result.

4.7 LQR with actuator dynamics and integral action

This controller extends the latter controller with three additional states, which is the integral term in x, y and yaw direction, such that the state vector to control is now of size 12. The benefit of integral action is that one is able to encounter slowly varying disturbances that otherwise would have resulted in a standard deviation. This effect comes forward in Figure 5.6.

However, when using the integral term, one has to make sure that the integral term frequency is at least an order of magnitude lower than the velocity and position terms. This is necessary in order to avoid too much overshoot. Again, this results in that the integral action is slow, and will require a notable amount of time to encounter stronger standard deviations.

In order to reduce the effect of slow integral action dynamics, the integration is performed in the NED-frame, as a standard deviation encountered in a DP situations is likely to be caused by the mean of current, wind and second order wave forces. Integrating in the NED-frame results in that the integral action does not have to change whenever the vessel is rotating. The integral term is simply rotated to the BODY-frame before it is applied.
¨ Another issue that comes with integral action, is anti-windup. If a system does not include anti-windup, the system will be rendered unstable after a certain time, given that the vessel is in a situation where it can never reach its setpoint. Solutions to this is to include a anti-windup setting that is low enough to ensure stability. Furthermore, adding to the integral term when actuators are saturated is meaningless, as they are already saturated. The Matlab simulator is programmed to stop integrating whenever an actuator is saturated.

The method of integral action is described in detail in [14]

4.8 Low-speed maneuvering

Low-speed maneuvering is applicable when maneuvering in confined areas is required, such as when in a docking situation. The maneuvering task often includes maneuvering the vessel after a predefined trajectory.

Feedforward control is essential in low speed maneuvering, as the feedback LQR is only intended to correct erroneous movements due to model error and disturbances, the feedforward control uses the trajectory acceleration and speed to calculate the required control vector $\boldsymbol{\tau}_F F$. This vector can be calculated directly from the vessels damping properties.

4.9 Thruster allocation

The thruster configuration considered is the Viknes 830 setup, that includes a main engine for forward/backward thrusts, a bow thruster and a stern thruster. The output from the DP algorithms described above has the torque/force-vector $\boldsymbol{\tau}$ as output. A mapping to thruster setpoint is needed.

The problem of thruster allocation is notably harder to solve when azimuth thrusters are present. In the case of small-craft maneuvering, the vessel will in most cases be equipped with fixed direction tunnel thrusters, in addition to one or two main propulsion thrusters. This is also the case for the Viknes 830 vessel.

Thruster allocation without considering thruster saturation limits can be done by applying the pseudo-inverse method (see Section 11.2.1 in [14]).

$$\boldsymbol{\tau}_{com} = \boldsymbol{T}(\boldsymbol{\alpha})\boldsymbol{f} \tag{4.32}$$

$$\boldsymbol{f} = \boldsymbol{K}\boldsymbol{u}, \tag{4.33}$$

where $\boldsymbol{\alpha}$ is the vector of azimuth angles[1]. $\boldsymbol{T}$ is the geometrical transformation from the thrust applied in each thruster to $\boldsymbol{\tau}_{com}$. Due to the pseudoinverse, it is applicable even though one has more thrusters than degrees of freedom. The final thrust vector is given by:

$$\boldsymbol{u} = \boldsymbol{K}^{-1}\boldsymbol{T}^{\dagger}(\boldsymbol{\alpha})\boldsymbol{\tau}_{com}, \tag{4.34}$$

$$\boldsymbol{T}^{\dagger}(\boldsymbol{\alpha}) = \boldsymbol{W}^{-1}\boldsymbol{T}^{\top}(\alpha)[\boldsymbol{T}(\alpha)\boldsymbol{W}^{-1}\boldsymbol{T}^{\top}(\boldsymbol{\alpha})]^{-1}, \tag{4.35}$$

where $\boldsymbol{W}$ is the thruster weighting matrix, and $\boldsymbol{T}^{\dagger}$ indicates the pseudoinverse of $\boldsymbol{T}$. This method is implemented in the simulator described in

[1]The azimuth angle is the current rotation angle on a rotatable thruster.

Chapter 5. Note that this method is applied on the thruster setpoint, not on the thruster force, such that thruster dynamics is also considered.

Chapter 5

Simulation results

This chapter presents simulation results from the DP LQR controllers that have been implemented. Simulations has been performed in the Matlab DP Simulator that has been developed in order to provide a functional development environment for the thesis.

A properly identified vessel model is required in order to achieve probable results. Accurate vessel identification of a small craft is hard to find in the literature, therefore, model data from the model-vessel CyberShip II, owned by NTNU, has been used. Data are taken from [22], which performs a model identification analysis of the CyberShip II vessel. The following data were used in the simulator:

- Inertia matrix
- First, second, and third order damping matrix
- Vessel dimension (for correct display of the vessel in the simulator)

5.1 Matlab simulator

A Matlab DP simulator has been created to provide a functional development environment for the thesis. The simulator presented here is the extended version of the simulator developed during the pre-project for this thesis.

While making this simulator, ease of use, intuitive graphical display and modularity have been prioritized criterions. The Matlab simulator features the following functionality:

- Simulation of an arbitrary number of vessels simultaneously, using an easy-to-understand visualization of vessel, reference position, and reference model position.

- Thruster dynamics are modeled using individually configurable time-constants and saturation values.
- Each vessel can have individual properties such as inertia matrix, first, second, and third order damping matrices
- Environmental disturbances such as wind and current can be added individually to each vessel.
- Easy-to-understand graphical presentation of thruster setpoints and the force acting on the vessel
- Each vessel can be configured with an integrator of choice, in order to compare integrator performance.
- Each vessel can be configured with a DP controller of choice, in order to compare controller performance.
- DP Controller reference position and attitude can be updated runtime using an easy-to-use arrow-keys implementation.
- Simulator is easily configurable, due to the applied object-orented approach.
- The simulator includes a Simulink s-function that can read vessel states and set thruster setpoints. This functionality makes the simulator available to Simulink applications. This feature is used to connect to the hardware installed on the Viknes 830 vessel, such that one can import the actual vessel position and heading (or likewise position and heading from the HIL-simulator), hence making comparison on real world dynamics and simulator dynamics easy. This also allows for easy model identification

  This functionality also allows for the inclusion of a Simulink based vessel observer (see the Simulink diagram enclosed as Appendix B).

5.1.1 Vessel dynamics

The vessel dynamics applied in the simulator is described by (3.1). This equation is rearranged such that one get an expression for $\dot{\mathbf{x}}$, where $\boldsymbol{x}$ is the 3-DOF state vector. Note that $\boldsymbol{A}$ in (5.1) is time variant, since it is dependent on the current vessel rotation relative to the world-frame.

$$\dot{\mathbf{x}}_{k+1} = \mathbf{A}_k \mathbf{x}_k + \mathbf{B}\mathbf{u}_k. \tag{5.1}$$

First, second, and third order damping term is modeled in the simulator, in order to also provide proper response at higher speeds.

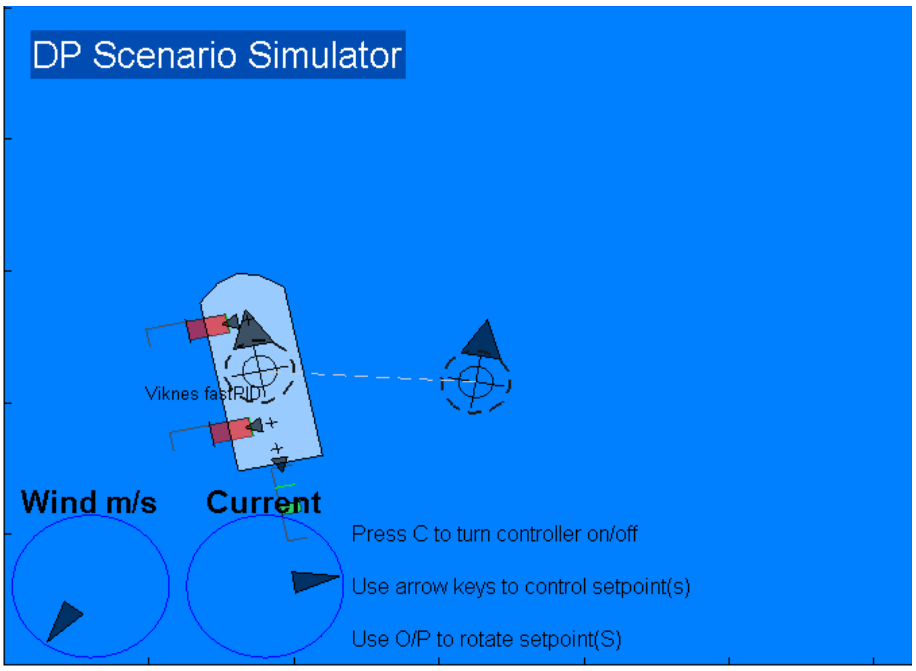

DP Scenario Simulator
Viknes fastRIDT
Wind m/s
Current
Press C to turn controller on/off
Use arrow keys to control setpoint(s)
Use O/P to rotate setpoint(S)

Figure 5.1: Simulator GUI example.

5.1.2 Environmental dynamics

Waves

Waves are implemented using an *Force RAO* approach.The wave spectrum is calculated using the MPM spectrum, due to reflect waves that are likely to be present in the northern sea, as this is the proposed operating area for the USVs developed at Maritime Robotics AS.

In order to generate a realistic wave response, wave forces have been calculated for a vessel with the dimensions of the Viknes vessel, and furthermore converted to forces for the CyberShip II model, which is used in the simulator. The conversion is done using the *Bis-system*, which is described in [14]. The Bis-system can be used for zero speed, which is crucial for a DP application.

Furthermore, in order to make the simulation appear as it would have done for the Viknes vessel, the time is scaled from the CyberShip2 vessel to the Viknes vessel (using the Bis-system), such that the rapid movements of the small Cybership2-vessel can be compared to the slower movements of the larger Viknes 830-vessel.

5.1.3 Observer

The simulator features an observer. The observer used is a modified version of the 'Passive DP wave filter' that is available with the GNC toolbox [1]. As this observer is designed for low-speed maneuvering, the built-in vessel model accepts only the first order damping term. However, in order to enhance the performance at higher speeds, this observer is augmented to a nonlinear version with second-order and third-order damping matrices included.

The plot in Figure 5.2 displays a two minute simulation of the CyberShip2 vessel model freely drifting under the influence of a wave load consisting of first and second order wave loads. The wave load peak frequency is set to 0.5 rad/sec, giving a period of 2 seconds. Notice how this simulation was started with the vessel of 3 meters in the x-position in order to illustrate how the observer quickly diverges to the correct position. Note that the observer should normally be initialized with the vessel in origo.

5.1.4 Equation solver

Numerical simulation is always a tradeoff between simulation time and simulation precision. However, if one can be certain that the model to be simulated is not 100 % accurate, a very accurate calculation makes little sense, as the bottleneck will be the mathematical model. In our case, there are great chances that both the mathematical model is inaccurate, and also the methods for solving equations to simplified to a level that justifies an equation solver without a particularly high rate of precision. On the other hand, one must be careful not to allow solver timesteps that renders the simulation unstable.

Rather that using a specified solver for the whole simulator, it is been designed such that one can choose a custom solver for each vessel. This can be specified individually for each vessel. This can be useful for example if one would like to measure the solvers influence on the simulation. One could for example design a scenario with two equal vessels, exposed to equal environmental conditions, but with different solvers. This is useful for determining the maximum stepsize that one can allow in order to perform a trustworthy simulation. A reduction in stepsize is directly comparable to a faster simulation time, or likewise a faster framerate when simulating in real time. At the current stage, the simulator is able to perform realtime simulations of no more than four or five vessels on a regular computer of todays standard. As mentioned in Section 5.1.6, this performance can be severely increased by using, i.e., a C++ implementation instead of an m-script-implementation.

Currently, there are implemented three different solvers for the simulator. These are the standard Euler solver, Matlab's built in ODE45[1] solver, and also Matlab's built in ODE23 solver. A particular solver can be assigned to a vessel upon creation. An example follows:

```
%Add a new vessel with a Euler integrator
scenario = addVessel(scenario,...
'state', [10 10 0   0 0 pi/4   0 0 0   0 0 0]', ...
'refState', [14 10 0   0 0 2.3   0 0 0   0 0 0]', ...
'integrator', 'euler', ..
'name', 'EulerShip);
```

This example should be applied such as described in Appendix D.

5.1.5 Graphical presentation

One of the main goals with the simulator has been the visual appearance. The motivation is that a good animation of the scenario can be a good supplement to standard plots. This section explains the graphical elements in the simulator.

The overall appearance of the simulator is as shown in Figure D.2. A detailed explanation follows:

5.1.6 Development environment

Matlab has been chosen as development environment for the simulator due to its widespread acceptance, ease of use, and the fact that much of what has been previously done within this field involves Matlab. E.g, the GNC toolbox[1] is only available as a Matlab implementation.

Matlab includes the option to use Simulink [2] and/or an m-script approach. While Simulink® often allows for a faster development process because of its well designed graphical user interface, the m-script approach also has its strengths. It can be hard to keep Simulink® applications ordered in a clean and orderly way when building applications of larger size. Also, because the Simulink® approach does not include coding, the software is writing the code for the programmer - and is not giving you the same level of control.

[1]ODE45, or Dormand Prince 5-4, is the default used in Matlab Simulink® , among others.

[2]Simulink® is an environment for simulation and Model-Based Design for dynamic systems using an interactive graphical environment, which is fundamentally different than writing m-code.

While writing m-code is more time-consuming, it gives you more control on how your program is structured. Matlab allows for object oriented programming, and thus all the benefits that comes with this paradigm. Both in respect to coding structure, ease of expansion, and the reduced time required for others to understand the code. Because of this, I've chosen to implement the simulator as an m-script-application using a object-oriented approach.

An additional argument in favor of the m-script approach, is that m-script code is relatively easily transformable to other languages, such as C++. This is an important property, due to that m-scripts tends to be orders of magnitude slower in execution that a properly implemented C++-application. A typically offshore scenario can easily become highly complex as one i.e., adds several vessels and complex environmental dynamics. The better performance of C++-applications can also be used to reduce solver steps, such that accuracy increases, or to perform simulations faster than realtime.

Matlab 7.4.0 (R2007a) is the version used throughout development. The complete simulator source code is available on the enclosed DVD (see Appendix A). A tutorial on how to use the simulator is included in Appendix E, and a more in-depth description on simulator development is included in Appendix D.

5.2 Simulation results

This section presents simulations done using the three proposed controllers in the set of scenarios given beneath. These scenarios share the following properties:

- All simulations are done using parameters identified for the CyberShip II model vessel (see [22]).
- The LQR controllers are tuned by manually optimizing the weighting matrices R and Q (see Section 4.4) by comparing plots and using the visualization, in order to produce the most stable and yet responsive result.

The presentations is finalized with a comparison on results.

5.2.1 Scenarios

Two DP scenarios are considered. Note that in order to see a notably difference on the three controllers, environmental load must be present, as tests performed with low environmental load turns out to perform almost the same. For changing setpoints, this is mostly due to the feedforward controller, which is the same for all three LQR implementations. The following scenarios are considered:

Scenario 1 - A station keeping situation

The vessel is required to stay in the same location with the same heading using the best possible accuracy.

Scenario 2 - A docking situation

The vessel is required to perform a docking operation at a harbor with a reduced maneuvering space. The DP algorithm is guided by a higher-level algorithm or human interaction through a joystick.

The path planning is simulated by spontaneous operator change of setpoints, which mimics a docking operation performed with an operator using a joystick. The reference model generates a path with position, speed and acceleration for every point on the track from the current vessel configuration to the current target. The change of operator input, with an adapting reference model is illustrated in Figure 5.4, 5.5, and 5.6.

Environmental conditions

Environmental conditions are chosen to be:

- Zero winds
- Fixed current at 0.15 m/sec.
- First and second order wave load. Significant wave height is 36 cm, and wave periode is 2 seconds.

5.2.2 DP plots

Figure 5.4, 5.5, and 5.6 illustrates the vessels behavior with the LQR, LQR with acuator dynamics modeled, and LQR with both modeled actuator dynamics and integral action. Simulations are performed for 250 seconds, with varying operator input throughout the period. These plots are intended to cover both scenarios described in 5.2.1. Though wave loads are based on a random algorithm, all vessels are simulated simultaneously, such that both environmental load and operator input are equal for all three simulations.

5.2.3 Discussion

The performance of the controllers have shown that including more control states for the LQR actuator to work with results in higher performance. The most notably difference is when advancing from the simple LQR controller to the LQR controller with modeled actuator dynamics. The fact that the controller becomes aware of the actuator dynamic makes it possible to tune the weighting matrices much more aggressively, while still maintaining stability. This effect is apparent upon studying the plots of the three controllers.

The effect of the integral action included in the latter controller is very apparent in the plot, as one can see how the integral term removes the standard deviation. However, upon studying the plot, one will notice that the integral action uses a considerable amount of time to remove the standard deviation, considering the small size of the vessel (CyberShip II). This is because weighting of the integral term in the $\boldsymbol{Q}$ matrix has been done relatively conservative in order not to maintain system. robustness.

These findings does not reveal that the winnings of the augmentations that are shown actually does render the system better, and that implementing the LQR controller with both modeled actuator dynamics and integral action is superior to the rest. This suggests that implementing the latter controller *will* have an effect.

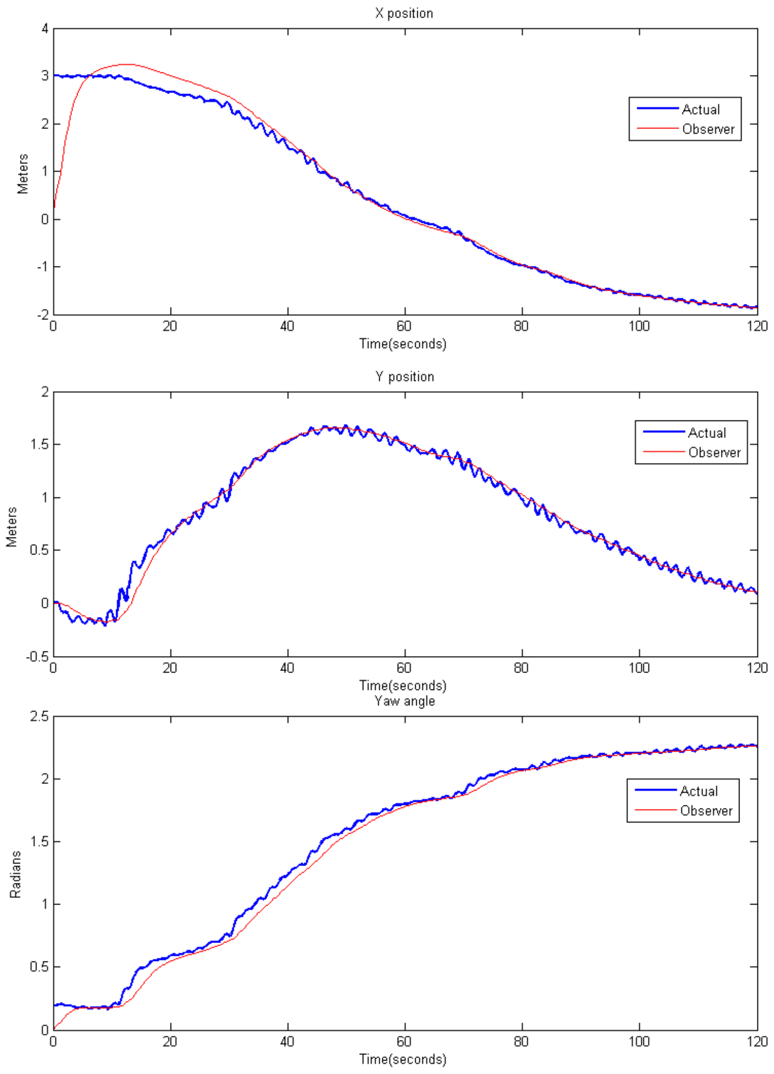


Figure 5.2: The DP observer described in Section 5.1.3.

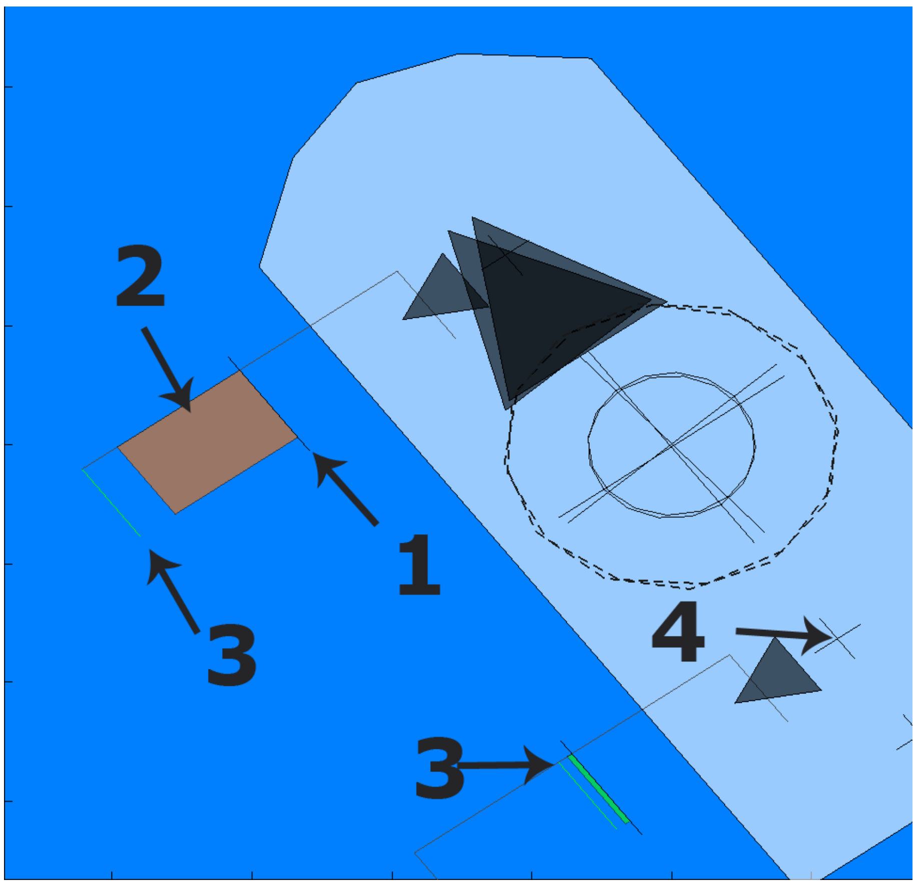


Figure 5.3: A closer look at a vessel during simulation. This particular situations displays a vessel that is just slightly out of its setpoint position. Its bow thruster is currently working at almost full power, applying a force that is pushing the vessel toward its port direction, while the stern thruster is almost at its neutral position.

1: Thruster zero value. If the colored bar is not visible, it indicates that the current momentum generated by this thruster is zero.
2: Thruster force indicator. This force is meant to indicate the current force applied on the vessel in the point where the thruster is located on the vessel. The color of the thruster force bar goes from green to yellow to red while going from zero power to full power.
3: Current thruster setpoint, indicated by a green line. As thrusters have dynamics, the applied force from a thruster is not necessarily what the controller commands. Note how the setpoint indicated in the bow thruster is saturated by the physical thrusters force limit.
4: The actual position of the thruster on the vessel. The arrow in conjunction with the cross illustrates the direction the thruster is mounted.

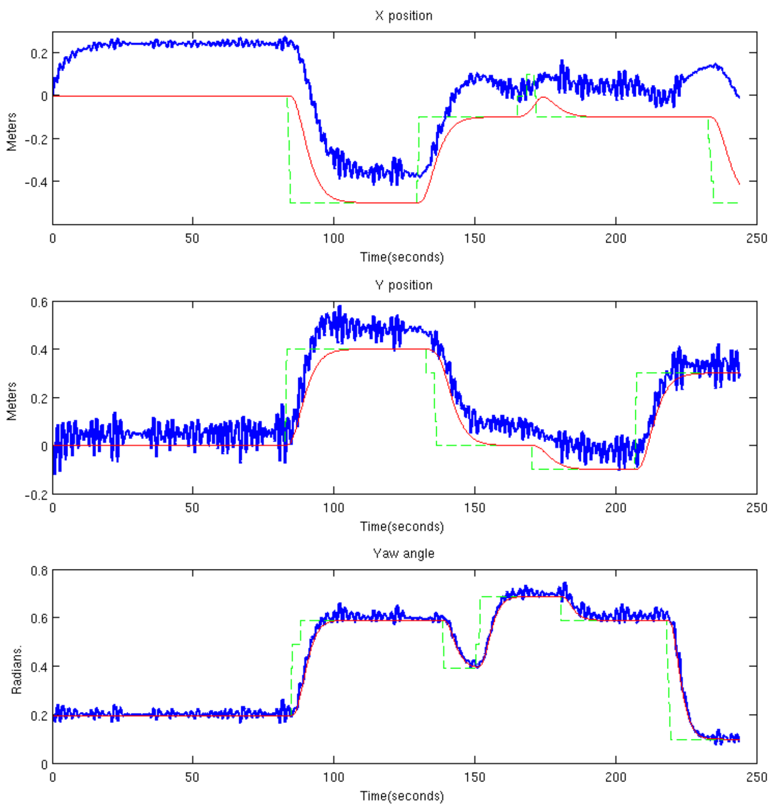


Figure 5.4: Performance of the LQR controller over a period of 250 seconds. Simulated environmental conditions as described in 5.2.1. Note the standard deviation in both position and yaw due to currents and second order wave loads.

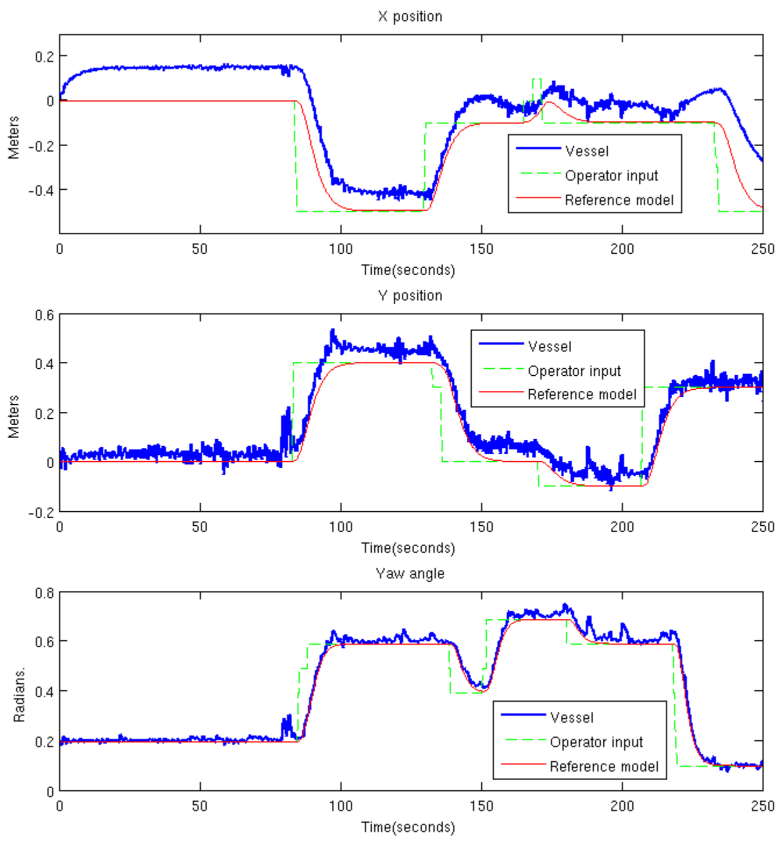


Figure 5.5: Performance of the LQR controller with actuator dynamics modeled. Simulated for 250 secconds. Simulated environmental conditions as described in 5.2.1. Note how this controller is comparable to the LQR controller that also includes integral action.

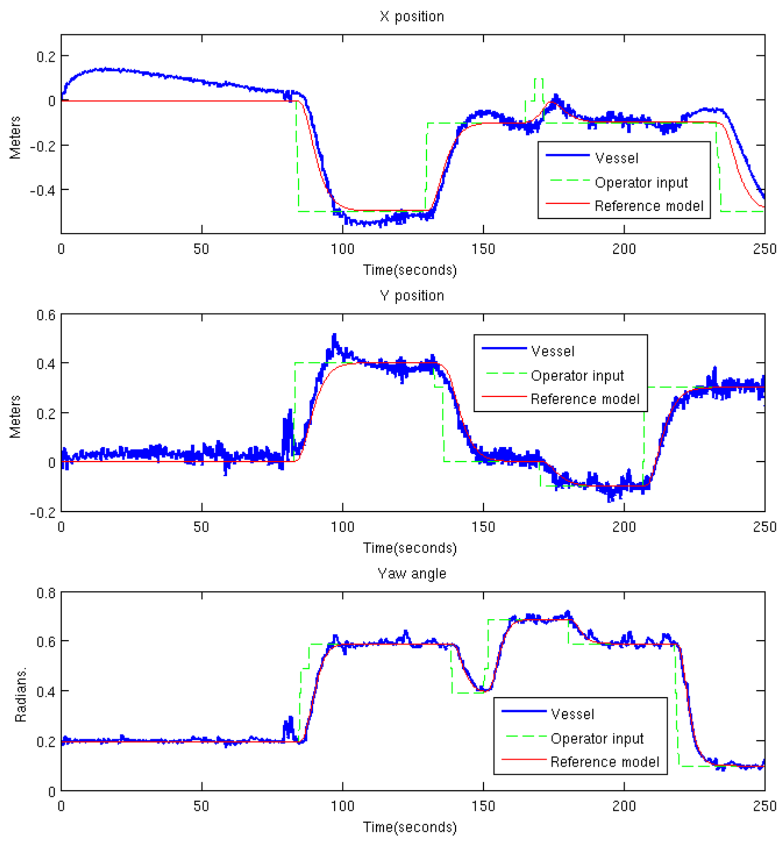


Figure 5.6: Performance of the LQR controller with integral action and actuator dynamics modeled.Simulated for 250 seconds. Simulated environmental conditions as described in 5.2.1. Notice how the integrating term slowly encounters the currents and second order wave load.

Chapter 6

Hardware-in-the-loop simulation

This chapter covers the concept of Hardware-in-the-loop (HIL), and introduces its usage within the DP field. A HIL simulator available at Maritime Robotics AS has been used as preparation of full-scale tests on the Viknes 830 vessel. However, this particular vessel was not ready before the end of this thesis, such that the performed simulations are not confirmed by sea trials.

6.1 Introduction to HIL

A vessel DP implementation is a fairly complicated process. In addition to correct hardware installation, verification of performance and security demands is of high importance. Such a task can be tedious and expensive if performed on an actual vessel.

The concept of HIL-Simulation is based on a reconstruction of the vessel plant at a degree where the control system is facing the same interface as it would otherwise face on board the vessel. This interface can either be created by a software simulation of all hardware components, or that the hardware is included in-the-loop, or anywhere in-between these. This relation is named the *SW-to-HW ratio* [21].

HIL testing is an effective way of proving complex embedded real-time systems. While the most apparent reason is the ability to track down and resolve implementation-related issues prior to vessel installation, there are benefits in terms of the ability to perform failure checks that would otherwise be hard to complete on the actual vessel. A more comprehensive introduction to HIL testing is available in [17], [21], and [9].

6.2 HIL for small marine craft

This section describes the application of the HIL simulator available at Maritime Robotics AS. The intention is the preparation of a DP system for the Viknes 830 vessel. As the necessary hardware (thrusters) is not available at the time of writing, sea trials are not performed.

6.2.1 Modeling

In order to create a simulation that is as realistic as possible, the data from the model vessel 'CyberShip II' has been used in the HIL simulator. This is the same model as is used during simulations when developing the algorithms in the Matlab simulator (see [22]). The CyberShip II data has been chosen superior to the data know for the Viknes vessel, as a proper model identification of the Viknes vessel can not be performed before the vessel is properly installed with transversal thrusters.

In essence, this will require the controllers to be retuned for the Viknes 830 vessel when these model parameters are available and installed in the HIL simulator. However, the integrity of hardware interfaces, etc, is not affected by the vessel model, and will reflect the hardware available at the Viknes 830 vessel. Note that though damping and inertia data are taken from the CyberShip II model, the thruster placement is taken from the Viknes 830 vessel, such that the transfer from CyberShip II to the Viknes vessel will become easier.

The mathematical model implemented in the HIL simulator is based on (3.1). The system is implemented as a 3-DOF model, where first, second, and third order damping matrices are used. This model is similar to the model used in the Matlab simulator (see 5.1).

In order to visualize the simulations performed in the HIL simulator, a interface similar to the interface provided in the Matlab simulator is available (see Figure 6.1).

At the time of writing, environmental modeling is not yet available on the HIL simulator. This should however, be a minor issue, as the main purpose will be to reveal issues related to interface, unmodeled lag in the simulator, etc.

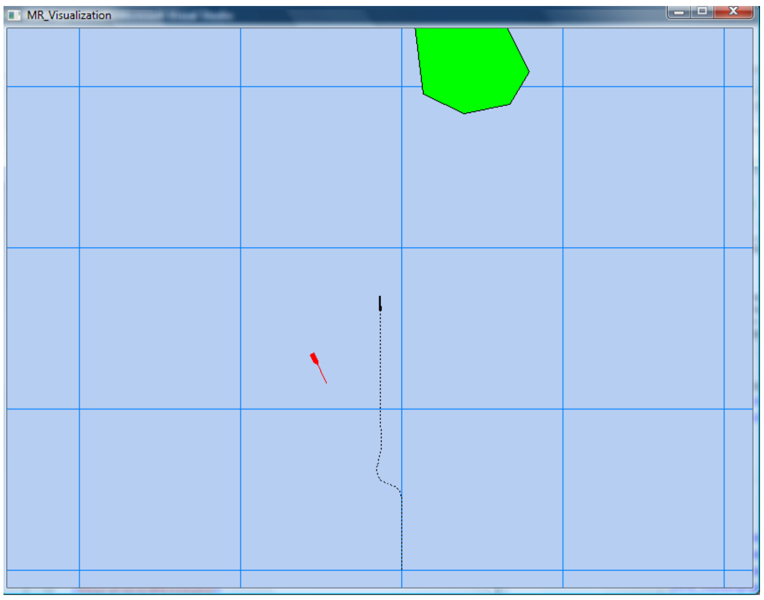


Figure 6.1: The HIL simulator built-in visualizer displaying two vessels operating outside of Munkholmen, Trondheim. Courtesy of Maritime Robotics AS.

6.2.2 Hardware configuration

The HIL simulator utilized is assembled as a rack-mounted setup (see Figure 6.2). The setup is assemble of the following components:

- HIL application computer(running Linux Debian), running the C++ HIL application that simulates the environment present at sea.
- On-Board-Computer (OBC)
  The OBC(Linux Debian) handles all sensors, actuators and communication on board the vessel. Communication with sensors and actuators are performed with applicable DAQ and serial interfaces.
- An AIS-unit coupled with a roof-mounted antenna
  The AIS receives messages from surrounding ships containing their current position, heading, speed and other relevant information. This unit is equal to the one present on board the vessel.
- Interface setup
  In order to provide the exact same interface for the OBC, one has to provide equal analog and serial interfaces in-between the HIL simulator computer and the OBC. The HIL computer is equipped with a DAQ card, that enables the HIL simulator to respond with analog outputs, equal to the response of the actual vessel peripherals. See Section

6.2.3 for more on how a message traverses through the HIL simulator system.

6.2.3 Software configuration

The configuration of the OBC and the HIL computer includes a relatively high number of software components in order to provide the required functionality, while at the same being well presented to the user. A schematic overview of the complete software configuration of the HIL simulator and the OBC is included as Appendix C.

However, there is only a subset of these software components that are directly involved in the DP operation, that is, feeding the filtered vessel state to the controller and moreover feeding the resulting control law to the actuators. In order to perform this cycle such that the HIL simulator emulates the actual vessel responses sufficiently, the setup illustrated in Figure 6.3 has been applied.

As can be seen in Figure 6.3, messages containing data on vessel position and attitude are passed from the GPS model using the NMEA format. NMEA, or more accurately NMEA0183, is a standard used for digital communication in maritime applications. A typical GPS receiver generates NMEA messages with information on vessel position (and sometimes estimated heading and velocity), at a rate at about 1 Hz. These messages reaches the OBC in a similar way that messages from a real-life GPS will reach the OBC. Moreover, the vessel state vector is reconstructed from the NMEA messages, and passed to the DP controller. Note that in a production situation, this DP controller will be present inside this system as a C++ software component. However, DP controllers are currently implemented in Matlab.

Furthermore, the control vector $\boldsymbol{\tau}$ from the DP controller is used to allocate matching thruster setpoints. These setpoints are transferred to the thrusters using a analog signal generated by a DAQ-card present in the OBC. However, when operating using a HIL simulator, the analog signal is converted back to a digital signal and fed into the HIL-simulator, where the thruster forces are converted back to the control vector, and fed to the vessel model.

This cycle introduces a time-delay measured to about 0.3 seconds, which must be considered when evaluating controller stability.

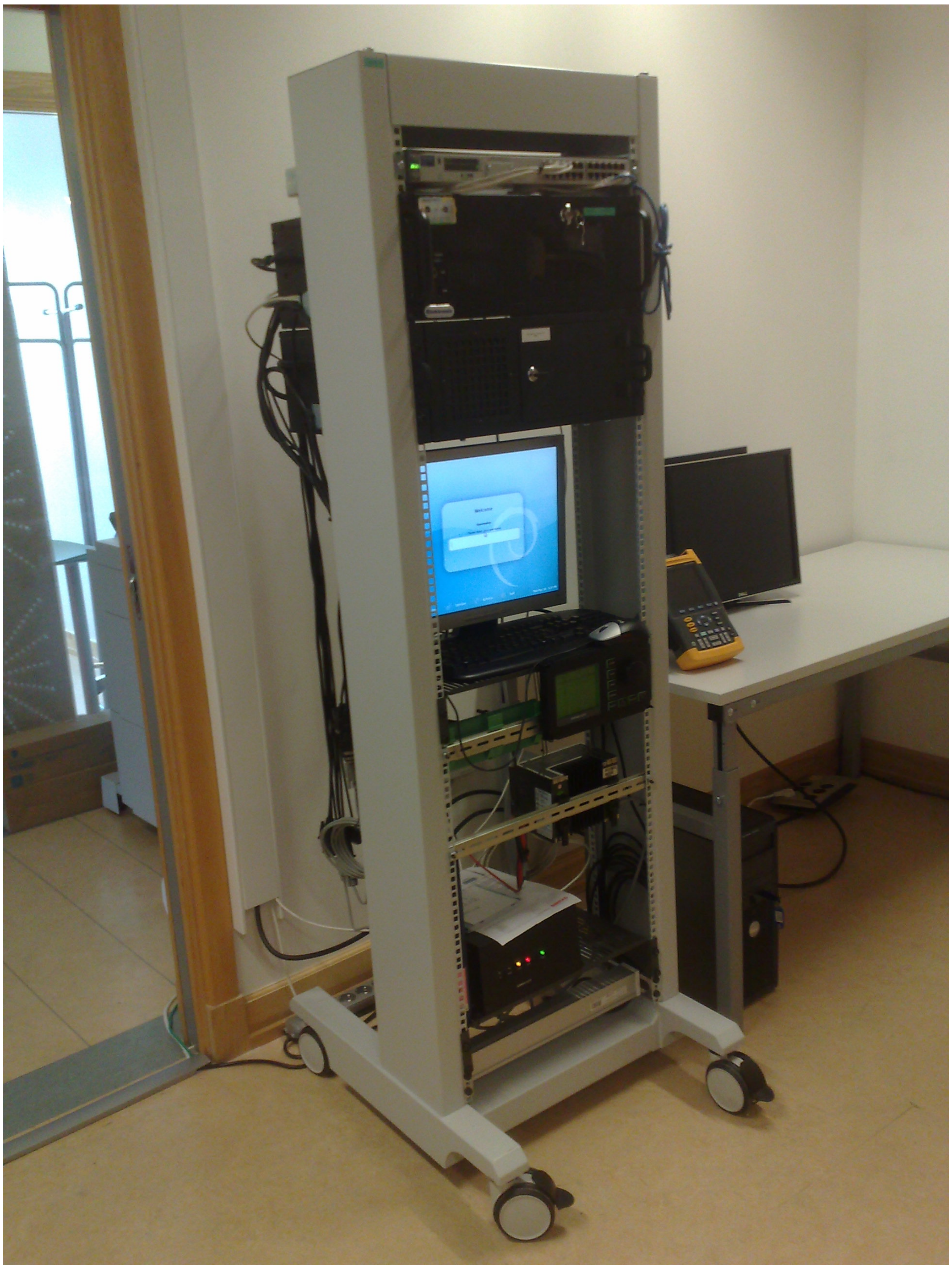


Figure 6.2: The HIL simulator rack used during simulations. Courtesy of Maritime Robotics AS.

Matlab integration

There are many situations where development of algorithms is most effectively done in languages like Matlab (used in this thesis), instead of more general and strict languages like C and C++. The motivation for this is the high availability of built in math-tools. Furthermore, as one of the major strengths of an C++-application is the potential for utilizing the computer in a notably more effective way, however, this is not needed as the algorithms applied in this thesis is relatively computationally cheap.

The software design for the Viknes 830 OBC includes the DP controller as a C++ software component. In order to follow the design specification, this software controller is designed to communicate using the UDP/IP interface with the remote Matlab application. As this setup is network based, any computer operating in the same network as the OBC can be utilized, typically a laptop computer running Matlab on board the vessel. When using the HIL simulator, this is typically a computer on the local network.

A Simulink UDP interface is used to communicate with the OBC through the network. The m-script DP simulator is connected to the Simulink UDP bridge using a custom-made Simulink s-function[1]. This setup is illustrated in the Simulink model illustration in Appendix B.

In effect, the combination of the software design described in Section 6.2.3 and the UDP interface described above results in a connection inbetween the DP controller and the thrusters on the vessel (or the simulated thruster components running on the HIL simulator). This relatively complex travel of information has been found (through measurements done in Matlab) to induce a lag of about 0.3 seconds. It is a notable lag, but should not be of a major importance when compared to the much slower dynamics present in both the vessel and the thrusters themselves Also, when running tests on the HIL simulator, this lag is believed to induce a less significant error than the error induced through erroneous model dynamics.

6.3 Simulation results

By applying the Matlab simulator in conjunction with the Simulink UDP interface(see Appendix B), a connection between the HIL simulator and the Matlab simulator was successfully established.

[1]An s-function is a custom made Simulink function that can be programmed using a variety of programming languages.

The vessel model applied in the HIL simulator is the Viknes 830 parameters[2]. No environmental disturbances are simulated, as this is currently not implemented in the HIL simulator. The controller used in the Matlab simulates the LQR controller with modeled actuator dynamics, described in Section 4.6.

A 500 seconds simulation was performed, where the reference position was changed several times during this period. Figure 6.4 is the plot created by the Matlab simulator. Additionally, the HIL simulator features visualization of the path traveled by the vessel. A screenshot of the visualizer is provided in Figure 6.5. Notice how these two plots can be compared.

During simulation, no problems were discovered. As expected, the HIL simulator introduces a lag (recorded to about 300 ms), however, this lag is much faster than the vessel dynamics, such that the lag impact can be neglected.

[2]Parameters for the Viknes 830 vessel are not fully identified, however, the missing parameters are empirically estimated using visual inspection of the model compared with the real vessel.

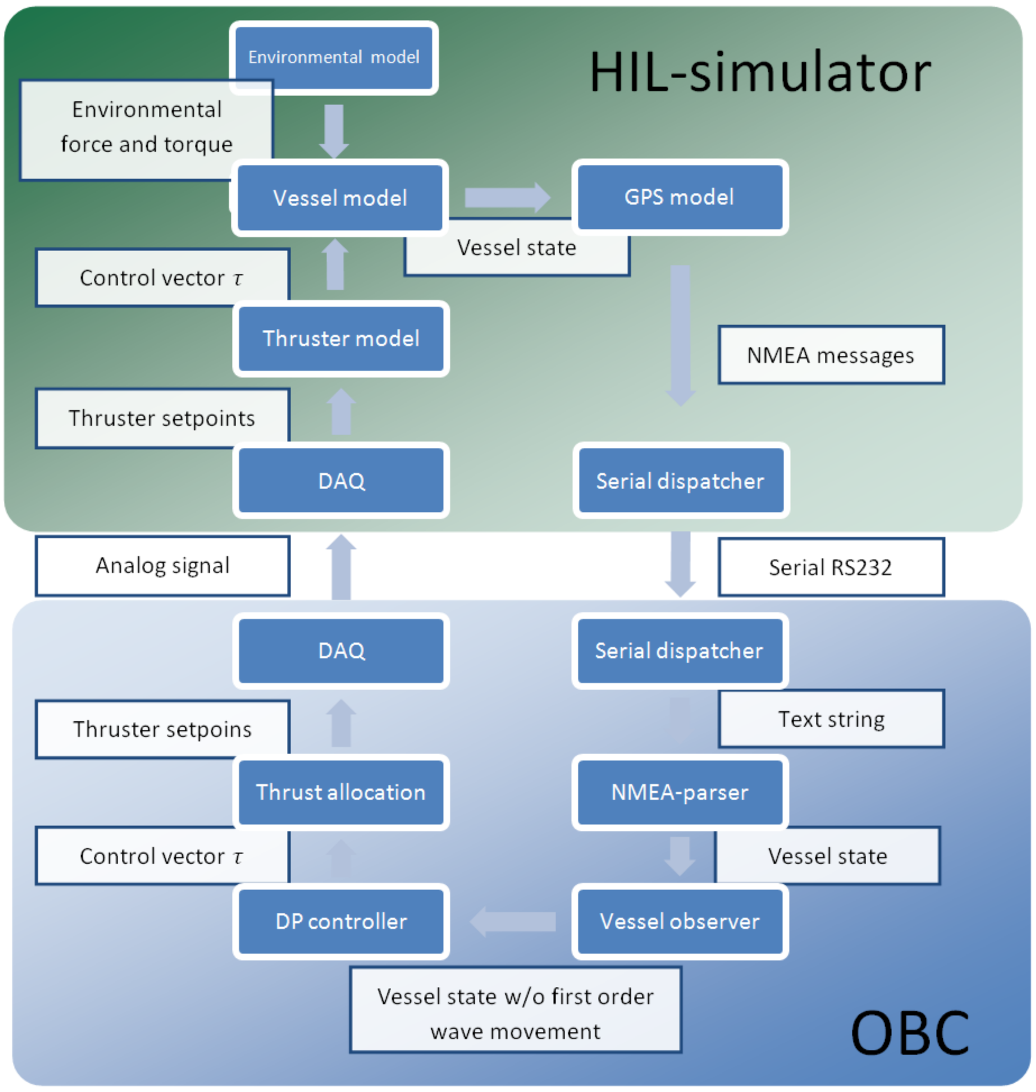


Figure 6.3: Software and hardware components involved when running DP on the HIL simulator. The square boxes indicate what is being transferred in-between the surrounding components. The corresponding hardware is illustrated in Figure 6.2.

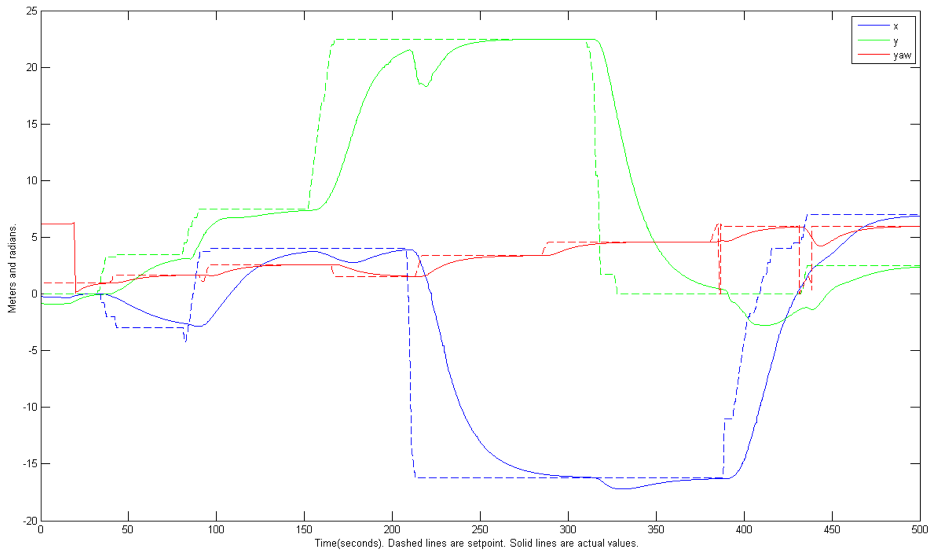


Figure 6.4: A DP operation performed on the HIL simulator at Maritime Robotics AS. Note that the seemingly steep steps in yaw appear because the yaw angle is always kept within $[0, 2\pi]$.

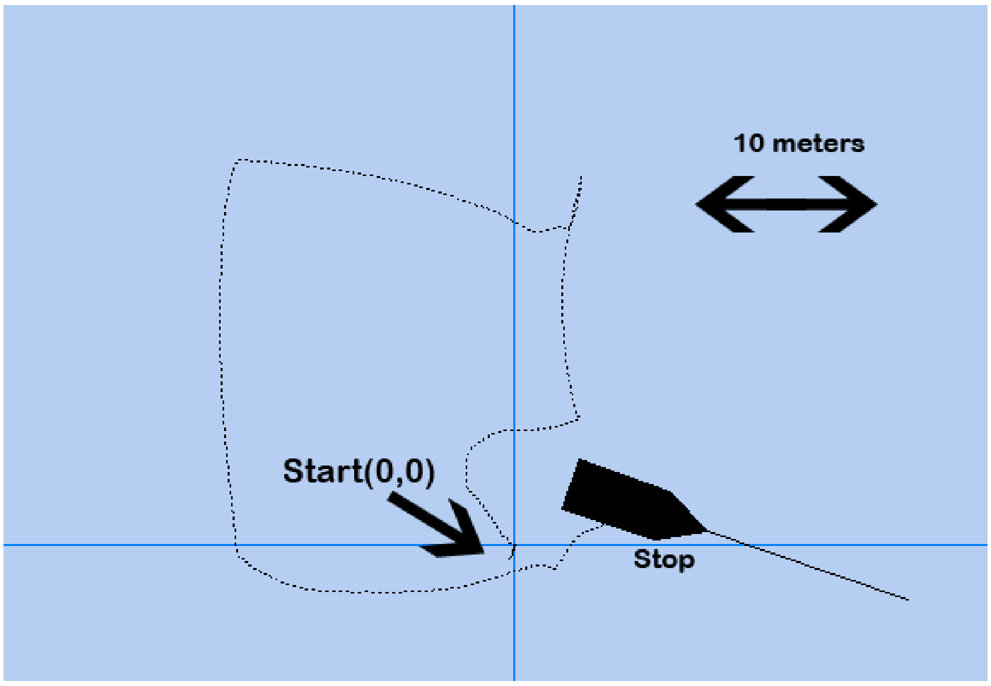


Figure 6.5: This is the output from the HIL simulator visualizer that was captured simulataneously with the plot in Figure 6.4. Arrows and text are edited in after capture.

Chapter 7

The Viknes 830 vessel

A controller has been developed with a vessel of type ”Viknes 830” from ”Viknes båt og Service AS” (see Figure 7.4) in mind. This is due to it being one of the vessels available at ”Maritime Robotics AS”. The ”Viknes 830” is over 8 meters in length, and is thus big enough to house all necessary equipment for computer controlled operation , while still providing the necessary facilities to work on board.

This vessel makes a good development platform, due to the availability of an on-board computer with the necessary interfaces for on-board peripherals such as GPS, INS, and thruster actuators installed. At the time of writing, the hardware state of the vessel is that computer-control of rudder and throttle is installed and are tested to be working. However, the scheduled installation of proportional bow and stern-thrusters is delayed, and will not be completed until the after finalization of this thesis. However, control development is done using a HIL-simulator which mimics the hardware interface available on board the Viknes vessel.

This section presents the technical properties of the vessel, the modifications done in order to make it computer controllable, as well as a section on model identification.

7.1 Vessel setup

The Viknes 830 is produces by 'Viknes Båt og service AS'. Key technical specifications are given in Table 7.3.

See [6] for the complete technical specification, as stated by the factory.

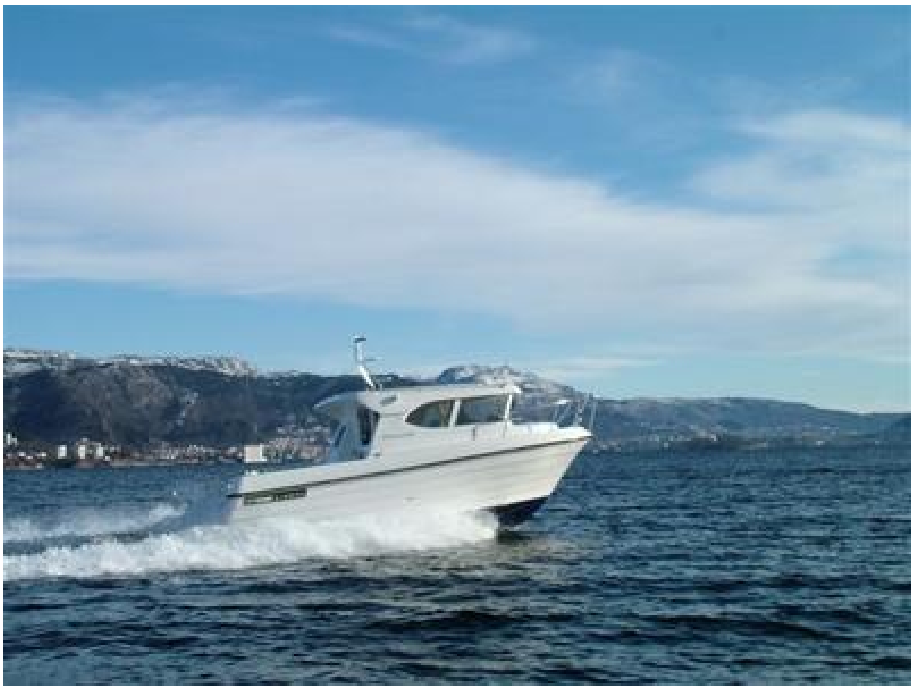


Figure 7.1: Viknes 830. Courtesy of Viknes Båt og Service AS [7].

Table 7.1: Viknes 830 Standard specifications

Length	8.6 m
Width	2.97 m
Weight fully loaded	3.300 kg
Engine:	Yanmar 144 hp
Bow thruster:	On-off Sleipner Motor electrical thruster

7.1.1 Modifications done by Maritime Robotics AS

The current vessel setup is designed to function as a USV testing platform. This includes, e.g., an on board computer, radar, a GPS-system, and communication solutions.

7.2 Hardware and software setup

Picture of the hardware and software layout installed on the vessel, that is used for DP operation.

7.3 Thrusters

The thruster configuration of a vessel is crucial for DP performance. In the case of smaller craft, there is currently a limited choice of setups compared to the range of thruster configurations available for larger vessels. Smaller craft are typically equipped with either fixed angle thrusters in addition to the main propulsion unit, or a dual pump-jet configuration (see 2.2).

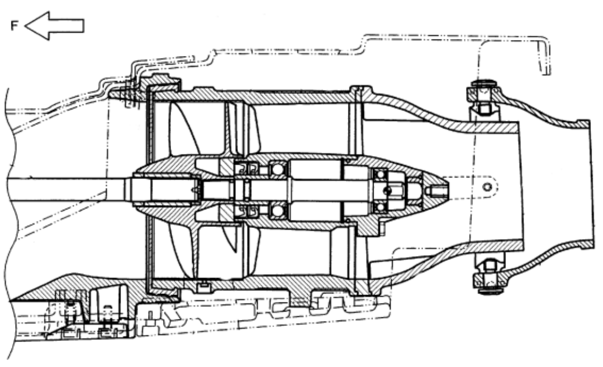


Figure 7.2: A pump-jet engine. Courtesy of US Patent #6682676.

The LQR dp controller produces a control vector $\boldsymbol{\tau}$, that is distributed to the available thrusters. However, this approach requires the vessel to be fully actuated in order to be able to produce all possible $\boldsymbol{\tau}$, that can be commanded by the controller. A dual pump-jet setup is fully actuated, and can be used in conjunction with the LQR controller. However, this includes the use of azimuth thrusters, which requires a more complex thruster allocation algorithm.

The more traditionally setup includes a main propulsion engine, a rudder, and zero or or more transversal[1] thrusters installed. In order for such vessels to be fully actuated, at least two transversal thrusters are required. In the case of one or zero thrusters, the vessel will be underactuated, and the LQR controller is not suitable. There are however, many articles available on the field of underactuated vessel control (see Section 2.3).

In the case of 1 thruster installed, this will typically be a bow thruster[2]. This vessel is also underactuated, but the thruster provides an extra degree

[1]A transversal thruster is a thruster capable of creating a force in the vessel sway direction.

[2]A bow thruster is typically a thruster mounted in a tunnel as far down and forward in the vessel as possible. The bow thruster provides force in the vessel sway direction.

of control, such that maneuvering challenges is greatly reduced. In the case of 2 thrusters installed, this will typically be a bow thruster and a stern thruster[3]. In this case the vessel will be fully actuated in surge, sway and yaw.

Figure 7.3: The electric thruster SP95, provided by Sleipner Motor AS. Courtesy of [4].

7.3.1 Thruster considerations

When installing thrusters on larger vessels, one will have to consider several factors. These include the number of thrusters, which can be anything from zero to ten, or more, but also whether they should be fixed or azimuth thrusters.

Designing a thruster setup for smaller vessels reduces the number of options significantly. The main design decisions are:

- Whether to use both a bow and stern-thruster, or a bow-thruster only.
- Whether to use a proportional or ON-OFF thruster setup.
- Whether to use an electric or hydraulic driven thruster setup, and choosing the thruster strength.

On-off vs. proportional thrusters

The traditional thruster installed on recreational smaller vessels is an on-off[4] electrical thruster. From a DP point of view, these might be unsuitable due to that one will have to turn on and off the thruster to often in order

[3]A stern thruster is a thruster mounted in the rear of the vessel, providing force in the sway direction.

[4]An on-off thruster is a thruster that either can deliver full throttle in both direction, or no throttle.

to create a reliable DP system. Such behavior would result in burning the thruster relays. The alternative is a proportional thruster, that will give the controller the ability to choose output power. This feature will make the controller able to keep the thruster at a constant speed instead of turning the thruster rapidly on and off. In addition to less wear and tear, this will result in severely reduced energy usage. The suggested hardware upgrade for the Viknes 830 vessel includes proportional controllable thrusters.

Hydraulic vs. electric thrusters

Given a hydraulic thruster setup, there will be mounted a hydraulic oil pump on the vessel main engine, such that energy can be transferred mechanically from the engine to the thrusters. The most significant strength of hydraulic thrusters is their relatively large power / size - ratio. Also, there is no need for a powerful generator. This makes hydraulic thrusters the only suitable choice for medium sized vessels. However, today's commercially available electric thrusters can power up to a 30 meter vessels [4]. As electrical thrusters tend to be less expensive than hydraulic thrusters, the choice of whether to install a hydraulic or electric thruster system on a small craft DP is thus dependent on the need for continuous DP operation. This will normally would require hydraulic thrusters, due to the extensive power-drain by the electric alternative.

Main engine

The main propulsion unit on small craft is usually a diesel or petrol-engine. These engines have other dynamics than what is the case for electrical or hydraulic thrusters. Most importantly is that they are connected with a gear, that requires special treatment.For example, one does not want to change gears too often, such that, e.g., a DP algorithm must make sure that the vessel does not overshoot when approaching the reference position.

Furthermore, a combustion-engine can never run below a certain RPM (typically 800 for small craft). As this makes applying small amounts of force difficult, the problem is usually solved by using a troller-gear.

See Section 7.4.3 for more on identification of the main engine.

7.4 Model identification

A proper model identification of the vessel is necessary in order to design sufficiently accurate models. This section describes how the surge dynamics of the Viknes 830 vessel is identified, and furthermore suggests how sway

and yaw damping models can be identified. Note, as the hardware required to identify the Viknes 830 in sway and yaw is missing, model properties for the CyberShip II has been applied as a substitute throughout the thesis. Whenever the complete set of model parameters for the Viknes vessel is available, it should replace the CyberShip II parameters.
Model parameters are applied in the following sections:

- The Matlab DP Simulator (see Section 5).
- The HIL simulator vessel model (see Section 6).
- The LQR DP controller algorithms.

Vessel parameters that need to be identified for DP operation are:

- Vessel mass and moment of inertia:
  These values are used to design the inertia-matrix for the vessel model.
- First and second order damping terms in surge, sway and yaw:
  The first and second order damping terms are used in the Matlab DP Simulator and the HIL simulator, while the LQR-controller requires a linear system, and uses only the first order damping-term. However, at low speeds, a linear damping model of the vessel is sufficiently accurate (see Section 4).
- Thruster characteristics:
  Thruster time-constants and saturation limits need to be determined in order to apply correct setpoints for the vessel actuators.

The inertia matrix can be determined by the specifications provided by the manufacturer. The mass of the vessel is given, as well as the mass of heavier units such as engine, gear, water tanks, etc. This also allows for easy calculation of vessels moment of inertia, which was calculated in the following way:

7.4.1 Surge damping

The surge damping profile of the Viknes 830 was archived by traveling at a set speed, and then switching to neutral gear in order to record the step response. The resulting speed profile was logged, and later plotted side by side with data from the Matlab simulator in order to determine the first and second order damping matrix (see Figure 7.4).

Ideally, a single run going at maximum speed should reveal the whole response for the vessel. However, because the wave resulting from traveling at higher speeds can affect the vessel when reaching lower speeds, runs staring at lower speeds are also performed. Furthermore, performing the tests

Table 7.2: Calculation of the Viknes 830 moment of inertia about the z-axis

Unit	Weight (kg)	Average distance from CG (m)	Moment of inertia (kgm^2)
Hull	2780	2	11120
Engine / gear	420	2	1680
Equipment	70	3,2	716,8
Diesel tanks	300	3,5	3675
Water tanks	120	2,6	811,2
Septik	20	2	80
Targa	90	4	1440
2 persons:	180	1	180
Total mass:	3980	Total moment of inertia:	19703

at lower speeds allowed for testing in a confined area, such that these results were not as affected by waves and current. The varying plots are plotted on top of each other in order to make the modeled version fit the mean of the recorded values. The model properties resulted in the following values for damping in the surge direction:

Table 7.3: Model paramters identified for Viknes 830 in surge direction

First-order damping:	50
Second-order damping:	135

7.4.2 Sway and yaw damping

Sway and yaw damping identification is not performed due to lack of transversal thruster for the Viknes 830. However, the identification of these parameters can be completed similar to the identification performed in surge direction, thus forcing a maximum torque on the vessel by running the bow and stern thruster at max force in opposite directions until the vessel reaches a terminal yaw speed. The thruster can then be switched to zero force as quickly as allowed by the hardware, and the resulting step response in speed should be recorded by the use of a gyro compass. In sway, the method that was applied in surge direction can be used.

Furthermore, coupled damping dynamics is likely to have a notable influence on the vessel movement. This can be identified by a proper measurement on, e.g., change in position resulting from a yaw torque. See **??** for more on model identification.

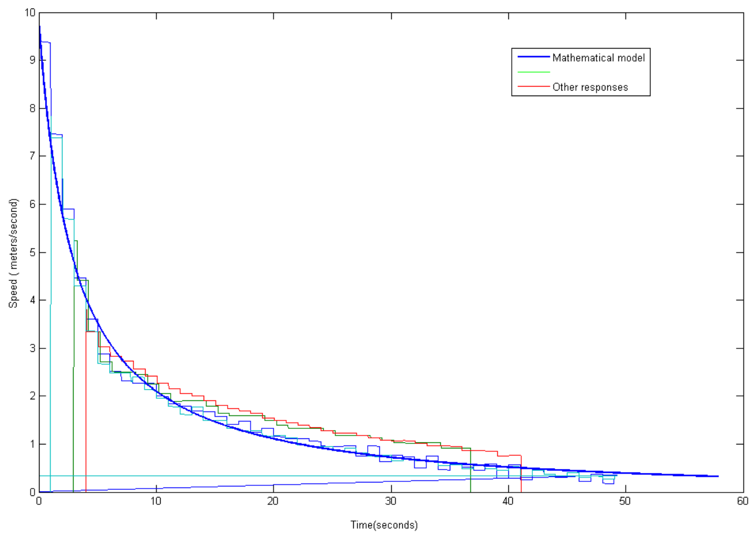


Figure 7.4: Surge response at engine cut-off. Viknes 830. Model identification performed April 22nd, Trondheim harbor. Low winds, and about 0.3 meters significant wave height. All recordings are presented in the same plot in order to illustrate that the identified second order model match the mean of the measurements.

7.4.3 Main engine

The DP controller control vector $\boldsymbol{\tau}$ is given in Newton, which implies that this will need to be translated to RPM, which is the control input for the main engine. The power/RPM diagram provided in Figure 7.5 is provided by the manufacturer. However, this table alone is not enough for a proper identification, as the force is a product of speed and power, a speed profile is recorded. The stationary vessel speed was recorded at all RPMs from 800 to 3300 in steps of 250 RPM. Note that this is for forward movement. Backwards movement remains to be identified. Data is presented in table 7.4, and plotted in Figure **??**.

Note that for low speed applications such as DP, one has to properly identify the dynamics where the troller-gear is in use, which in effect is when the engine is commanded to run at a lower speed than the minimum RPM.

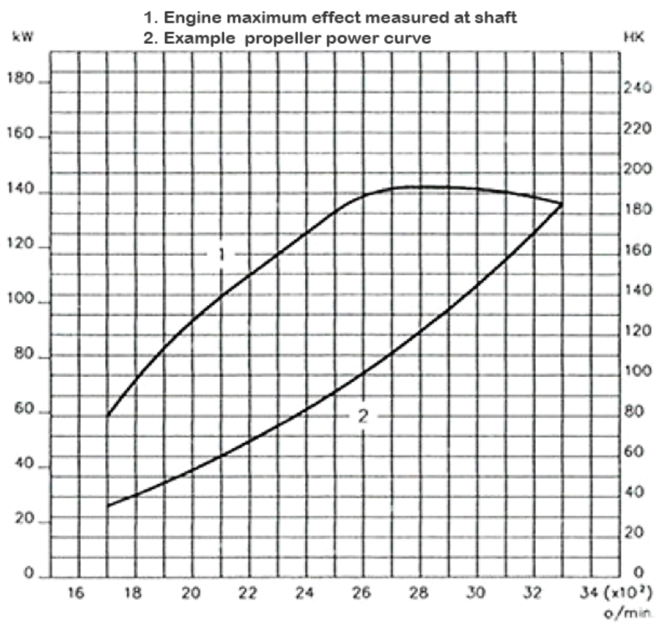


Figure 7.5: Power/RPM plot for the Yanmar 4LHA-DTP engine installed in the Viknes 830 vessel. Courtesy of Yanmar Norge AS.

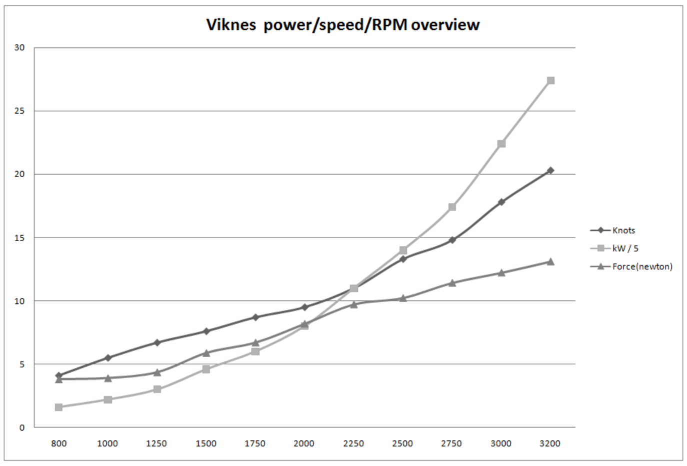


Figure 7.6: Speed measurements and data from Figure 7.5 have been joined to form this plot. Notice how the Newton/RPM curve can be approximated quite good with a linear fitting. Figure is generated from numbers in Table 7.4.

Table 7.4: Force/RPM profile recorded for the 4LHA-DTP engine installed in the Viknes 830 vessel

RPM:	Speed(m/s)	Power(kW)	Force(kN)	Speed (m/s)
800	4.1	8	3.8	2.11
1000	5.5	11	3.9	2.83
1250	6.7	15	4.4	3.45
1500	7.6	23	5.9	3.91
1750	8.7	30	6.7	4.48
2000	9.5	40	8.2	4.89
2250	11	55	9.7	5.66
2500	13.3	70	10.2	6.84
2750	14.8	87	11.4	7.61
3000	17.8	112	12.2	9.16
3200	20.3	137	13.1	10.44

Chapter 8

Conclusion

The LQR controllers have been implemented and tested on the both the Matlab simulator and the HIL simulator. It has become clear that the LQR controller with modeled actuator dynamics and integral action is superior to the simpler alternatives discussed. Note that all LQR controllers used this includes the use of a reference model and feedforward control. Thus, the LQR with modeled actuator dynamics and integral action, aided by a reference model and a feedforward control is the proposed controller for the Viknes 830 vessel.

Experience reveals that tuning the controller model with faster actuator dynamics than what is the actual case might result in better controller stability, due to the saturation limits of the thrusters. Furthermore, experiments reveals that the accuracy of vessel model implemented in the controller is relatively low, as most model errors can be compensated for by tuning the weighting matrices.

A Matlab simulator has been developed in order to make development of DP algorithms faster. This simulator has been augmented with a UDP connection, such that communication with the on board-system on the Viknes 830 vessel is possible. This has been verified by performing a successful connection in-between the Matlab simulator and the HIL simulator at Maritime Robotics. Furthermore, a successful DP operation simulation has been performed using this connection. In effect, this means that the Viknes 830 vessel can be tested for DP operation by simply connection the Matlab Simulator to the already installed computer in the Viknes vessel.

In order to further simplify the development of DP functionality for the Viknes vessel, the Matlab simulator has been constructed such that it can be of assistance when preforming model identification. This simplification can be done because the Matlab simulator is able to visualize the actual vessel on top of the modeled vessel in real-time while performing the tests. This will allow the operator to rapidly try new parameters in order to find a good match.

The application of a HIL simulator when developing the controller has turned out to be very useful, even though an installation on an actual vessel is not yet performed. The winnings of using the HIL-simulator is that one can easily find and correct problems that are related to interfaces, etc. prior to installation on the vessel. Other hardware related problems such as delays will also surface when performing tests on the HIL simulator.

8.1 Further work

This thesis has treated the LQR controller approach. There are many alternatives to the LQR approach, e.g, the nonlinear sliding mode and the backstepping approaches. A natural continuation on this project would to compare the performance of the LQR controller with these controllers, and determine what can be the winnings of the different controllers over the LQR controller.

Furthermore, as this thesis leads up to the implementation of a DP system on the Viknes 830 vessel, a natural continuation of this work is to carry out the implementation whenever the required hardware becomes available. The installation of transversal thrusters on the Viknes 830 will also allow for a proper model identification in the sway and yaw degrees of freedom, which will make the HIL simulator able to simulate the Viknes vessel more accurate. According to the results of this thesis, an implementation of the LQR controller with modeled actuator dynamics and integral action should be a good alternative.

Appendix A

DVD contents

The enclosed DVD contains the Matlab® DP Scenario Simulator, with complete source code, a digital copy of this report and a copy of the referred articles. The report is located in the root folder *report*, Matlab code is located in the root folder *matlab_code*, while referred articles are located in *referred_articles* folder. Additionally, there is a *readme.txt* containing this text.

Appendix B

Simulink® model

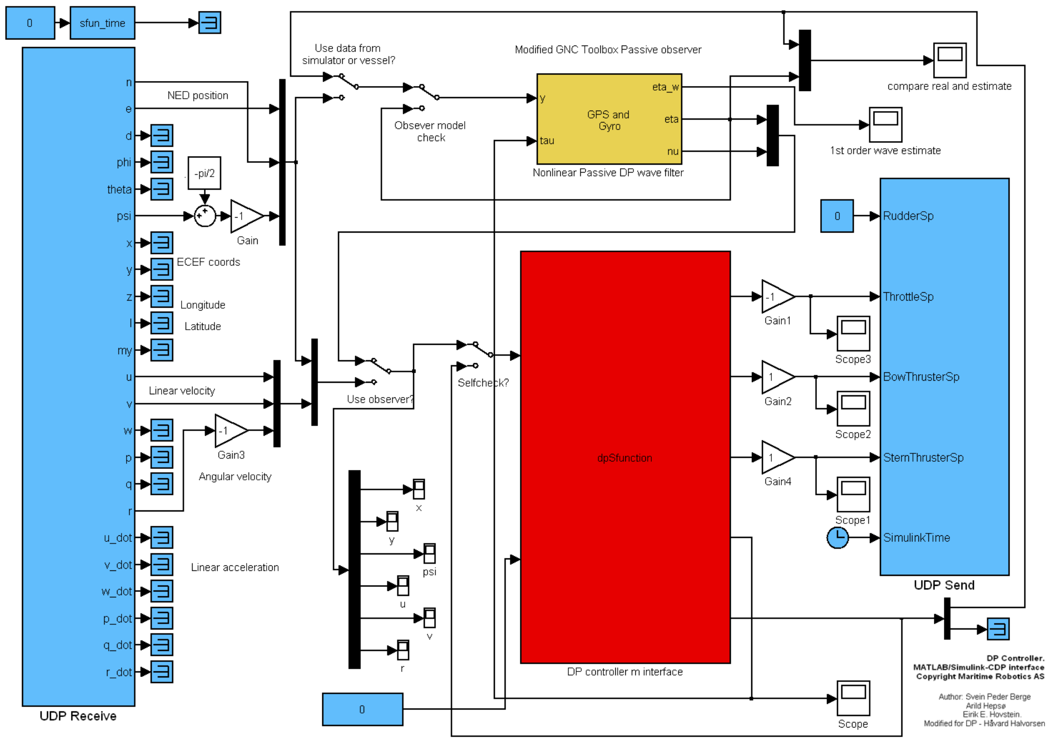


Figure B.1: The Simulink® model utilized to combine the m-file based DP controller with the Simulink-based observer and UDP interface. Note the observer switch. In upper position, this mode makes the observer output fed back to the input of the observer, effectively removing all observer terms but the vessel model. This mode is used to check the integrity of the observer vessel model (see Section 4.1). Also note the conversion that is performed from the coordinate system used when received from the UDP connection with the HIL-simulator.

Appendix C

HIL simulator and OBC software overview

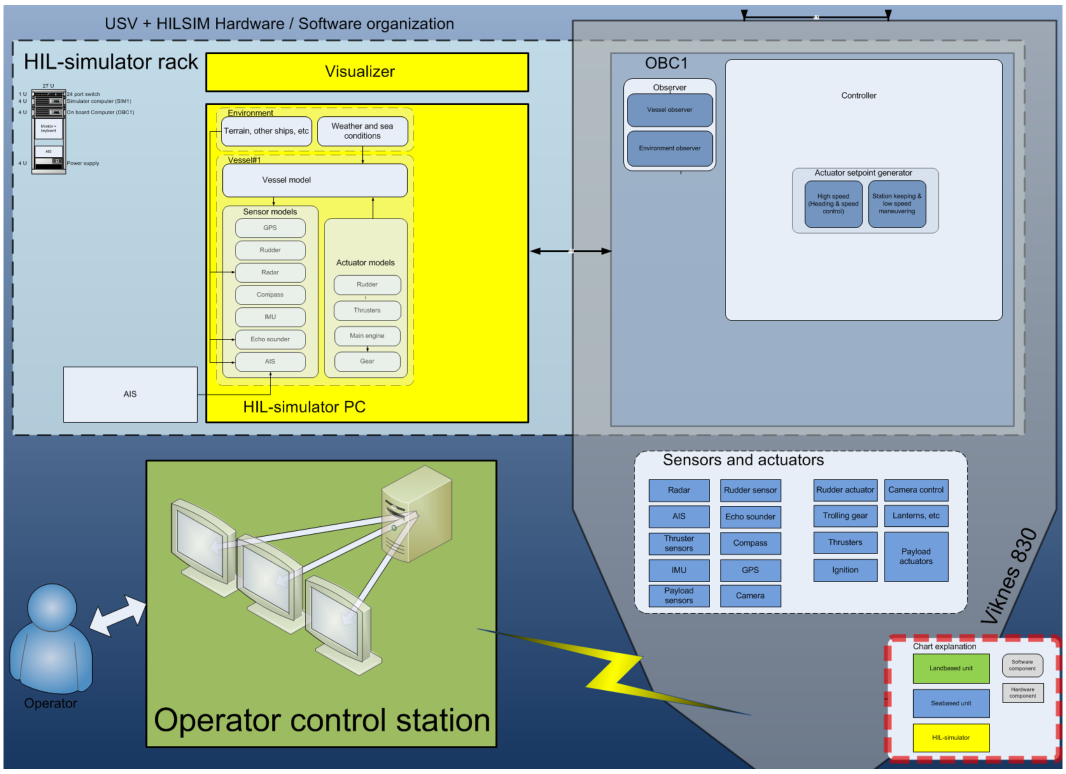


Figure C.1: A brief illustration on the hardware/software components that are used in the HIL simulator-setup available at Maritime Robotics AS. This illustration is meant to illustrate both HIL simulator and regular operation. Note that this illustration is intended to give the viewer an overall overview, and has been simplified from the original version.

Appendix D

Matlab simulator user guide

This appendix includes a step by step tutorial on how the Matlab DP simulator developed throughout the project and thesis period is used. This includes creating a new DP scenario, defining vessels and environmental disturbances, and finally simulating the scenario through the built-in graphical presentation.

In addition to the example given below, there are more examples included on the DVD. These are:

- **Integrator test**. This example demonstrates two equal vessels, situated with the same initial condition in the same location. However, one of the vessels uses the Euler integrator, while the other uses the ODE45 integrator. This example shows the superiority of the ODE45 integrator. However, the initial condition given to these vessels are very unrealistic. Given normal conditions, the euler integrator will perform fine.
- **Formation test**. This example demonstrates the ability to simulate several ships simultaneously. See Figure D.1.

The creation of a new scenario has to be done by creating an m-file present in the simulator base directory. I recommend looking on one of the provided example scripts. These are prefixed with 'EXAMPLE_'. Executing these files should start the simulator with the varying scenarios. This section describes how to setup a scenario in its simplest form. This scenario includes a single vessel given a start state, a reference state, and configured to use a PID control-algorithm as described in Section **??**. This is implemented in the file 'EXAMPLE_simple.m', located in the base simulator directory on the enclosed DVD. Note: Remember to install the GNC Toolbox, provided at [1](Complete instruction is available in the included 'readme.txt')

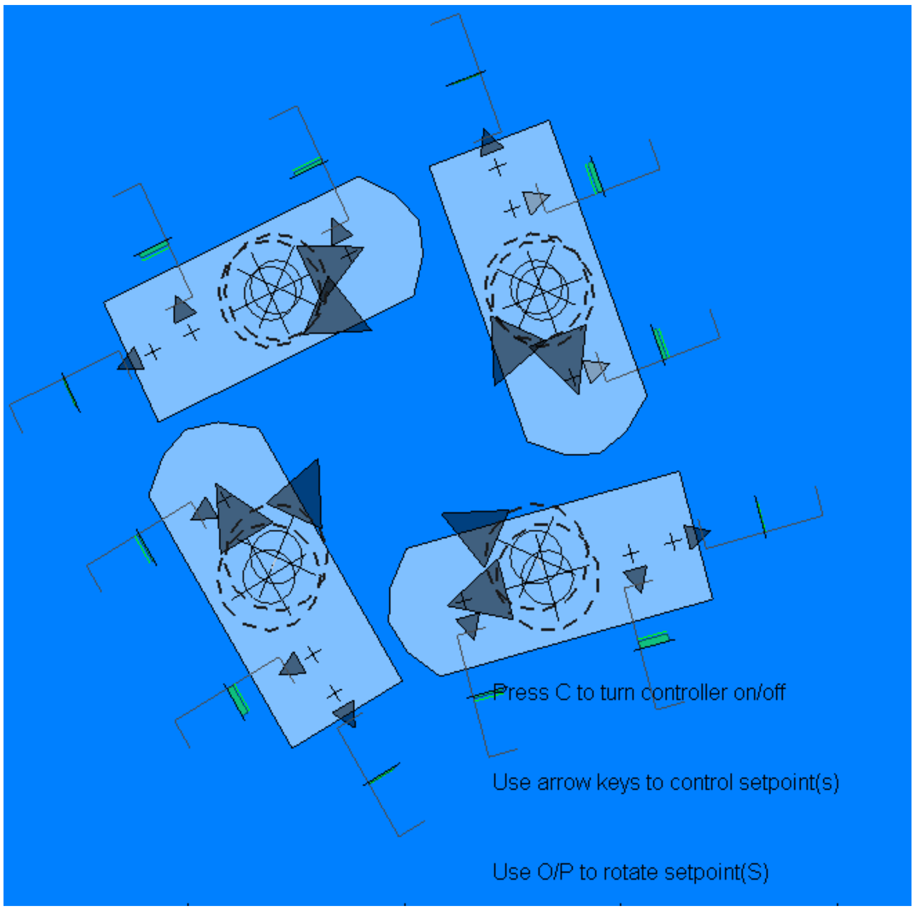

Press C to turn controller on/off
Use arrow keys to control setpoint(s)
Use O/P to rotate setpoint(S)

Figure D.1: The DP simulator as shown from running the 'EXAMPLE_formationTest.m' script.

1. Create and open a new file in the base directory of the simulator. For example 'myScenario.m'

2. As the simulator is object oriented, everything is treated in terms of objects. The first one will need to do is creating a Simulator singleton[1] object. The scenario object is also made global in order to make all objects in the scenario access information and handles stored within it.

```
%Step 1. Create a Scenario object
global scenario; scenario = Scenario();
```

This scenario object will contain all other vessels, winds, currents, etc.

[1]A singleton is an object which only should be instantiated once.

3. Let us now add a vessel object to the scenario. As you can see from the example, three parameters are given, namely the initial state, the reference state and a vessel name. There are more available parameters that can be added, such as controller, solver, etc. Note that you can add an arbitrary number of vessels to the scenario. The 'state' and 'refState' parameters are specified according to the statevector $\boldsymbol{x}$.

```
%Step 2. New vessel object
scenario = addVessel(scenario,...
    'state', [10 10 0   0 0 pi/4   0 0 0   0 0 0]',...
    'refState', [14 10 0   0 0 2.3  0 0 0   0 0 0]',...
    'name', 'TestVessel');
```

4. Let us now add wind and current objects to the scenario. Note that you could in theory add more than one wind object, or more than one current object. However, this would not make sense, since wind and current only can have one velocity and direction at a given time. You can however adjust the behavior of the environmental disturbances by passing more arguments.

```
%Step 3. Add wind and current
scenario = addWind(scenario, ...
    'direction', pi/2, ...
    'velocity', 50);

scenario = addCurrent(scenario, ...
    'direction', 0, ...
    'velocity', 3, ...
    'directionDynamics', 'sinus', ...
    'directionRotSpeed', 2);
```

   In these settings, velocity is always given in meters per second, and direction is always given in radians.

5. The only thing remaining is to start the scenario simulation.

```
%Step 4. Start
run(scenario);
```

You should now see a running scenario, similar to Figure D.2. A plot figure similar to Figure D.3 will also appear. Notice how the setpoint values changes as time passes by studying the plot figure. Setpoint changes can be triggered by keyboard input while the scenario windows is active. Instructions are show inside the figure. Note: When adding more than one vessel to the scenario, a separate plot figure will be added for each vessel.

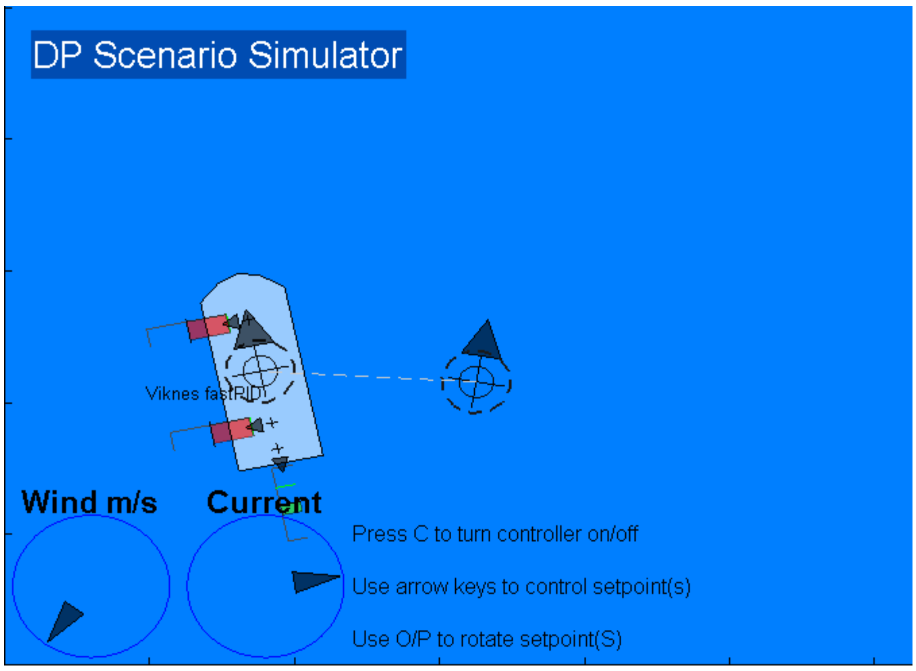

DP Scenario Simulator
Viknes fastRIDT
Wind m/s
Current
Press C to turn controller on/off
Use arrow keys to control setpoint(s)
Use O/P to rotate setpoint(S)

Figure D.2: The DP simulator as shown from running the 'EXAMPLE_simple.m' script.

```
function [] = run(voidScenario)
%RUN Start scenario simulation

   %Timestep. If necessary, the Scenario class can be augmented
   %to include timestep as a property, rather than a hardcoded value.
   timestep = .01;
   endtime = inf;

   %Make sure we have access to the global scenario singleton.
   global scenario;

   %Get scenario vessels
   vessels = get(scenario,'vessels');

   %Get scenario winds
   winds = get(scenario,'winds');
   currents = get(scenario,'currents');

   %Main loop
   for time = 0:timestep:endtime

       %Update reference state for the vessels whose reference position
       %has been changed by user keypress.
       newVessels = get(scenario,'vessels');
```

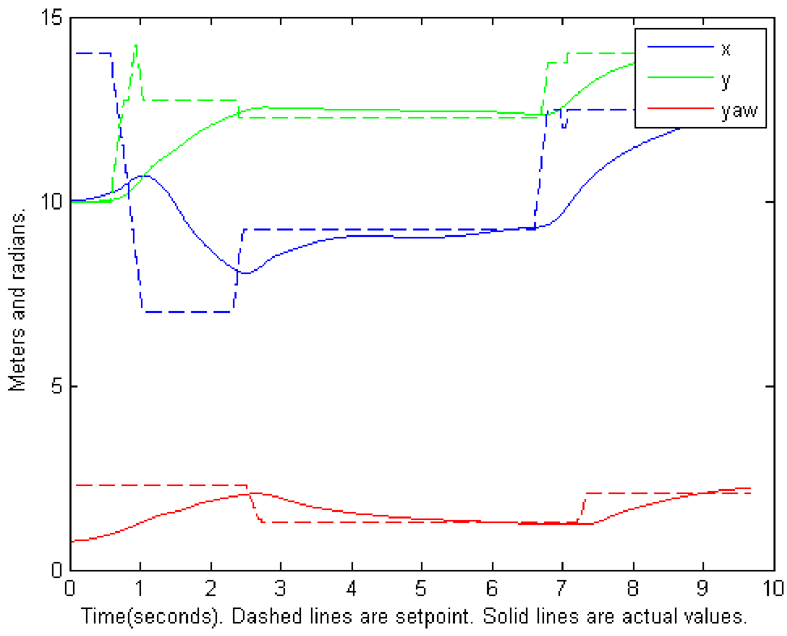


Figure D.3: Plot of x,y and yaw for the simulated ship in 'EXAMPLE_simple.m' script. Note the relatively quick vessel response. This is due to that the vessel is configured with very powerful thrusters.

```
        for i=1:length(vessels)
            vessels(i) = set(vessels(i), 'refState', ...
                get(newVessels(i), 'refState'));
            vessels(i) = set(vessels(i),'controllerEnabled', ...
                get(newVessels(i),'controllerEnabled'));
        end

        %Update scenario disturbances
        for i=1:length(winds)
            winds(i) = calcNewState(winds(i),timestep,time);
        end
        for i=1:length(currents)
            currents(i) = calcNewState(currents(i),timestep,time);
        end

        %Loop hrough all vessels
        for i=1:length(vessels)

            %Reset forces and disturbances
            vessels(i) = set(vessels(i),'input',zeros(6,1));
            vessels(i) = set(vessels(i),'disturbance',zeros(6,1));

            %Apply all scenario winds to the vessel
            for j=1:size(winds,1)
                vessels(i) = applyWind(vessels(i),winds(j));
            end

            %Apply current(s)
            for j=1:size(currents,1)
                vessels(i) = applyCurrent(vessels(i),currents(j));
            end

            %Apply controller
            vessels(i) = applyController(vessels(i),time);

            %Map calculated controller momentum from controller to thrusters.
            vessels(i) = mapThrusters(vessels(i));

            %Convert thruster forces to global momentum. This has to be done
            %back and forth because the setpoint calculated in mapThrusters
            %isnt necessarily the current force in the thruster due to thrust
            %dynamics. Also, the thrusters can be saturated.
            vessels(i) = mapThrusterForces(vessels(i));

            %Update vessel states using the chosen ship model
            vessels(i) = calcNewState(vessels(i),timestep,time);

            %Upddate transformation ation matrices after state matrices, such
            %that graphics can be draw properly.
            updateTransforms(vessels(i));

            %Make sure all vessels are drawn properly.
            draw(vessels(i));

        end

        %Make sure all graphical objects are properly updated
        drawnow();

        %Free processor time to other processes
        pause(timestep);

    end
end
```

Bibliography

[1] Marine Cybernetics - GNC Toolbox. `http://www.marinecontrol.org/download.html`. Accessed 20.12.2007.

[2] Mercury Marine webpage. `http://northamerica.mercurymarine.com/otherproducts/smartcraft/smartcraftinaction/zeus.php`. Accessed 18.12.2007.

[3] The Mercury Marine Zeus system - An unparalleled level of performance, safety & vessel control. `http://www.cmdmarine.com/PDFs/zeus-brochure.pdf`. Accessed 19.12.2007.

[4] Sleipner Motor AS webpage. `http://www.sleipner.no`. Accessed 18.12.2007.

[5] The Friede Goldman website. `http://www.fng.com`. Accessed 17.12.2007.

[6] Viknes 830 specifications. `http://www.viknes.no/index.php?side=bat&id=136&vis=tekniskedata`. Accessed 15.12.2007.

[7] Viknes Båt og Service AS - Viknes 830 photos. `http://www.viknes.no/index.php?side=bat&id=136&bilde=1891`. Accessed 15.12.2007.

[8] Volvo Penta IPS webpage. `http://www.volvo.com/volvopenta/norway/no-no/marineengines/volvo_penta_ips/howitworks`. Accessed 15.12.2007.

[9] What is Hardware-in-the-Loop Simulation?. `http://www.adi.com/products_sim_qhil.htm`. Accessed 29.04.2008.

[10] *Bï¿½tmagasinet*, 9:78–79, 2007.

[11] D. Bray. *Dynamic Positioning*, volume 9. Oilfields Publications Inc, 2003.

[12] O. Doucy and F. Ghozlan. Advanced Functions for USV. *ATMA*, 2008.

[13] H. Fay. *Dynamic Positioning Systems - Principles, Design and Applications*. Editions Technip, 1990.

[14] T. I. Fossen. *Marine Control Systems: Guidance, Navigation, and Control of Ships, Rigs and Underwater Vehicles.* Marine Cybernetics AS, Trondheim, 1st. edition, 2002.

[15] T. I. Fossen and J. P. Strand. Nonlinear passive weather optimal positioning control(WOPC) system for ships and rigs: experimental results. *Automatica*, 37, 2001.

[16] O. G. Hvamb. A New Concept for Fuel Tight DP Control. *Proceedings of the 1998 Dynamic Positioning Conference, Houston, Texas, USA.*, 2001.

[17] T. A. Johansen, A. J. Sørensen, O. J. Nordahl, O. Mo, and T. I. Fossen. Experiences from Hardware-in-the-loop (HIL) Testing of Dynamic Positioning and Power Management Systems. *OSV Singapore, 24-25 September, Singapore*, 2007.

[18] T. Kim and T. Basar. Asymptotic Stabilization of an Underactuated Surface Vessel via Logic-Based Control. *Proceedings of the American Control Conference, Anchorage, AK, USA*, 2002.

[19] K. Y. Pettersen and T. I. Fossen. Underactuated dynamic positioning of a ship - experimental results. *IEEE Transactions on Control Systems Technology*, 8(5), 2000.

[20] D. F. Phillips. The dynamic positioning of ships; the problem solved? *Proceedings of the UKACC '96. 2-5 September.*, 1996.

[21] R. Skjetne and O. Egeland. Hardware-In-the-Loop Simulation for Testing of DP Vessels. *IEEE Electric Ship Technologies Symposium (ESTS), Philadelphia*, 2005.

[22] R. Skjetne, N. Smogeli, and T. I. Fossen. A nonlinear ship maneuvering model: Identification and adaptive control with experiments for a model ship. *MIC-25(1):3-27, 2004*, 2005.

[23] A. J. Sørensen. *Marine Cybernetics: Modeling and Control. Lecture Notes, Fifth Edition.* Norwegian University of Science and Technology, 2005.

[24] B. Volker. Unmanned Surface Vehicles - A Survey. *France*, 2008.