NTNU
Norwegian University of
Science and Technology

Optimal path following for underactuated marine vessels

Håvard Nordahl

Master of Science in Engineering Cybernetics
Submission date: June 2008
Supervisor: Alexey Pavlov, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem Description

Path following for marine vehicles (ships, AUVs, ROVs) is a challenging practical problem especially in the underactuated case, i.e. when the vehicle has less actuators than degrees of freedom. Several solutions to this problem have been recently obtained. The controllers proposed in these solutions guarantee that the vehicle does follow the desired path (which can be, for example, the straight line corresponding to a desired course). At the same time little has been done regarding finding an optimal controller for path following or optimizing existing controllers to obtain better performance. MPC is well suited for optimal control design, and has previously been applied to some marine applications. The main goal of this project is to investigate the possibility of employing model predictive control methods for the problem of path following.

Assignment given: 07. January 2008
Supervisor: Alexey Pavlov, ITK

Optimal path following for marine vehicles
Master's Thesis

Håvard Nordahl

Norwegian University of Science and Technology NTNU
NO-7491,Trondheim, NORWAY

June 2, 2008

MASTEROPPGAVE

Kandidatens navn:	**Håvard Nordahl**
Fag:	**Teknisk kybernetikk**
Oppgavens tittel (norsk):	**Optimal baneføLging for marine fartøyer**
Oppgavens tittel (engelsk):	**Optimal path following for marine vehicles**

Oppgavens tekst:

Path following for marine vehicles (ships, AUVs, ROVs) is a challenging practical problem especially in the underactuated case, i.e. when the vehicle has less actuators than degrees of freedom. Several solutions to this problem have been recently obtained. The controllers proposed in these solutions guarantee that the vehicle does follow the desired path (which can be, for example, the straight line corresponding to a desired course). At the same time little has been done regarding finding an optimal controller for path following or optimizing existing controllers to obtain better performance. MPC is well suited for optimal control design, and has previously been applied to some marine applications. The main goal of this project is to investigate the possibility of employing model predictive control methods for the problem of path following.

The assignment consists of several steps.

- Study the literature on path following for marine vehicles, and on nonlinear MPC.
- Formulate practically reasonable optimality criteria for path following.
- Design controllers based on a combination of MPC and Line-of-sight guidance for path following with the proposed optimality criteria.
- Investigate the performance of these MPC designs relative to the existing LOS design by means of numerical simulations.
- Summarize the results in a report.

Oppgaven gitt: 07.01.2008
Besvarelsen leveres: 02.06.2008

Utført ved Institutt for teknisk kybernetikk
Veileder: Assistant Professor Alexey Pavlov
Medveileder: PhD-student Even Børhaug
Medveileder: PostDoc. Erik Kyrkjebø

Preface

This report presents the results from the work done in connection with my master's thesis. A lot of time and work has been put into achieving good results. The report is therefore quite long, but don't worry there are many figures. I would like to thank my supervisors Alexey Pavlov, Even Børhaug and Erik Kyrkjebø for their advice and help, and Giancarlo Marafioti for taking time to give me helpful information and tips. I would also like to thank my class mates Øiving, Roger, Robert, Kristoffer and Aleksander for making this year great.

Håvard Nordahl

Abstract

This report presents two optimization problems, where three cost-functions are suggested for each. The goal for the first optimization problem is to find a time variant look-ahead distance which improves the performance of the vessel in terms of the cross-track error, relative to constant look-ahead distances. The second optimization problem is an extension of the first, where a time variant surge velocity reference is also calculated. This results in smaller cross-track errors, at the cost of increased calculation time. It is assumed that the path and the desired surge velocity on the path is supplied by some external source.

Existing κ-exponential control laws are used to track the references resulting from the calculated optimal look-ahead distance and surge velocity reference. The predictions needed to solve the optimization problem are made from a model where the control laws are inserted into the dynamics to simplify the model.

The optimization problems are solved for two different approaches. The first approach uses an LTV model for predictions and a QP-solver to solve the optimization problem. The second approach forms predictions of the states by numerical integration of the system dynamics and uses an optimization problem solver for general non-linear functions to solve the optimization problem. The latter approach generally results in longer calculation times but better accuracy, while the first approach yields convexity of the optimization problem. A passive observer is used to estimate the current such that it can be included in the predictions.

Four of the six suggested cost-functions leads to significantly improved performance of the vessel in terms of the cross-track error. This is true both in the case of no disturbances and in the presence of a constant irrotatinonal current. That is, the position of the vessel converges faster to the path, the vessel stays closer to the path and has less over-shoot in the cross-track error, for the optimal inputs than for constant inputs. The cost-functions includes weights for tuning where the tuning process is easy for some of the cost-functions.

One of the cost-functions where only the optimal look-ahead distance is calculated, results in calculation times shorter than the time between samples, after a few time steps. This is a promising result since the application of MPC to fast-dynamic systems such as marine vessels is desired, but often problematic due to too long calculation times for solving the optimization problem. It takes more time to solve the optimization problem when the optimal surge velocity reference is found in addition to the optimal look-ahead distance.

Contents

List of Figures

1 Introduction

Model predictive procedures used to find optimal control inputs, or to generate optimal reference trajectories, are known as model predictive control (MPC) and model predictive guidance (MPG), respectively. The advantage in using such procedures is that the controller can predict the systems response to the control inputs, or reference trajectories, enabling the controller to plan the series of future control inputs (reference trajectories). This can be exploited in the path following problem by finding a series of inputs, based on the predicted response of the cross-track error, which allows fast reduction of large cross-track errors, and reducing reduction rate of the cross-track error as the cross-track error becomes small. This will cause fast convergence of the cross-track error with little or no over-shoot.

The general maneuvering problem, as stated by *Ivar-Andre F. Ihle, Roger Skjetne and Thor I. Fossen* [1], divides the control problem into a geometric and a dynamic task, where the geometric task is to reach the desired path and then stay on it, while the dynamic task is to satisfy a time, speed or acceleration assignment along the path. The desired path for a marine vehicle can be expressed as way-points, where the position reference trajectory is the line connecting two successive way-points. In this report a way point is a fixed point in space, given in Cartesian coordinates in an inertial reference frame. As stated by *Even Børhaug and Kristin Y. Pettersen* [3] this choice of reference trajectory leads to a desirable property. That is, the desired path is composed of linear sub-paths which are independent of time. This again leads to decoupling between the geometric task of controlling position and heading, and the dynamic speed control. Hence, the desired path can be decided without considerations of the speed profile, and the speed profile can be chosen without changing the desired path. This is in contrast to the case where the desired path is given as a time-dependent trajectory where position, heading and speed control is coupled.

The LOS algorithm has successfully been used in guidance laws for many different control applications, ranging from missile guidance to path following for marine vehicles. LOS-based control laws which globally κ-exponentially stabilizes an underactuated marine surface vessel to paths in two dimentions, and an underactuated autonomuos vehicle to paths in three dimensions, are presented by *E. Fredriksen and K.Y. Pettersen* [2] and *Even Børhaug and Kristin Y. Pettersen* [3], respectively. The LOS algorithm achieves stability of the geometric position to a reference trajectory given that a condition on the look-ahead distance Δ is met (see *K.Y. Pettersen and E. Lefeber* [6] for the case of diagonal inertia and damping matrices, and [2] for non-diagonal inertia and damping matrices), and that the heading tracks its desired value. In this report, as done in [2] and [3], the property of the way-point description of the desired path is exploited by choosing a controller that tracks the surge velocity reference, and a controller that tracks the desired heading returned from the LOS-algorithm. Though such controllers have already been obtained, there has been done little regarding optimizing them.

There are some publications on the subject optimal path following for marine vehicles. In the paper by *Even Børhaug, Kristin Y. Pettersen and Alexey Pavlov* [4] an optimal guidance scheme for cross-track control of underactuated underwater vehicles is proposed. This scheme is a model predictive guidance law for cross-track control (MPG). The proposed scheme calculates reference trajectories to be tracked by the heading and pitch controllers, based on some optimization criteria. The results presented in the paper [4] indicates that the proposed control-scheme yields faster convergence then what is achievable if the reference trajectories for the heading and pitch controllers are generated by the LOS-algorithm where the look-ahead distance is constant. An application of MPC to AUVs is presented by *Giancarlo Marafioti* [14], where both a linear time invariant model and a liner time variant model are used for prediction. Both constant and state dependent weights are used in [14], where the state dependent weights shows improvement on the control performance. The MPC application in [14] is used for a control goal of keeping a constant distance from the ocean bottom, assuming constant surge velocity. An MPC scheme designed for an autonomous underwater vehicle is presented in *W. Naeem* [5], where a simple line of sight guidance scheme is utilized to generate the reference headings, which is tracked by a model predictive controller.

The work presented in this report endeavors to optimize path following by finding an optimal time

variant look-ahead distance, used to generate the heading reference for the κ-exponentially stabilizing LOS-based control laws for underactuated surface vessels presented in [2]. The goal is to find cost-functions with constant weights such that the resulting optimal look-ahead distances leads to as fast as possible convergence of the position of the vessel to the path, with as small as possible over-shoot in the cross-track error. The gain in also optimizing the surge velocity reference will be investigated. Cost-functions are mathematical expressions with minima for desirable behavior if constructed properly. The cost-function value is found from predictions of the future behavior of the vessel. The optimal time variant look-ahead distance and surge velocity reference will be found by minimizing the cost-function once for each sample time.

1.1 Lay out

The next chapter presents some definitions used in this report to ease the discussion and the assumptions made in this report. Chapter 3 presents the theroy of the LOS algorithm, rotation of the reference frame and way-point description of paths. A scheme for switching to the next way-point and for choosing proper rotation angles is also presented. The models needed for simulations and predictions are derived in Chapter 4. The prediction models are simplified by insertion of existing control laws. Chapter 5 presents the observer used to estimate the current and shows that the accuracy of the predictions is good after an initial transient. The MPC theory needed is presented in Chapter 6. The optimization algorithm, solvers and the constraints on the optimization problems are also presented in this chapter. The optimization problems are defined in Chapter 7 by analyzing the states for desirable behavior and undesirable behavior. The simulation results are presented in Chapter 8. Two main scenarios are simulated, both in the presence of disturbance and under perfect conditions. Each simulation is discussed. The effects of changing given weights are investigated and some tuning rules are presented. The conclusions are summarized, and suggested further work is presented in Chapter 9.

2 Preliminaries

The following definitions are used in this thesis:

Small look-ahead distance : *A look-ahead distance which causes fast reduction of the cross-track error, but large over-shoot if it was used in a LOS-algorithm where the look-ahead distance is constant.*

Large look-ahead distance : *A look-ahead distance which causes slow reduction of the cross-track error, and small or no over-shoot if it was used in a LOS-algorithm where the look-ahead distance is constant.*

Over-shoot in the cross-track error: *understood as the cross-track error crossing zero before converging.*

H_p : *The prediction horizon is the number of estimated future samples.*

H_u : *The input horizon is the number of calculated future inputs.*

u_d : *The desired surge velocity returned by the optimization problem in the case of time variant desired surge velocity, otherwise constant.*

u_{dp} : *The desired surge velocity which the vessel should track when the vessel is on the path. When the desired surge velocity is constant, we have that* $u_{dp} = u_d$.

$\overline{u}_d$: *The difference between the desired surge velocity returned by the optimization problem and the desired velocity on the path.*

$U1$: *The initial guess of the optimal solution where the inputs* $g_{i,H_u} = 0, i = 1, 2, 3$, *while inputs* $g_{j,k}, j = 1, 2, 3, k = 1, 2...H_u - 1$ *are taken to be the previously found optimal inputs shifted once in time.*

$U2$: *The initial guess of the optimal solution where the input* $g_{1,H_u} = 0.51$ *and inputs* $g_{i,H_u}, i = 2, 3 = 0$, *while inputs* $g_{j,k}$, $j = 1, 2, 3, k = 1, 2...H_u - 1$ *are taken to be the previously found optimal inputs shifted once in time.*

Note *Initial guess U1 implies that the starting point violates the constraints due to* $u_{1,H_u} = 0$, *that is* $\Delta_{H_u} = 0$. *However, this is handled by the optimizer and will be explained in a later section.*

The following assumptions are made

- The force in surge may be both negative and positive. This will be used in some implementations to reduce the surge velocity in order to reduce or avoid over-shoot in the cross-track error.
- The control-laws implemented are based on the vessel model, hence perfect knowledge of the vessel model is assumed.
- Predictions are made based on the same model as the control-laws, inserting the control-laws in the system equations. Hence, the predictions are made assuming no constraints on the control forces.
- The magnitudes of the calculated control inputs are constrained, however the rate is not constrained. Hence it is assumed that the control inputs (τ) are commanded forces supplied to some low-level controller which tracks the commanded forces perfectly.
- It is assumed that the optimal look-ahead distance is the look-ahead distance which results in the smallest possible cross-track error with the smallest possible over-shoot, disregarding the use of control inputs.
- Only constant disturbances modeled as a constant irrotational current affects the vessel.

3 Line of sight and Way points

Some definitions are needed in order to understand the concept of line-of-sight guidance (LOS). The terms that needs to be defined are the cross-track error, the line of sight, the look ahead distance, way-points and the circle of acceptance. First the three lines in the LOS algorithm are defined

Definitions 1, 2 and 3:

1: The cross-track error **2**: The line of sight **3**: The look-ahead distance

The three lines 1, 2 and 3 together make a right triangle, see Figure 3.

1) *The cross-track error is the shortest distance from the body-fixed origin to the path. That is, the cathetus in the triangle made up by 1, 2 and 3, which starts in the body-fixed origin ends in a point on the path.*

2) *The line of sight is a line from the body fixed origin to either the next way point, or to a point on the path determined by the look-ahead distance. That is, the hypotenuse in the triangle made up by 1, 2 and 3.*

3) *The look-ahead distance is the distance along the path which determines which point on the path the line of sight points to. That is, the cathetus in the triangle made up by 1, 2 and 3, which is part of the path.*

The definitions of way-points and the circle of acceptance are based on those given by Thor I. Fossen [11].

Definitions 4 and 5

4: Way-points, **5**: The circle of acceptance

1) *A way-point is a point in space which a vehicle is to pass through. Each way-point is described in Cartesian coordinates (x_k, y_k) for k=1,...,n A set of way points defines a path which a vehicle is to follow. This path consists of the straight lines connecting two successive way-points.*

2) *The circle of acceptance is used in a switching mechanism for selecting the next way-point. The circle of acceptance is a circle with its center in a way-point and with radius R_0. When the vehicle is within the circle of acceptance, it is considered to be close enough to the current way-point to start tracking the next. This can be expressed as*

$$[x_k - x(t)]^2 + [y_k - y(t)]^2 \leq R_0^2$$

3.1 LOS algorithm

The Line-Of-Sight-algorithm (LOS-algorithm) is a well known algorithm which generates a reference trajectory for the heading in order to track a desired path given by way-points. There exists control laws which guarantee stability to the path for heading references generated by the LOS algorithm. An intuitive approach, used in many applications, is to define the line of sight as in Thor I. Fossen [11] where the line of sight is taken to be a vector from the body-fixed origin to the next way-point (x_k, y_k). This corresponds to defining the set-point for the course autopilot (the desired heading) as [11]:

$$\psi_d(t) = atan2(y_k - y(t), x_k - x(t)) \tag{3.1.1}$$

where (x,y) is the vessel position measurement usually measured with a satellite navigation system. The four quadrant inverse tangent function atan2(y,x) is used to ensure that:

$$-\pi \leq atan2(y, x) \leq \pi \tag{3.1.2}$$

The drawback with a LOS vector pointing to the next way-point is that a way-point located far away from the vessel will result in large cross-track errors if there are transverse wind, current and wave disturbances.

Another approach, also defined in [11], is to choose the line of sight vector as a vector from the body-fixed origin to an intersecting point on the straight line connecting two successive way-points, as opposed to the LOS vector defined in 3.1.1. Let us call this choice of LOS vector *LOS vector 2* and the approach in 3.1.1 *LOS vector 1*. The gain in using *LOS vector 2*, is that the environmental forces acting on the vehicle will to some extent be compensated for. To see how this works, picture that the vehicle is to track a path between two points in space, say p_0 and p_1, and *LOS vector 1* is chosen. If the vehicle is influenced by an environmental force such that its motion is no longer purely in the surge direction, the vehicle will no longer move towards p_1 on the straight line connecting p_0 and p_1 (the desired path). The heading will keep pointing towards p_1, but when the vehicle is far from p_1 its position will drift away from the desired path. As the vehicle approaches p_1 it will approach the desired path again and reach the circle of acceptance. In short, the cross-track error will be relatively large when the vehicle is far from the way-point it is tracking. If instead the *LOS vector 2* is used, the cross-track error will be smaller. This is because the point the vehicle is tracking is closer to the vehicle than in the case where the *LOS vector 1* is used. Thus the heading angle will be sharper in the sense that the difference between the vehicles heading angle and the angle of the path will be larger when the cross-track error is different from zero than when the *LOS vector 1* is used. This leads to faster reduction of the cross-track error, and to reduction of the effects of environmental disturbances. This is illustrated in Figure 1 and Figure 2 where an underactuated surface vessel, influenced by a constant current, is simulated using *LOS vector 1* and *LOS vector 2* respectively. The same controllers, initial conditions, current, and circle of acceptance are used in both simulations.

3.1.1 Rotation of the inertial frame.

The cross-track error e, the desired heading ψ_d, and the look-ahead distance Δ can be found in Figure 3, where $[x,y]$ denotes the Cartesian coordinates in the inertial reference frame, and $[x_b, y_b]$ denotes the Cartesian coordinates in the body fixed reference frame.

According to Alexey Pavlov et.al [8] the expression for the cross-track error e simplifies if the coordinate system is rotated an angle θ about the z-axis to align the x-axis with the path. In Figure 4 the rotated coordinate system together with new definitions for e and ψ_d is shown. The desired heading ψ_d and the cross-track error e are now defined as

$$e = y - D_y \tag{3.1.3}$$

$$\psi_d = -arctan(\frac{e}{\Delta}) \tag{3.1.4}$$

$$\dot{e} = \dot{y} = u \sin\psi + v \cos\psi \tag{3.1.5}$$

where:

$$\dot{D_y} = 0, \ \textit{since } D_y \textit{ is constant.} \tag{3.1.6}$$

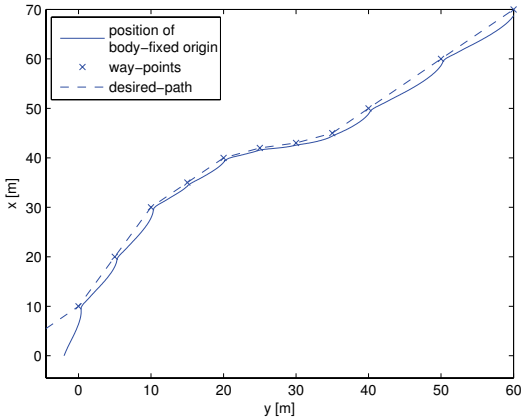

Figure 1: LOS vector 1.

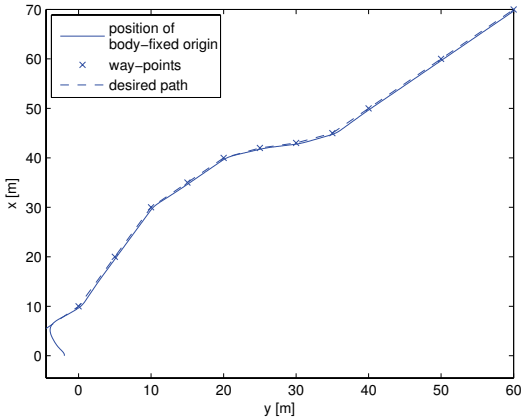

Figure 2: LOS vector 2.

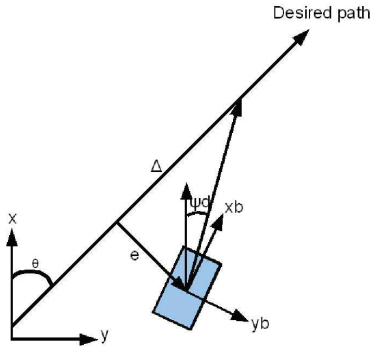

Figure 3: Delta and the cross track error.

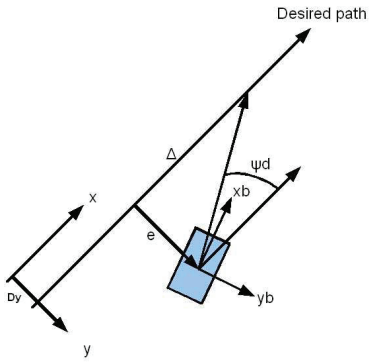

Figure 4: Rotated coordinate system.

NOTE: In equation 3.1.5 the heading ψ is relative to the rotated reference frame in Figure 4.

3.1.2 Condition on the look-ahead distance Δ

The LOS algorithm ensures that the cross-track error e converges to zero if the heading of the vehicle tracks the desired heading and heading rate returned by the LOS algorithm, if in addition some condition on the look-ahead distance Δ is met. This condition, and the resulting rate of convergence, depends on the chosen control laws. If the control laws presented in Fredriksen and Pettersen [2] are used to control the surge velocity and the heading, the states 3.1.7 are κ-exponentially stable if 3.1.8 holds. A proof of 3.1.8 can be found in Fredriksen and Pettersen [2].

$$\zeta = \begin{bmatrix} \overline{y} \\ \overline{v} \\ z_1 \\ z_2 \\ \overline{u} \end{bmatrix} \tag{3.1.7}$$

$$\Delta > \frac{X + u_c}{Y} \tag{3.1.8}$$

$$X = (\Theta u_c + M) < 0 \tag{3.1.9}$$

$$Y = (\Xi u_c + N) < 0 \tag{3.1.10}$$

The states 3.1.7 will be defined later in the report. The significance of the κ-exponential stability of 3.1.7 is that it implies that the cross-track error, the surge velocity and the heading are κ-exponential stable.

Note: As stated by Fredriksen and Pettersen [2], the assumption of X and Y negative puts restrictions (3.1.9 and 3.1.10) on the desired surge speed. *How hard these constraints on u_c are will depend on the specific ship. For most ships Θ, $\Xi \leq 0$, such that 3.1.9 - 3.1.10 give lower bounds on u_c.*

3.2 Way point switching, angle of the path and initial look-ahead distance

When the vessel is tracking the straight line path between two successive way-points, the reference frame is rotated so that the y-coordinate is equal to the cross-track error. To achieve this, the angle of the path (θ) in the inertial reference frame needs to be calculated, and the heading reference has to be set correctly in the inertial frame. This can be done by

$$\theta_k = arctan\left(\frac{y_{wp,k+1} - y_{wp,k}}{x_{wp,k+1} - x_{wp,k}}\right) \tag{3.2.1}$$

$$y = (x_I - x_{wp,k+1})\cos\theta_k + (y_I - y_{wp,k+1})\sin\theta_k \tag{3.2.2}$$

$$\psi_d = -arctan\left(\frac{y}{\Delta}\right) \tag{3.2.3}$$

where x_I, y_I are the Cartesian coordinates in the inertial reference frame, x_{wp}, y_{wp} are the way-points in the inertial reference frame, and y is the cross-track error. This choice ensures that the angle of the path is in the interval $[-\pi...\pi]$, while ψ_d is in the interval $[-\frac{3\pi}{2}...\frac{3\pi}{2}]$. A drawback to this choice is that when two successive lines in the path have angles which are more than π apart, the vessel may end up rotating the opposite direction of what would be the shortest way, or even a full 360 degrees. An example which illustrates this problem is shown in Figures 5 and 7. The desired path starts in the origin, see the circle in Figure 5, with an angle of $\theta_k = -\frac{3\pi}{4}$, and ends in [-18 0] marked by an x. The angle of the path line k+1, θ_{k+1}, is π. When the vessel starts tracking line k+1, the heading of the vessel will be approximately $-\frac{3\pi}{4}$ while the heading reference will step to π plus arctan(y/Δ), as can be seen in Figure 7. The consequence of this can be seen in Figure 5, the vessel rotates more than 360 degrees clockwise which causes the vessel to move away from the path, before the vessel starts tracking the path again. This behavior is not desirable, and can be avoided by adding some rule which will ensure that the angle of the path line at time k+1 is not more than

π different from the angle of the path line at time k. This can be done by exploiting that a heading angle ψ equals $2\pi \pm \psi$. An implementation of such a rule could be

$$\begin{aligned} &if \ \ |\theta_{k+1} - \theta_k| > \pi \\ &\quad \theta_{k+1} = sign(\theta_k)2\pi + \theta_{k+1} \\ &end \end{aligned}$$

If the next path angle is more than π different from the current path angle, set the next path angle equal to the equivalent angle closest to the previous angle. Figures 6 and 8 shows that the problem is removed when this rule is imposed. However, we do not want to allow heading or path angles much larger than 2π, thus another rule needs to be imposed to constrain these angles

$$\begin{aligned} &if \ \ |\psi_{k+1}| > 2\pi \\ &\quad \psi_{k+1} = \psi_{k+1} - 2\pi sign(\psi_{k+1}) \\ &\quad \theta_{k+1} = \theta_{k+1} - 2\pi sign(\psi_{k+1}) \\ &end \end{aligned}$$

If these two rules are used, the heading will stay in [-2π...2π], and the θ_{k+1} which is closest to $\theta_k$will always be chosen.

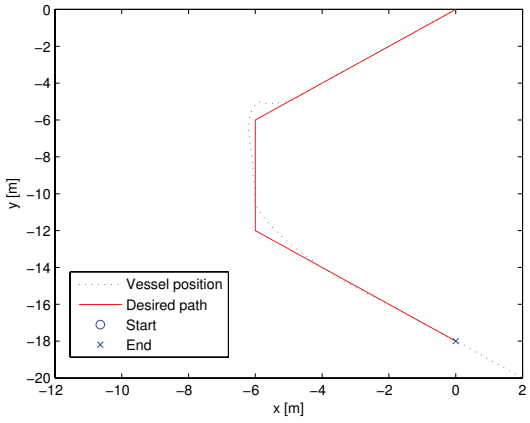

Figure 5: Vessel position where the reference frame rotation angle is not chosen with care.

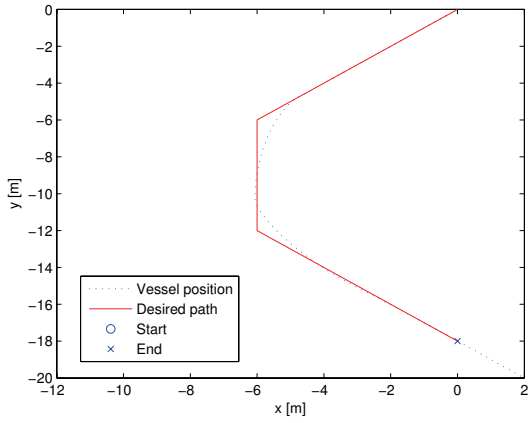

Figure 6: Vessel position where the reference frame rotation angle is chosen with care.

The heading reference in Figure 8 is better than that of Figure 7. However, the heading reference still has a large step during the transition from way-point k to way-point k+1. This can not be avoided when Δ is constant, however the step can be avoided when Δ is time varying. The idea is to choose Δ_{k+1} such that $\psi_{d,k} = \psi_{d,k+1}$ during the switch. At the next time steps k+2+i, Δ_{k+2+i} are chosen by the optimizer, and may increase or decrease rapidly, however, since $\Delta(t)$ is constrained so that it is continuous, the step in the heading reference will be avoided. The derivatives $\dot{\Delta}$ and $\ddot{\Delta}$ must be chosen so that $r_d = 0$ and $\dot{r}_d = 0$ during the switch, since the desired heading ψ_d should be constant during the switch.

By comparing Figures 8 and 9, it can be seen that the heading reference does not have the step during the transition between way-points when Δ and the derivatives of Δ are chosen such that the desired heading does not change until after the transition.

This restricts $|\theta_{k+1} - \theta_k| < \pi/2$ since there are no positive values of Δ which solves the problem for $|\theta_{k+1} - \theta_k| >= \pi/2$. One solution to the case where $|\theta_k + 1 - \theta_k| >= \pi/2$, is to set $\Delta = \Delta_{min}$, since this will be the value of Δ closest to the value which would remove the step in the desired heading. One could argue that if the path has to make a sharp turn, this turn should be split

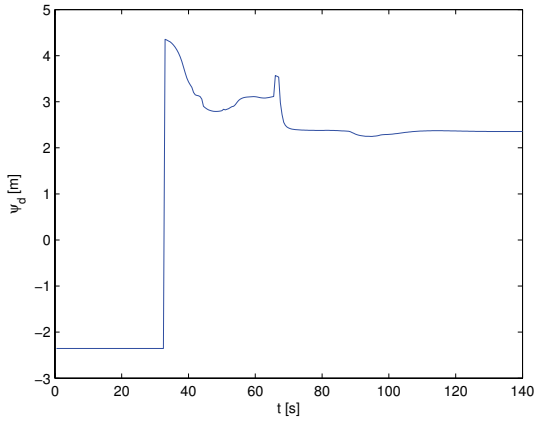

Figure 7: Heading reference from the case where the reference frame rotation angle is not chosen with care.

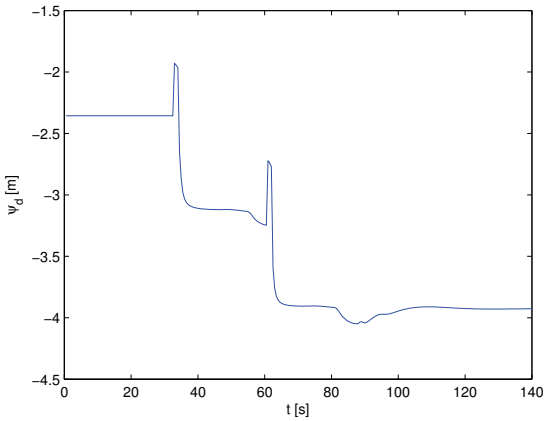

Figure 8: Heading reference from the case where the reference frame rotation angle is chosen with care, Δ not chosen to avoid the step in the heading.

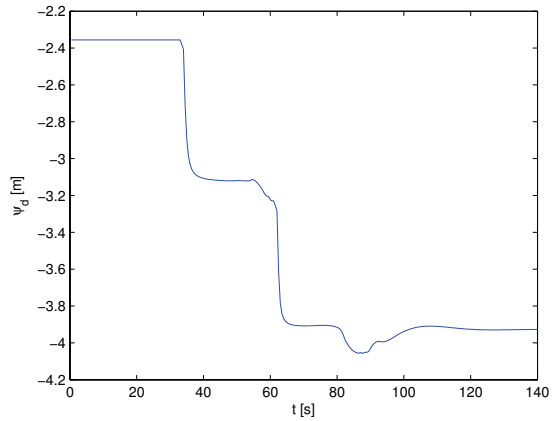

Figure 9: Heading reference from the case where the reference frame rotation angle is chosen with care, and Δ is chosen to avoid the step in the heading reference.

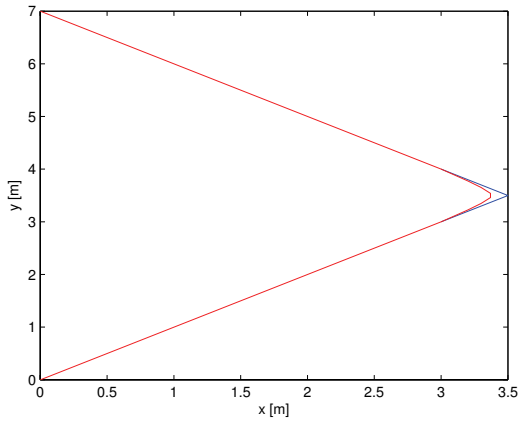

Figure 10: The red and blue path together is the original path, while the red path is the path where an extra line has been added to reduce the difference between the sub-path angles.

into several straight lines which does not have angles which differs as much as $\pi/2$, as illustrated in Figure 10, since large differences between the angles of two successive lines of the path will result in large over-shoots in the cross-track error. The way-points in this report are chosen such that $|\theta_{k+1} - \theta_k| < \pi/2$. If it is desired to allow $|\theta_{k+1} - \theta_k| > \pi/2$, the choice $\Delta = \Delta_{min}$ should be used, when two path lines have angles which differs by more than $\pi/2$.

Note: *The time variant look-ahead distance used to produce the figures 5-9, results from one of the optimization problems derived in this report.*

Note: *The results in Figure 5 are not feasible. The actual position would deviate more from the path. This is because the control inputs in the simulations leading to Figure 5 have very fast changes. However, Figure 5 illustrates the problem.*

4 Modelling

The 6 degrees of freedom (6-DOF) model of a surface vessel can be reduced to a 3-DOF model since the motions in heave , pitch and roll can be neglected. This is because heave, pitch and roll motions are small under normal conditions, and because they are stable due to restoring forces. This also means that they do not need to be controlled (though in some cases, as in large passenger vessels, stabilizing fins are used to damp these motions). Since there are no restoring forces in the remaining three degrees of freedom, the restoring forces term is neglected altogether.

4.1 Models used to simulate the vessel

A model for the underactuated surface vessel considered in this thesis can be found in *Thor I. Fossen 2002* [11]. This 3-DOF horizontal model describes the dynamics for surge, sway and yaw velocities, and the relation between these velocities and the geometric position and orientation.

$$\mathbf{M}\dot{\nu} + \mathbf{C}(\nu)\nu + \mathbf{D}\nu = \tau \tag{4.1.1}$$

$$\dot{\eta} = \mathbf{R}(\psi)\nu \tag{4.1.2}$$

$$\mathbf{M} = \begin{bmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & m_{23} \\ 0 & m_{32} & m_{33} \end{bmatrix}, \ \mathbf{D} = \begin{bmatrix} d_{11} & 0 & 0 \\ 0 & d_{22} & d_{23} \\ 0 & d_{32} & d_{33} \end{bmatrix}$$

$$\mathbf{C}(\nu) = \begin{bmatrix} 0 & 0 & -m_{22}v - m_{23}r \\ 0 & 0 & m_{11}u \\ m_{22}v + m_{23}r & -m_{11}u & 0 \end{bmatrix}, \ \mathbf{R}(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\tau = [\tau_u \ N_\delta \delta \ Y_\delta \delta]^T \tag{4.1.3}$$

When the vessel is subject to a constant irrotaional current, the vessel dynamics becomes [11]

$$\dot{\eta} = \mathbf{R}(\psi)\nu \tag{4.1.4}$$

$$M\dot{\nu} + \mathbf{C}(\nu)\nu + D\nu_r = \tau \tag{4.1.5}$$

$$\nu_r = \nu - \nu_{cur} \tag{4.1.6}$$

$$where$$

$$\nu_{cur} = \begin{bmatrix} u_{cur} \\ v_{cur} \\ r \end{bmatrix} \tag{4.1.7}$$

$$u_{cur} = V_{cur} cos(\beta_{cur} - \psi) \tag{4.1.8}$$

$$v_{cur} = V_{cur} sin(\beta_{cur} - \psi) \tag{4.1.9}$$

$$V_{cur} = \sqrt{u_{cur}^2 + v_{cur}^2} \tag{4.1.10}$$

In both models $[x \ y]^T$ is the position of the vessel in the inertial reference frame and ψ is the yaw angle of the vessel (the orientation of the vessel relative to the inertial frame, commonly referred to as the heading). The vector ν is the velocity vector where u is the surge velocity along the body-fixed x-axis, v is the sway velocity along the body-fixed y-axis and r is the angular velocity about the body-fixed z-axis (yaw rate/heading rate).

4.2 Models used for predictions

The MPC algorithm uses predictions of the states to determine the optimal inputs. These predictions are made by integrating the states over the prediction horizon, inserting the inputs at each sample time k. The models used for simulation may also be used for prediction. However, the models used

for simulation are expressed in matrix form and the time spent integrating such models will be large compared to the time spent integrating models expressed by one differential equation for each state. This is because matrix multiplication has greater overhead than scalar multiplication. Thus, it is desirable to express each state by one differential equation. The dynamic for each state in the case of no disturbance becomes

$$\dot{u} = \frac{1}{m_{11}}\Big((m_{22}v + m_{23}r)r - d_{11}u\Big) + \frac{1}{m_{11}}\tau_u \tag{4.2.1}$$

$$\dot{v} = X(u)r + Y(u)v + F(\delta) \tag{4.2.2}$$

$$\dot{r} = \Omega(u)r + \Phi(u)v + \frac{\delta}{m_{22}m_{33} - m_{23}^2}(m_{22}N_\delta - m_{23}Y_\delta) \tag{4.2.3}$$

$$\dot{x} = u\cos\psi - v\sin\psi \tag{4.2.4}$$

$$\dot{y} = u\sin\psi + v\cos\psi \tag{4.2.5}$$

$$\dot{\psi} = r \tag{4.2.6}$$

where δ is the control input in yaw and $F(\delta)$ is a term which represents the influence of the rudder on the sway dynamics. The terms $X(u)$, $\Omega(u)$, $Y(u)$, and $\Phi(u)$ are linear terms in the surge velocity u. Their expressions are quite lengthy and not of interest in this report since the model will be re-written to find the rudder independent dynamics in sway.

The dynamic for each state in the case of a constant irrotaional current

$$\dot{u} = \frac{1}{m_{11}}\Big((m_{22}v + m_{23}r)r - d_{11}u_r\Big) + \frac{1}{m_{11}}\tau_u \tag{4.2.7}$$

$$\dot{v} = X(u)r + Y_1(u)v + Y_2(u)v_r + F(\delta) \tag{4.2.8}$$

$$\dot{r} = \Omega(u)r + \Phi_1(u)v + \Phi_2(u)v_r + \frac{\delta}{m_{22}m_{33} - m_{23}^2}(m_{22}N_\delta - m_{23}Y_\delta) \tag{4.2.9}$$

$$\dot{x} = u\cos\psi - v\sin\psi \tag{4.2.10}$$

$$\dot{y} = u\sin\psi + v\cos\psi \tag{4.2.11}$$

$$\dot{\psi} = r \tag{4.2.12}$$

where the terms $X(u)$, $\Omega(u)$, $Y_1(u)$, $Y_2(u)$, $\Phi_1(u)$, and $\Phi_2(u)$ are linear terms in the surge velocity u, where the linear terms $Y(u)$ and $\Phi(u)$ have been split into the parts not effected by the current $(Y_1(u), \Phi_1(u))$ and the parts effected by the current $(Y_2(u), \Phi_2(u))$.

4.2.1 Rudder independent sway dynamics

Unfortunately the un-actuated sway dynamics is influenced by the rudder through $F(\delta)$. This complicates the model, and we wish to remove this effect from the sway dynamics. This can be done by a coordinate transformation found in *E. Fredriksen and K.Y Pettersen* [2]. First the equilibrium point of the surge velocity is shifted to the desired velocity u_d

$$\overline{u} = u - u_d \tag{4.2.13}$$

then according to *E. Fredriksen and K.Y Pettersen* [2]:[1] *To decouple the transformed sway dynamics from the rudder control, we use the following coordinate transformation, inspired by Do and Pan (2003a), which removes the effect from δ in the $\overline{v}$-dynamics:*

$$\overline{x} = x + \varepsilon\cos\psi \tag{4.2.14}$$

$$\overline{y} = y + \varepsilon\sin\psi \tag{4.2.15}$$

$$\overline{v} = v + \varepsilon r \tag{4.2.16}$$

Where:

$$\varepsilon = -\left(\frac{m_{33}Y_\delta - m_{23}N_\delta}{m_{22}N_\delta - m_{23}Y_\delta}\right) \tag{4.2.17}$$

This corresponds to moving the origin along the x-axis of the body-fixed coordinate system to that point where the rudder gives only a rotational moment and no sway force. The transformed system equations are

$$\dot{\overline{y}} = \sin(\psi)(\overline{u} + u_d) + \cos(\psi)\overline{v}, \tag{4.2.18}$$

$$\dot{\psi} = r, \tag{4.2.19}$$

$$\begin{aligned}\dot{\overline{v}} &= \dot{v} + \varepsilon\dot{r} \\ &= (\Upsilon\overline{u} + \Theta u_d + M)r + (\Lambda\overline{u} + \Xi u_d + N)\overline{v},\end{aligned} \tag{4.2.20}$$

$$\dot{r} = \frac{\delta}{\Gamma}(m_{22}N_\delta - m_{23}Y_\delta) + \Omega r + F\overline{v}, \tag{4.2.21}$$

$$\dot{\overline{u}} = \frac{1}{m_{11}}(\tau_u + (m_{22}v + m_{23}r)r - d_{11}u) \tag{4.2.22}$$

in the case of no disturbance, while in the case of a constant irrotaional current, the system equations becomes

$$\dot{\overline{y}} = \sin(\psi)(\overline{u} + u_d) + \cos(\psi)\overline{v}, \tag{4.2.23}$$

$$\dot{\psi} = r, \tag{4.2.24}$$

$$\begin{aligned}\dot{\overline{v}} &= \dot{v} + \varepsilon\dot{r} \\ &= (\Upsilon\overline{u} + \Theta u_d + M)r + (\Lambda\overline{u} + \Xi u_d)\overline{v} + N\overline{v}_r,\end{aligned} \tag{4.2.25}$$

$$\dot{r} = \frac{\delta}{\Gamma}(m_{22}N_\delta - m_{23}Y_\delta) + \Omega r + F_1\overline{v}_r + F_2\overline{v}, \tag{4.2.26}$$

$$\dot{\overline{u}} = \frac{1}{m_{11}}(\tau_u + (m_{22}v + m_{23}r)r - d_{11}u_r) \tag{4.2.27}$$

where

$$\overline{v}_r = v_r + \varepsilon r \tag{4.2.28}$$

Where the term F in equation 4.2.21 has been split into the two parts F_1 and F_2 in equation 4.2.26, where F_2 is the part of F not effected by the current while F_1 is the part of F effected by the current. We also have from [2]

$$\Gamma = m_{22}m_{33} - m_{23}^2 > 0, \tag{4.2.29}$$

$$\Theta = \frac{1}{\Gamma}(-2m_{23}m_{22}\varepsilon + m_{22}^2\varepsilon^2 + m_{23}^2 - m_{22}m_{11}\varepsilon^2 - m_{33}m_{11} + 2m_{23}m_{11}\varepsilon), \tag{4.2.30}$$

$$\begin{aligned}M = &\frac{1}{\Gamma}(m_{23}d_{33} - \varepsilon m_{22}d_{33} - m_{33}d_{23} - m_{23}d_{32}\varepsilon \\ &+ \varepsilon m_{23}d_{23} + m_{22}d_{32}\varepsilon^2 + m_{33}d_{22}\varepsilon - m_{23}d_{22}\varepsilon^2),\end{aligned} \tag{4.2.31}$$

$$\Xi = \frac{1}{\Gamma}(m_{22}m_{11}\varepsilon - m_{22}^2\varepsilon - m_{23}m_{11} + m_{22}m_{23}), \tag{4.2.32}$$

$$N = \frac{1}{\Gamma}(m_{23}d_{22}\varepsilon - m_{22}d_{32}\varepsilon - m_{33}d_{22} + m_{23}d_{32}), \tag{4.2.33}$$

$$\begin{aligned}\Omega = &\frac{1}{\Gamma}(m_{23}m_{11}(\overline{u} + u_d) + m_{22}^2(\overline{u} + u_d)\varepsilon - m_{23}d_{22}\varepsilon + m_{23}d_{23} \\ &- m_{22}m_{11}(\overline{u} + u_d)\varepsilon - m_{22}m_{23}(\overline{u} + u_d) - m_{22}d_{33} + m_{22}d_{32}\varepsilon),\end{aligned} \tag{4.2.34}$$

$$F = \frac{1}{\Gamma}(m_{23}d_{22} - m_{22}^2(\overline{u} + u_d) - m_{22}d_{32} + m_{22}m_{11}(\overline{u} + u_d)). \tag{4.2.35}$$

where F has been divided into F_1 and F_2

$$F_1 = \frac{1}{\Gamma}(m_{23}d_{22} - m_{22}d_{32}) \tag{4.2.36}$$

$$F_2 = \frac{1}{\Gamma}(m_{22}m_{11}(\overline{u} + u_d) - m_{22}^2(\overline{u} + u_d) \tag{4.2.37}$$

[1] The quote from E. Fredriksen and K.Y Pettersen [2] cites a paper by Do and Pan (Do and Pan (2003a)) in this thesis the cited paper is listed under Do and Pan [12].

4.2.2 Prediction models expressed with Δ, $\dot{\Delta}$, $\ddot{\Delta}$, u_c and $\dot{u}_c$ as inputs

The MPC algorithms developed in this thesis does not calculate the control inputs, but the parameters Δ, $\dot{\Delta}$, $\ddot{\Delta}$, the reference u_d and $\dot{u}_d$. Thus, control laws to stabilize the system are needed. These control laws will be inserted into the equations for the system dynamics so that the models used for prediction are expressed in terms of these inputs. The control laws presented in *E. Fredriksen and K.Y Pettersen* [2] uses the LOS algorithm to set the heading reference, and they yield global κ-exponential convergence of the system. However, these control laws are expressed for the case of Δ and u_d constant. Thus, the control laws needs to be changed to take into account that Δ and u_d are no longer constant. This may have consequences for the κ-exponential stability of the system. That is, it it is not known whether the proofs for κ-exponential stability of the system still holds. Thus the resulting control laws should be analyzed to prove stability. However, this will not be addressed in this thesis.

The control laws presented in [2] are

define

$$\psi_d = -arctan(\frac{\overline{y}}{\Delta}) \tag{4.2.38}$$

$$r_d = \dot{\psi}_d = -\frac{\Delta\dot{\overline{y}}}{\Delta^2 + \overline{y}^2} \tag{4.2.39}$$

$$z_1 = \psi - \psi_d \tag{4.2.40}$$

$$z_2 = \dot{z_1} = r - r_d \tag{4.2.41}$$

$$\dot{r}_d = \frac{2\Delta\overline{y}(\dot{\overline{y}})^2}{(\Delta^2 + \overline{y}^2)^2} - \frac{\Delta\ddot{\overline{y}}}{\Delta^2 + \overline{y}^2} \tag{4.2.42}$$

then

$$\tau_u = -(m_{22}v + m_{23}r)r + d_{11}u - m_{11}k_u\overline{u} \tag{4.2.43}$$

$$\delta = \frac{\Gamma}{(m_{22}N_\delta - m_{23}Y_\delta)}\Big(- \Omega r - F\overline{v} + \dot{r}_d - k_1 z_2 - k_0 z_1\Big) \tag{4.2.44}$$

where

$$\ddot{\overline{y}} = r\cos\psi(\overline{u} + u_d) + \sin\psi\dot{\overline{u}} - r\sin\psi\overline{v} + \cos\psi\dot{\overline{v}} \tag{4.2.45}$$

E. Fredriksen and K.Y Pettersen [2] has chosen $\psi_d = -arctan(\frac{\overline{y}}{\Delta})$ as opposed to the heading reference suggested in the section *Line of sight and Way points*: $\psi_d = -arctan(\frac{y}{\Delta})$. The relation between $\overline{y}$ and y is: $\overline{y} = y + \varepsilon\sin\psi$, which means that when $\overline{y}$ converges to zero, so does y. However, when the cost function for the MPC problem is defined, it is the cross-track error y and not the transformed coordinate $\overline{y}$ which should be minimized.

When the look-ahead distance and the surge reference are time variant, the expressions for r_d, $\dot{r}_d$ and $\ddot{\overline{y}}$, equations 4.2.39, 4.2.42 and 4.2.45 respectively, are no longer correct. The expressions for r_d, $\dot{r}_d$ and $\ddot{\overline{y}}$ has to be developed where Δ and u_d are not assumed constant.

$$\psi_d = -arctan(\frac{\overline{y}}{\Delta}) \tag{4.2.46}$$

$$r_d = \dot{\psi}_d = \frac{\dot{\Delta}\overline{y} - \Delta\dot{\overline{y}}}{\Delta^2 + \overline{y}^2} \tag{4.2.47}$$

$$\dot{r}_d = \frac{(\Delta\dot{\overline{y}} - \dot{\Delta}\overline{y})(2\Delta\dot{\Delta} + 2\overline{y}\dot{\overline{y}})}{(\Delta^2 + \overline{y}^2)^2} + \frac{\ddot{\Delta}\overline{y} - \Delta\ddot{\overline{y}}}{\Delta^2 + \overline{y}^2} \tag{4.2.48}$$

$$\ddot{\overline{y}} = r\cos\psi(\overline{u} + u_d) + \sin\psi(\dot{\overline{u}} + \dot{u}_d) - r\sin\psi\overline{v} + \cos\psi\dot{\overline{v}} \tag{4.2.49}$$

NOTE: *The surge velocity reference is constant in some of the MPC algorithms suggested, in these algorithms $\ddot{\overline{y}}$ from equation 4.2.45 is used in the control law and in the prediction model.*

The control law δ can now be changed to take the time varying property of Δ and u_d into account by inserting the new expressions for r_d, $\dot{r}_d$ and $\ddot{\overline{y}}$, equations 4.2.47, 4.2.48 and 4.2.49 respectively,

into equation 4.2.44. The control law τ_u in equation 4.2.43 does not need to be changed.

The system equations 4.2.18-4.2.22 and 4.2.23-4.2.27 can be simplified by inserting the control laws into the equations. This will also be necessary to express the cost function in the linear quadratic problem formulation, where the cost function has to be linear quadratic in the inputs ($\Delta(t)$, $\dot{\Delta}$...). The predictions used by the non-linear solver can be made by integration of the vessel model where the control-laws are not inserted into the state dynamics. However, the calculation time of the predictions is reduced when the dynamics for each state is written as scalar equations. Further more, the prediction model has to be continuous with continuous derivatives since the optimization problem solver uses numerical estimates of the gradient and the Hessian of the cost-function. The derivatives of the cost function will not be continuous if the control-inputs in the prediction model are constrained. Any constraints in the control-inputs have to be added as constraints on the free variables in the optimization problem if they are to be respected. Thus, all predictions in this thesis will be made by integration of the simplified model where the control-laws are inserted into the state equations.

The effect of the current will be small in surge and yaw due to the corrective terms in the respective control-laws 4.2.43 and 4.2.44. Thus, the prediction model neglects the effect of the current in surge and yaw. However, the effect of the current in the sway velocity will be too large to neglect since the sway velocity is not controlled. Thus, the effect of the current in sway will be included in the prediction model. This can be achieved by estimating the current and predict the effect of the current based on this estimate.

The resulting system equations used to form the predictions are then

$$\psi_d = -arctan(\frac{\overline{y}}{\Delta}) \tag{4.2.50}$$

$$r_d = \dot{\psi}_d = \frac{\dot{\Delta}\overline{y} - \Delta\dot{\overline{y}}}{\Delta^2 + \overline{y}^2} \tag{4.2.51}$$

$$\dot{r}_d = \frac{(\Delta\dot{\overline{y}} - \dot{\Delta}\overline{y})(2\Delta\dot{\Delta} + 2\overline{y}\dot{\overline{y}})}{(\Delta^2 + \overline{y}^2)^2} + \frac{\ddot{\Delta}\overline{y} - \Delta\ddot{\overline{y}}}{\Delta^2 + \overline{y}^2} \tag{4.2.52}$$

$$\dot{\overline{y}} = \sin(\psi)(\overline{u} + u_d) + \cos(\psi)\overline{v}, \tag{4.2.53}$$

$$\dot{y} = \sin(\psi)(\overline{u} + u_d) + \cos(\psi)v, \tag{4.2.54}$$

$$\dot{\psi} = r, \tag{4.2.55}$$

$$\dot{\overline{v}} = \begin{cases} (\Upsilon\overline{u} + \Theta u_d + M)r + (\Lambda\overline{u} + \Xi u_d)\overline{v} + N\overline{v}_r, \\ (\Upsilon\overline{u} + \Theta u_d + M)r + (\Lambda\overline{u} + \Xi u_d + N)\overline{v}, \end{cases} \tag{4.2.56}$$

$$\dot{r} = \dot{r}_d - k_1 z_2 - k_0 z_1 \tag{4.2.57}$$

$$\dot{\overline{u}} = -k_u\overline{u} \tag{4.2.58}$$

$$\dot{v} = \dot{\overline{v}} - \varepsilon r \tag{4.2.59}$$

$$\ddot{\overline{y}} = \begin{cases} r\cos\psi(\overline{u} + u_d) + \sin\psi(\dot{\overline{u}} + \dot{u}_d) - r\sin\psi\overline{v} + \cos\psi\dot{\overline{v}} \\ r\cos\psi(\overline{u} + u_d) + \sin\psi\dot{\overline{u}} - r\sin\psi\overline{v} + \cos\psi\dot{\overline{v}} \end{cases} \tag{4.2.60}$$

Notice that the effect of the current on the sway velocity is included in equation 4.2.56. The inputs will be renamed to g_i for simplicity, where

$$\begin{aligned} g_1 &= \Delta \\ g_2 &= \dot{\Delta} \\ g_3 &= \ddot{\Delta} \\ g_4 &= u_d \\ g_5 &= \dot{u}_d \end{aligned}$$

Note: *The transformed states $\overline{y}$ and $\overline{v}$ should not be predicted by integrating their respective state equations. Better accuracy is achieved if they are predicted by evaluating 4.2.15 and 4.2.16. However, the dynamics for these states are used to form the LTV model.*

4.2.3 Discrete Linear time variant model

An LTV model is a model which is linear in the states and the inputs, but where the linear dependencies vary with time. The models used for the underactuated surface vessel in this thesis are non-linear. These models can be linearized about one operating point, however such linear models would only be valid close to the operating point. Linear models which are valid for several operating points can be obtained if the models are linearized about several operating points to form LTV models. These operating points should be chosen such that the states of the vessel at time k are close to operating point k. This is achieved by predicting the states of the vessel by numerical integration of the non-linear system, inserting the predicted inputs at each sample time, and linearizing about the predicted states and inputs at each sample time. The choice of predicted inputs will be discussed in the chapter *MPC*.

When the predictions of the states are made, the predicted input vector needs to be extended so that its length is equal to the prediction horizon. Let us denote the input vector used for predictions at time k as $U_{pred,k}$. The input vector can be extended by keeping the inputs g_1 and g_4 constant, and inputs $g_i = 0, i = 2, 3, 5$, after the input horizon

$$U_{pred,k} = \begin{bmatrix} g_{i,k} \\ g_{i,k+1} \\ g_{i,k+2} \\ \vdots \\ g_{i,k+H_u-2} \\ g_{i,k+H_u-1} \\ g_{1,k+H_u-1} \\ 0 \\ 0 \\ g_{4,k+H_u-1} \\ 0 \\ \vdots \\ g_{1,k+H_u-1} \\ 0 \\ 0 \\ g_{4,k+H_u-1} \\ 0 \end{bmatrix}, dim(U_{pred,k}) = 5H_p \tag{4.2.61}$$

The predicted states, $X_{pred,k}$, are found from $U_{pred,k}$ by integration of the system equations, inserting the inputs at each sample time.

$$x = \int f(x, g)dt \tag{4.2.62}$$

$$X_{pred,k} = \begin{bmatrix} x_k \\ x_{k+1} \\ \vdots \\ x_{k+H_u-1} \\ \vdots \\ x_{k+H_p-1} \end{bmatrix}, dim(X_{pred,k}) = nH_p \tag{4.2.63}$$

Where n is the number of states. Accuracy will depend on the chosen numerical integration, and on the choice of step length in the numerical integrator.

A discrete LTV model for the system can now be obtained by using, e.g. forward Euler

$$x_{k+1} = \Phi_k x_k + \Gamma_k u_k \tag{4.2.64}$$
$$y_k = C x_k \tag{4.2.65}$$

$$\Phi_k = I + T_s \frac{\partial f(x,g)}{\partial x}\bigg|_{x=x_{pred,k}, g=g_{pred,k}} \tag{4.2.66}$$
$$\Gamma_k = T_s \frac{\partial f(x,g)}{\partial g}\bigg|_{x=x_{pred,k}, g=g_{pred,k}} \tag{4.2.67}$$

where in our case

$$x = \begin{bmatrix} y \\ \overline{y} \\ \psi \\ v \\ \overline{v} \\ r \\ \dot{y} \\ u \end{bmatrix} \tag{4.2.68}$$

$$g = \begin{bmatrix} \Delta \\ \dot{\Delta} \\ \ddot{\Delta} \\ u_d \\ \dot{u}_d \end{bmatrix} \tag{4.2.69}$$

The states $x_{pred,k}$ and inputs $g_{pred,k}$ are the samples from the predictions $X_{pred,k}$ and $U_{pred,k}$ respectively. Notice that Φ_{k+1} is found from $x_{pred,k+1}$ in $X_{pred,k}$, not from $X_{pred,k+1}$.

The expressions for the partial derivatives $\frac{\partial f(x,g)}{\partial x}$ and $\frac{\partial f(x,g)}{\partial g}$ in the case where $u = u_d$ and the states $x = [y, \overline{y}, \psi, v, \overline{v}, r]$ are estimated, can be found in *Appendix A*.

5 Observer for estimating the current

The vessel will be subjected to a constant irrotational current in some of the simulations. Hence, the predictions of the future states will have increased accuracy if the current is estimated so that the effect of the current can be taken into account. The current will be estimated by using the nonlinear passive observer presented in *Thor I. Fossen* [11]

$$\dot{\hat{\xi}} = A_w\hat{\xi} + K_1(\omega_0)\tilde{y} \tag{5.0.1}$$
$$\dot{\hat{\eta}} = R(y_3)\hat{\nu} + k_2\tilde{y} \tag{5.0.2}$$
$$\dot{\hat{b}} = -T^{-1}\hat{b} + K_3\tilde{y} \tag{5.0.3}$$
$$M\dot{\hat{\nu}} = -D\hat{\nu} + R^T(y_3)\hat{b} + \tau + R^T(y_3)K_4\tilde{y} \tag{5.0.4}$$
$$\hat{y} = \hat{\eta} + C_w\hat{\xi} \tag{5.0.5}$$

where $\tilde{y} = y - \hat{y}$, y contains the state measurements, $k_i, i = 1, 2, 3, 4$ are observer gain matrices, b is the bias term and states $\hat{x}$ are estimated states. The term $-T^{-1}\hat{b}$ represents low-pass filtering of the bias term. Since waves will not be included in this thesis, the observer is reduced to

$$\dot{\hat{\eta}} = R(y_3)\hat{\nu} + k_2\tilde{y} \tag{5.0.6}$$
$$\dot{\hat{b}} = -T^{-1}\hat{b} + K_3\tilde{y} \tag{5.0.7}$$
$$M\dot{\hat{\nu}} = -D\hat{\nu} - C(\hat{\nu})\hat{\nu} + R^T(y_3)\hat{b} + \tau + R^T(y_3)K_4\tilde{y} \tag{5.0.8}$$
$$\hat{y} = \hat{\eta} \tag{5.0.9}$$

notice that the Coriolis matrix also is included in equation 5.0.8 while it is not in eqution 5.0.4. This is because the observer in [11] is based on a model which dose not include this matrix. The observer gains

$$k_2 = diag\{K_{21}, K_{22}, K_{23}\}$$
$$k_3 = diag\{K_{31}, K_{32}, K_{33}\}$$
$$k_4 = diag\{K_{41}, K_{42}, K_{43}\}$$

have to satisfy [11]

$$1/T_i \ll K_{3i}/K_{4i} < K_{2i}$$

where

$$1/T_i \gg 1$$

The bias term 5.0.7 represents the effect of the current. The estimates of the current parameters $\hat{V}_{cur}$ and $\hat{\beta}_{cur}$ are calculated from the bias term in order to include the effect of the current in the predictions. That is

$$\begin{aligned} D\hat{\nu}_{cur} &= R^T(y_3)\hat{b} \\ \hat{\nu}_{cur} &= D^{-1}R^T(y_3)\hat{b} \\ &= \begin{bmatrix}\hat{\nu}_{cur,1} & \hat{\nu}_{cur,2} & \hat{\nu}_{cur,3}\end{bmatrix} \end{aligned}$$

$$\begin{aligned} \nu_{cur} &= \begin{bmatrix}V_{cur}cos(\beta_{cur} - \psi) & V_{cur}sin(\beta_{cur} - \psi) & r\end{bmatrix} \\ \hat{\beta}_{cur} &= atan(\frac{\hat{\nu}_{cur,2}}{\hat{\nu}_{cur,1}}) + \psi \\ \hat{V}_{cur} &= \frac{\hat{\nu}_{cur,1}}{cos(\hat{\beta}_{cur} - \psi)} \end{aligned}$$

The following observer parameters gives good estimates of the current

$$T = \begin{bmatrix} 40 & 0 & 0 \\ 0 & 40 & 0 \\ 0 & 0 & 400 \end{bmatrix}$$
$$k_2 = \begin{bmatrix} 20 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 6 \end{bmatrix}$$
$$K_3 = \begin{bmatrix} 60000 & 0 & 0 \\ 0 & 60000 & 0 \\ 0 & 0 & 100 \end{bmatrix}$$
$$k_4 = \begin{bmatrix} 4000 & 0 & 0 \\ 0 & 4000 & 0 \\ 0 & 0 & 25 \end{bmatrix}$$

The estimates of the parameters of *current1* is seen in Figure 11, the estimates are the blue lines and the actual parameter values are the red lines. The estimates do vary quite much initially, however, since the estimates are sampled, they remain constant in the predictions. The estimates are stable and very close to the actual values after time $t \approx 20$ seconds. Notice the small oscillations at time $t \approx 88.5$ seconds and $t \approx 175,5$ seconds. These oscillations are caused by the switch between way-points.

Note: *The estimated current is relative to the inertial reference fram.*

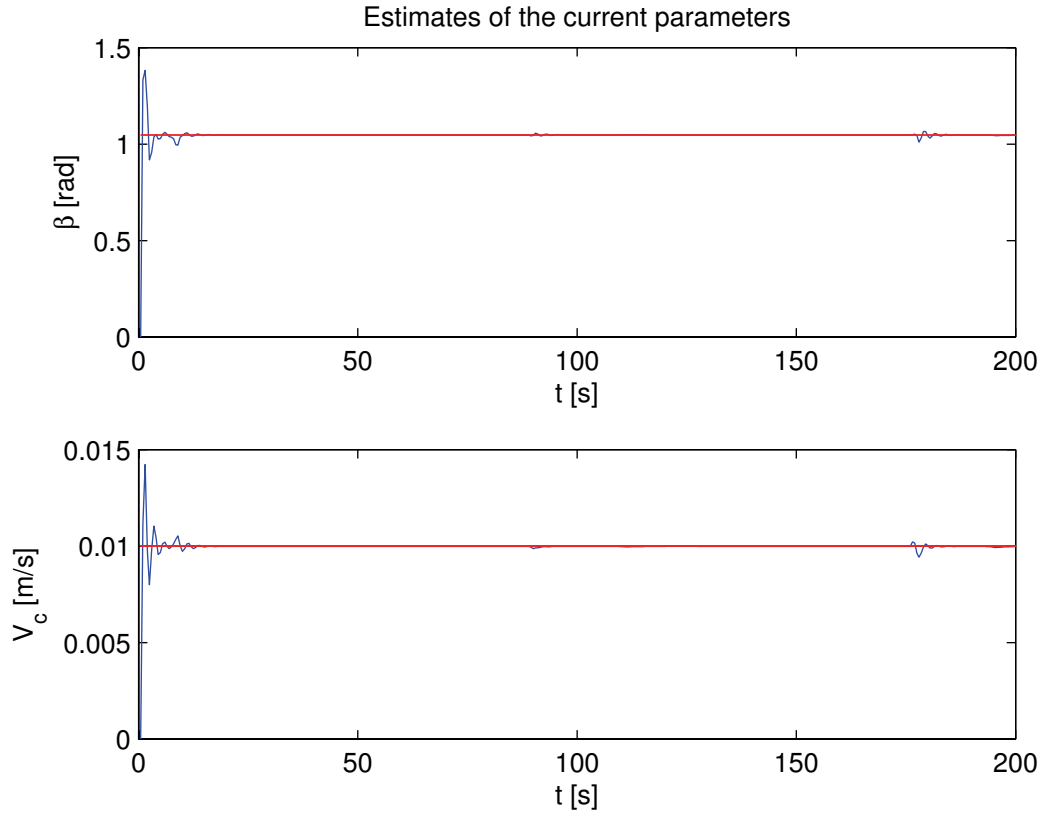

Figure 11: Estimated current parameters, *current1*.

6 MPC

An approach to solving a non-linear MPC problem (NMPC), presented in the paper by *Giancarlo Marafioti et al.* [14], is to linearize the system about the predicted response of the system (i.e about the response to good guesses of the inputs), and representing the discrete system as linearly time-variant (LTV). This choice, as opposed to a linear time invariant system, reduces the errors introduced by the linearization and has the effect of re-linearization about estimated operating-points. The reason for linearizing the system is that by doing so, the optimization problem can be written as a QP-problem[2], a convex optimization problem is obtained if Q is positive semidefinite, and the problem has one unique solution if the initial point is feasible. The cost function has to be quadratic in the variables which the QP problem is solved with respect to. The problem in this thesis is solved with respect to the inputs Δ, $\dot{\Delta}$, $\ddot{\Delta}$, u_d and $\dot{u}_d$. Thus, the cost-functions presented in this report has to be re-written in terms of the inputs, using the LTV model.

A different approach is to form predictions by numerical integration of the nonlinear discrete model and use these predictions to evaluate the cost function and the constraints. These evaluations of the cost function are used to form numerical estimates of the Hessian and the gradient of the cost function, which in turn are used to find the minimum. This approach has the gain of better accuracy in the predictions than a LTV model, but at the cost of increased calculation time and no guarantee of convexity, which leads to no guarantee of convergence to the global minimum. Another gain is that the constraints can be non-linear, and that the cost-functions presented in this report do not need to be rewritten since they are evaluated by inserting the predictions.

6.1 Convexity

An important property of a cost function is convexity. From *Jorge Nocedal and Stephen J. Wright* [9] we have that if the minimization algorithm converge to a stationary point of the cost function f, we know the following about the solution; *if we know that f is convex, then we can be sure that the algorithm has converged to a global minimizer.* The definition of convexity can be formulated as by *Jorge Nocedal and Stephen J. Wright* [9]

> *$S \in \mathbb{R}^n$ is a convex set if the straight line segment connecting any two points in S lies entirely inside S. Formally, for any two points $x \in S$ and $y \in S$, we have $\alpha x + (1-\alpha)y \in S$ for all $\alpha \in [0,1]$.*
>
> *f is a convex function if it's domain is a convex set and if for any two points x and y in this domain, the graph of f lies below the straight line connecting (x, f(x)) to (y, f(y)) in the space $\mathbb{R}^{n+1}$. That is, we have*
>
> $$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y), \quad \text{for all } \alpha \in [0,1]$$

[1] A QP-problem is a Quadratic Programming problem, and has linear constrains and the objective function (cost-function) is quadratic.

6.2 Receding Horizon

The goal of a model predictive controller is to bring the state measurements from their current states to their respective set-points along an ideal trajectory. This trajectory is defined as the *reference trajectory* by *J. M. Maciejowski* [10] and is frequently assumed to approach the set-point exponentially. The predictive controller finds the input trajectory which results in the best behavior of the system by predicting the response of the system to the input trajectory over the *prediction horizon*. The prediction horizon is the time length into the future which the predictions are made for. This input trajectory is a prediction of the optimal future inputs. The inputs $k+i$ found at time k are likely to be different at time $k+i$, as stated by *J. M. Maciejowski* [10].

The predicted optimal inputs are calculated once for each sample time k. Only the first input is applied to the system at time k, since new optimal inputs are predicted at times $k+i$. In short, at time k the states are measured, the ideal trajectories of the states from the current states to the set-points are found by predicting the response of the states to the input trajectory. The first input in the predicted optimal input trajectory is applied to the system. At time $k+1$ new measurements are taken and the procedure is repeated. As explained by *J. M. Maciejowski* [10]: "*Since the prediction horizon remains of the same length as before, but slides along by one sample interval at each step, this way of controlling a plant is often called a* receding horizon *strategy.*"

6.3 Main algorithm, LTV model

The MPC algorithm suggested by *Giancarlo Marafioti et al.* [14] is slightly altered to fit the problem in this report, and used to find the optimal inputs when the LTV model is used. The difference is that the weights in this report are constant, and that the states in this report are not estimated. In this algorithm, lets call it *Algorithm 1*, $U_{pred,k}$ is the initial guess of the optimal input vector U_k^*, while U_{k-1}^* is the previously found optimal input vector. The optimal solution at the previous step is shifted once and used as the initial guess of the optimal input vector. This will be explained with greater detail in the section *Choice of initial value* $U_{k,0}$.

Algorithm 1:

For a general time step k

Choose $U_{pred,k}$ *given* U_{k-1}^*
Use $U_{pred,k}$ *and* x_k *to compute* $X_{pred,k}$*, within the prediction horizon* H_p
Compute the Jacobians Φ_k *and* Γ_k *(4.2.66 and 4.2.67) about the predicted* $X_{pred,k}$ *and* $U_{pred,k}$
Find the optimal solution of the corresponding QP optimization problem
From the optimal solution $U_k^* = \{g_{1,1}^*, g_{2,1}^*, g_{3,1}^*, g_{4,1}^*, g_{5,1}^* \dots, g_{1,H_p}^*, g_{2,H_p}^*, g_{3,H_p}^*, g_{4,H_p}^*, g_{5,H_p}^*\}$
apply $g_{1,1}^*, g_{2,1}^*, g_{3,1}^*, g_{4,1}^*, g_{5,1}^*$ *to the system*

If the desired surge velocity is kept constant, all inputs $g_{i,j}, i = 4, 5$ are not used or calculated.

6.4 Choice of initial value $U_{k,0}$

It is desirable to have the initial value close to the optimum. If this is achieved, the number of iterations needed to solve the QP-problem will be reduced. *Giancarlo Marafioti et al.* [14] suggests that a good initial choice for the input vector is to choose the tail of the previous optimal input

vector. That is, for k=2 set

$$U_{pred,k} = \begin{bmatrix} g^*_{i,2} \\ g^*_{i,3} \\ \vdots \\ g^*_{i,H_u-1} \\ g^*_{i,H_u} \\ g^*_{1,H_u} \\ 0 \\ 0 \\ g^*_{4,H_u} \\ 0 \end{bmatrix} \tag{6.4.1}$$

Where

$$U^*_{k-1} = \begin{bmatrix} g^*_{i,1} \\ g^*_{i,2} \\ g^*_{i,3} \\ \vdots \\ g^*_{i,H_u-2} \\ g^*_{i,H_u-1} \\ g^*_{i,H_u} \end{bmatrix} \tag{6.4.2}$$

and

$$i = 1, 2, \ldots, 5$$

Notice that the inputs g_1 and g_4 are constant while inputs $g_i, i = 2, 3, 5$ are zero at the end of the input horizon.

6.5 Local QP-problem, LTV model

When the LTV model is used in the optimization problem, the cost-function needs to be re-written so that it is quadratic in the inputs. This can be done by applying the superposition principle to the inputs as suggested by *M. J. Maciejowski* [10]

$$g_{k+2} = g_{k-1} + \Delta g_k + \Delta g_{k+1}$$

and separating $X_{pred,k}$ into the free response of the system X_{free} and the response to the change in inputs $\Sigma\Delta U$

$$\begin{aligned}
x_{k+1} =& \Phi_k x_k + \Gamma_k g_k \\
=& \Phi_k x_k + \Gamma_k g_{k-1} + \Gamma_k \Delta g_k \\
x_{k+2} =& \Phi_{k+1} x_{k+1} + \Gamma_{k+1} g_{k+1} \\
=& \Phi_{k+1}\Phi_k x_k + \Phi_{k+1}\Gamma_k g_{k-1} + \Phi_{k+1}\Gamma_k \Delta g_k + \Gamma_{k+1}(g_{k-1} + \Delta g_k + \Delta g_{k+1}) \\
& \vdots \\
\begin{bmatrix} x_{k+1} \\ x_{k+2} \\ \vdots \end{bmatrix} =& X_{free} + \Sigma\Delta U \\
=& \begin{bmatrix} \Phi_k \\ \Phi_{k+1}\Phi_k \\ \vdots \end{bmatrix} x_k + \begin{bmatrix} \Gamma_k \\ \Gamma_{k+1} + \Phi_{k+1}\Gamma_k \\ \vdots \end{bmatrix} g(k-1) \\
& + \begin{bmatrix} \Gamma_k & 0 & \ldots \\ \Gamma_{k+1} + \Phi_{k+1}\Gamma_k & \Gamma_{k+1} & \ldots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \Delta g_k \\ \Delta g_{k+1} \\ \vdots \end{bmatrix}
\end{aligned} \tag{6.5.1}$$

The cost function can now be written as

$$V_k = (\Sigma \Delta U_k + X_{free,k})^T Q(\Sigma \Delta U_k + X_{free,k}) + \Delta U_k^T R \Delta U_k \tag{6.5.2}$$

where

$$x_{free} = \int f(x(t), g_{k-1}) dt \tag{6.5.3}$$

$$X_{free,k} = \begin{bmatrix} x_{free,k} \\ x_{free,k+1} \\ \vdots \\ x_{free,k+H_u} \\ \vdots \\ x_{free,k+H_p-1} \end{bmatrix} \tag{6.5.4}$$

$$\Sigma = \begin{bmatrix} \Gamma_k & 0 & & \dots \\ (\Phi_{k+1}\Gamma_k + \Gamma_{k+1}) & \Gamma_{k+1} & 0 & \dots \\ (\Phi_{k+2}\Phi_{k+1}\Gamma_k + \Phi_{k+2}\Gamma_{k+1} + \Gamma_{k+2}) & (\Phi_{k+2}\Gamma_{k+1} + \Gamma_{k+2}) & \Gamma_{k+2} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{6.5.5}$$

The free response x_{free} is evaluated at the sample times $k+i$ to form the vector $X_{free,k}$. It is important to notice that when calculating the elements of X_{free} in equation 6.5.3, the inputs g_1 and g_4 are kept constant, while when calculating the elements of X_{pred} the inputs are not constant. Thus, since g_2, g_3 and g_5 are the derivatives of g_1 and g_4 respectively, the derivatives g_2, g_3 and g_5 have to be zero when X_{free} is calculated.

When the desired surge velocity is time variant the desired surge velocity on the path has to be included. This can be done by

$$E_k = T_k - X_{free,k}$$
$$V_k = (\Sigma \Delta U_k - E_k)^T Q(\Sigma \Delta U_k - E_k) + \Delta U_k^T R \Delta U_k$$

where

$$T_k = \begin{bmatrix} 0 & \dots & u_{dp} & 0 & \dots & u_{dp} & 0 & \dots \end{bmatrix}^T$$

For implementation purposes it is practical to describe Σ in terms of S where S is

$$S = \begin{bmatrix} \Gamma_k & 0 & \dots & 0 \\ \Phi_{k+1}\Gamma_k & \Gamma_{k+1} & \dots & 0 \\ \Phi_{k+2}\Phi_{k+1}\Gamma_k & \Phi_{k+2}\Gamma_{k+1} & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ (\prod_{i=1}^{H_u-1} \Phi_{k+i})\Gamma_k & (\prod_{i=2}^{H_u-1} \Phi_{k+i})\Gamma_{k+1} & \dots & \Gamma_{k+H_u-1} \\ \vdots & \vdots & \dots & \vdots \\ (\prod_{i=1}^{H_p-1} \Phi_{k+i})\Gamma_k & (\prod_{i=2}^{H_p-1} \Phi_{k+i})\Gamma_{k+1} & \dots & (\prod_{i=H_u}^{H_p-1} \Phi_{k+i})\Gamma_{k+H_u-1} \end{bmatrix} \tag{6.5.6}$$

$$\Sigma = \begin{bmatrix} S_{11} & 0 & 0 & \dots & 0 \\ S_{21} + S_{22} & S_{22} & 0 & \dots & 0 \\ S_{31} + S_{32} + S_{33} & S_{32} + S_{33} & S_{33} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ \sum_{j=1}^{H_u} S_{H_u j} & \sum_{j=2}^{H_u} S_{H_u j} & \sum_{j=3}^{H_u} S_{H_u j} & \dots & S_{H_u H_u} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum_{j=1}^{H_u} S_{H_p j} & \sum_{j=2}^{H_u} S_{H_p j} & \sum_{j=3}^{H_u} S_{H_p j} & \dots & S_{H_p H_u} \end{bmatrix} \tag{6.5.7}$$

Notice that the prediction horizon is not infinite, thus the prediction horizon H_p can be found in the matrices 6.5.6 and 6.5.7. Also notice that the input $g_1 = \Delta$ is assumed to be constant in the

interval $[u_{k+H_u-1} \dots u_{k+H_p-1}]$.

Given the matrices Σ and $X_{free,k}$, the cost-function for the local QP problem can be written as

$$V_k = \Delta U^T H \Delta U + G \Delta U \tag{6.5.8}$$

where

$$H = \Sigma^T Q \Sigma \tag{6.5.9}$$

$$G = -2\Sigma^T Q E_k \tag{6.5.10}$$

where the constant term $X_{free,k}^T Q X_{free,k}$ has been dropped since the cost-functions value is not of interest, but rather its minimum. This can be done since the value of g_1 at which the cost-function has its minimum is not dependent on the constant term. Hence, the search for the minimum is not dependent on the constant term.

6.6 Constraints

All constraints have to be written in terms of the change in inputs ΔU. The most important constraints for the problem in this thesis, are the constraints on g. That is, 6.6.1-6.6.7 has to hold for all solution of $g(t)$. The constraint 6.6.2 is posed to avoid that the look-ahead distance becomes too large in the case where the vessel tries to avoid overshoot. This constraint is added because a too large look-ahead distance will reduce the reaction time when the look-ahead distance needs to be reduced again. The value of Δ_{max} can be chosen as best suited for the particular optimization problem.

$$g_1 > \Delta_{min} \tag{6.6.1}$$
$$g_1 < \Delta_{max} \tag{6.6.2}$$
$$g_4 > u_{d,min} \tag{6.6.3}$$
$$g_4 < u_{d,max} \tag{6.6.4}$$
$$g_2(t) = \dot{\Delta}(t) \tag{6.6.5}$$
$$g_3(t) = \ddot{\Delta}(t) \tag{6.6.6}$$
$$g_5(t) = \dot{u}_d(t) \tag{6.6.7}$$

where $\Delta(t)$ is the look-ahead distance, Δ_{min} is a value larger than the smallest value which guarantees $\kappa-$exponential stability when Δ is constant, $u_{d,max}$ and $u_{d,min}$ are the largest and smallest surge velocity that we allow. These constraints will ensure that the solutions $g_k = g_{1,k}, g_{2,k}, \ldots, g_{5,k}$ are chosen such that $g_1(t), g_4(t)$ are free variables, while $g_2(t)$ and $g_3(t)$ are equal to the derivatives of $g_1(t)$, and $g_5(t)$ is equal to the derivative of $g_4(t)$. Since the look-ahead distance is time variant, it is not known whether respecting the bound on the look-ahead distance for the case where the look-ahead distance is constant will guarantee stability when the look-ahead distance is time variant. However, it is safer to respect this bound than to ignore it.

The equations B.0.6 and B.0.7 in *Appendix B* are used to approximate the derivatives $\dot{\Delta}$, $\ddot{\Delta}$ and $\dot{u}_d$. The constraints 6.6.1-6.6.7 for the discrete case becomes

$$\begin{aligned}
g_{1,k} &> g_{1,min} \\
g_{1,k} &< g_{1,max} \\
g_{4,k} &> g_{4,min} \\
g_{4,k} &< g_{4,max} \\
g_{2,k} &= \frac{\Delta g_{1,k+1}}{T_s} \\
g_{3,k} &= \frac{\Delta g_{2,k+1}}{T_s} \\
g_{5,k} &= \frac{\Delta g_{4,k+1}}{T_s}
\end{aligned}$$

In terms of $\Delta g_{i,k}$ the constraints are

$$\begin{aligned}
-\Delta g_{1,k} &< g_{1,k-1} - g_{1,min} \\
-\Delta g_{1,k} - \Delta g_{1,k+1} &< g_{1,k-1} - g_{1,min} \\
-\Delta g_{1,k} - \Delta g_{1,k+1} - \Delta g_{1,k+2} &< g_{1,k-1} - g_{1,min} \\
&\vdots
\end{aligned}$$

$$
\begin{aligned}
\Delta g_{1,k} &< g_{1,max} - g_{1,k-1} \\
\Delta g_{1,k} + \Delta g_{1,k+1} &< g_{1,max} - g_{1,k-1} \\
\Delta g_{1,k} + \Delta g_{1,k+1} + \Delta g_{1,k+2} &< g_{1,max} - g_{1,k-1} \\
&\vdots \\
-\Delta g_{4,k} &< g_{4,k-1} - g_{4,min} \\
-\Delta g_{4,k} - \Delta g_{4,k+1} &< g_{4,k-1} - g_{4,min} \\
-\Delta g_{4,k} - \Delta g_{4,k+1} - \Delta g_{4,k+2} &< g_{4,k-1} - g_{4,min} \\
&\vdots \\
\Delta g_{4,k} &< g_{4,max} - g_{4,k-1} \\
\Delta g_{4,k} + \Delta g_{4,k+1} &< g_{4,max} - g_{4,k-1} \\
\Delta g_{4,k} + \Delta g_{4,k+1} + \Delta g_{4,k+2} &< g_{4,max} - g_{4,k-1} \\
&\vdots \\
\Delta g_{1,k} &= T_s g_{2,k-1} \\
\Delta g_{2,k} - \frac{\Delta g_{1,k+1}}{T_s} &= -g_{2,k-1} \\
\Delta g_{2,k} + \Delta g_{2,k+1} - \frac{\Delta g_{1,k+2}}{T_s} &= -g_{2,k-1} \\
\Delta g_{2,k} + \Delta g_{2,k+1} + \Delta g_{2,k+2} - \frac{\Delta g_{1,k+3}}{T_s} &= -g_{2,k-1} \\
&\vdots \\
\Delta g_{2,k} &= T_s g_{3,k-1} \\
\Delta g_{3,k} - \frac{\Delta g_{2,k+1}}{T_s} &= -g_{3,k-1} \\
\Delta g3,k + \Delta g_{3,k+1} - \frac{\Delta g_{2,k+2}}{T_s} &= -g_{3,k-1} \\
\Delta g_{3,k} + \Delta g_{3,k+1} + \Delta g_{3,k+2} - \frac{\Delta g_{2,k+3}}{T_s} &= -u_{3,k-1} \\
&\vdots \\
\Delta g_{4,k} &= T_s g_{5,k-1} \\
\Delta g_{5,k} - \frac{\Delta g_{4,k+1}}{T_s} &= -g_{5,k-1} \\
\Delta g_{5,k} + \Delta g_{5,k+1} - \frac{\Delta g_{4,k+2}}{T_s} &= -g_{5,k-1} \\
\Delta g_{5,k} + \Delta g_{5,k+1} + \Delta g_{5,k+2} - \frac{\Delta g_{4,k+3}}{T_s} &= -g_{5,k-1} \\
&\vdots
\end{aligned}
$$

where $g_{i,k-1}, i = 1, 2, 3$ were the inputs applied to the system at the previous time step and are thus known, while $g_{i,k}, i = 1, 2, 3$ are the inputs to be applied when the algorithm terminates. Recall that $\Delta g_{i,k} = g_{i,k} - g_{i,k-1}$.

When the constraints are collected in matrix form they can be written on the standard form used

e.g by the embedded *Matlab* function *quadprog*

$$F_c \Delta U \leq f_1 \tag{6.6.8}$$

$$A_{eq} \Delta U = b_{eq} \tag{6.6.9}$$

$$dim(\Delta U) = 3H_u \times 1$$

where

$$F_c = \begin{bmatrix} \hat{I}_1 & 0_{1\times5} & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_1 & \hat{I}_1 & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_1 & \hat{I}_1 & \hat{I}_1 & \dots & 0_{1\times5} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \hat{I}_1 & \hat{I}_1 & \hat{I}_1 & \dots & \hat{I}_1 \\ -\hat{I}_1 & 0_{1\times5} & 0_{1\times5} & \dots & 0_{1\times5} \\ -\hat{I}_1 & -\hat{I}_1 & 0_{1\times5} & \dots & 0_{1\times5} \\ -\hat{I}_1 & -\hat{I}_1 & -\hat{I}_1 & \dots & 0_{1\times5} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\hat{I}_1 & -\hat{I}_1 & -\hat{I}_1 & \dots & -\hat{I}_1 \\ \hat{I}_2 & 0_{1\times5} & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_2 & \hat{I}_2 & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_2 & \hat{I}_2 & \hat{I}_2 & \dots & 0_{1\times5} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \hat{I}_2 & \hat{I}_2 & \hat{I}_2 & \dots & \hat{I}_2 \\ -\hat{I}_2 & 0_{1\times5} & 0_{1\times5} & \dots & 0_{1\times5} \\ -\hat{I}_2 & -\hat{I}_2 & 0_{1\times5} & \dots & 0_{1\times5} \\ -\hat{I}_2 & -\hat{I}_2 & -\hat{I}_2 & \dots & 0_{1\times5} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\hat{I}_2 & -\hat{I}_2 & -\hat{I}_2 & \dots & -\hat{I}_2 \end{bmatrix}, f_1 = \begin{bmatrix} g_{1,k-1} - g_{1,min} \\ g_{1,k-1} - g_{1,min} \\ \vdots \\ g_{1,k-1} - g_{1,min} \\ g_{1,max} - g_{1,k-1} \\ g_{1,max} - g_{1,k-1} \\ \vdots \\ g_{1,max} - g_{1,k-1} \\ g_{4,k-1} - g_{4,min} \\ g_{4,k-1} - g_{4,min} \\ \vdots \\ g_{4,k-1} - g_{4,min} \\ g_{4,max} - g_{4,k-1} \\ g_{4,max} - g_{4,k-1} \\ \vdots \\ g_{4,max} - g_{4,k-1} \end{bmatrix} \tag{6.6.10}$$

$$dim(F_c) = H_u \times 5H_u, dim(f_1) = H_u \times 1$$

$$A_{eq} = \begin{bmatrix} -\hat{I}_1 & 0_{1\times5} & 0_{1\times5} & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_3 & 0_{1\times5} & 0_{1\times5} & 0_{1\times5} & \dots & 0_{1\times5} \\ -\hat{I}_2 & 0_{1\times5} & 0_{1\times5} & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_3 & \hat{T}_1 & 0_{1\times5} & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_4 & \hat{T}_2 & 0_{1\times5} & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_5 & \hat{T}_3 & 0_{1\times5} & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_3 & \hat{I}_3 & \hat{T}_1 & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_4 & \hat{I}_4 & \hat{T}_2 & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_5 & \hat{I}_5 & \hat{T}_3 & 0_{1\times5} & \dots & 0_{1\times5} \\ \hat{I}_3 & \hat{I}_3 & \hat{I}_3 & \hat{T}_1 & \dots & 0_{1\times5} \\ \hat{I}_4 & \hat{I}_4 & \hat{I}_4 & \hat{T}_2 & \dots & 0_{1\times5} \\ \hat{I}_5 & \hat{I}_5 & \hat{I}_5 & \hat{T}_3 & \dots & 0_{1\times5} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \\ \hat{I}_3 & \hat{I}_3 & \hat{I}_3 & \hat{I}_3 & \dots & \hat{T}_1 \\ \hat{I}_4 & \hat{I}_4 & \hat{I}_4 & \hat{I}_4 & \dots & \hat{T}_2 \\ \hat{I}_5 & \hat{I}_5 & \hat{I}_5 & \hat{I}_5 & \dots & \hat{T}_3 \end{bmatrix}, b_{eq} = \begin{bmatrix} T_s g_{2,k-1} \\ T_s g_{3,k-1} \\ T_s g_{5,k-1} \\ -g_{2,k-1} \\ -g_{3,k-1} \\ -g_{5,k-1} \\ -g_{2,k-1} \\ -g_{3,k-1} \\ -g_{5,k-1} \\ \vdots \\ -g_{2,k-1} \\ -g_{3,k-1} \\ -g_{5,k-1} \end{bmatrix} \tag{6.6.11}$$

$$dim(A_{eq}) = 3H_u \times 5H_u, dim(b_{eq}) = 2H_u \times 1$$

$$\hat{I}_1 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.6.12}$$
$$\hat{I}_2 = \begin{bmatrix} 0 & 0 & 0 & -1 & 0 \end{bmatrix} \tag{6.6.13}$$
$$\hat{I}_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{6.6.14}$$
$$\hat{I}_4 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{6.6.15}$$
$$\hat{I}_5 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{6.6.16}$$
$$\hat{T}_1 = \begin{bmatrix} \frac{-1}{T_s} & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.6.17}$$
$$\hat{T}_2 = \begin{bmatrix} 0 & \frac{-1}{T_s} & 0 & 0 & 0 \end{bmatrix} \tag{6.6.18}$$
$$\hat{T}_3 = \begin{bmatrix} 0 & 0 & 0 & \frac{-1}{T_s} & 0 \end{bmatrix} \tag{6.6.19}$$
$$0_{1\times 5} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.6.20}$$

If it is desired to add constraints on some of the states, this can be done by including

$$G_c \Sigma \Delta U \leq l_1 \tag{6.6.21}$$

and expressing the constraints on the states by appropriate choices of G_c and l_1. Such constraints are not included in this report.

6.7 Nonlinear solver

The nonlinear *Matlab* function *fmincon*, which according to the Matlab helpdesk[3]finds the minimum of a constrained nonlinear multi-variable function, will be used to solve the nonlinear optimization problem.

$$\sum (C_z X)^T Q C_z X + U^T Q_u U \tag{6.7.1}$$

subject to

$$g_1 > \Delta_{min} \tag{6.7.2}$$
$$g_1 < \Delta_{max} \tag{6.7.3}$$
$$g_4 > u_{d,min} \tag{6.7.4}$$
$$g_4 < u_{d,max} \tag{6.7.5}$$
$$g_2(t) = \dot{\Delta}(t) \tag{6.7.6}$$
$$g_3(t) = \ddot{\Delta}(t) \tag{6.7.7}$$
$$g_5(t) = \dot{u}_d(t) \tag{6.7.8}$$

where C_z is chosen such that $C_z X$ is a vector containing only the predicted states which are to be weighed in the cost function. The matrix Q contains the weights on the states and the matrix Q_u contains the weights on the inputs. The equality constraints are implemented by using the estimates in *Appendix B*. The calculation time of the optimization problem can to some extent be reduced by writing the cost function as a scalar equation when the cost-function is implemented. This is because matrix multiplication has more over-head than scalar multiplication.

The tool *fmincon* finds the minimum of cost-functions for nonlinear systems. It can handle both linear and nonlinear constraints. The advantage of using such a function instead of QP-solvers such as *quadprog* is that the system does not have to be linearized, which results in better accuracy of the solution. The drawback is that convexity can no longer be guaranteed, and that the calculation time may increase. However, if the initial guess of the solution is close enough to the solution, the tool will find the global minimum. Further, if the solutions at time k are close to the solutions at time k+1, the calculation time will not be significantly larger than for a QP-solver.

The initial guess in this report has been chosen to be the previously found solutions, shifted once in

[3]Type "helpdesk" in the Matlab command window and search for fmincon, for more details on this function.

time. This will result in short calculation time if the previously found solutions were good predictions of the current solutions. One way to make an guess of the initial solution which is close enough to the global minimum, is to impose some rule based on the knowledge of the system. Such a rule will be imposed in the chapter *Results* as it turns out that the non-linear optimization problem is not convex in some cases, if not in all cases.

The *Matlab* tool *fmincon* evaluates the cost-function and the constraints to form the gradients and the Hessian. These are used to form a search direction and to determine whether the solution has been found, and to ensure that the solution is in the feasible region or if there is a feasible solution at all.

7 Problem Definition

This report aims to define an optimization problem which, when used in a MPC context, results in a look-ahead distance $\Delta(t)$ which causes the cross-track error to converge as fast as possible, with as little over-shoot as possible. Thus, a cost-function which has its minimum close to this desired behavior needs to be formulated.

7.1 Desired behavior

Since it is desired that the cross-track error converges fast, the look-ahead distance should be small when the cross-track error is large. However, since the cross-track error should have small over-shoot, the look-ahead distance has to be large. Thus, if the cross-track error is large at time t_0, the initial look-ahead distance should be small, while as the cross-track error reduces, the look-ahead distance should increase to avoid over-shoot in the cross-track error.

By having the look-ahead distance starting small and increasing as the cross-track error reduces, both the fast reduction property of a small constant look-ahead distance and the small over-shoot property of the large constant look-ahead distance can be obtained. However, if the look-ahead distance stays large when the cross-track error is small, a prediction error due to disturbances or modeling errors (e.g $\Delta(t)$ starts to increase too soon or too late), will slow down the final approach towards the path. A too large value of the look-ahead distance when the vessel is subject to a current will cause problems even if the predictions are perfect. This is because a large look-ahead distance causes only small changes in the desired heading for a small increasing cross-track error.

Figure 12 illustrates the case where $\Delta(t)$ starts to increase to soon and keeps increasing. The cross-track error reduces fast, over-shoots, and the convergence rate of the cross-track error becomes very small. The same problem would arise if $\Delta(t)$ converged to a constant large value, or if the look-ahead distance was large and a current was effecting the vessel. Figure 13 illustrates the case where $\Delta(t)$ starts to increase to soon, but decreases as the cross-track error over-shoots. The cross-track error converges fast, though there is a slight over-shoot. Figure 14 illustrates the ideal case where the cross-track error converges fast with no over-shoot. The results shown in Figures 12-14 motivates for formulating a optimization problem which results in a look-ahead distance similar to that of Figure 14.

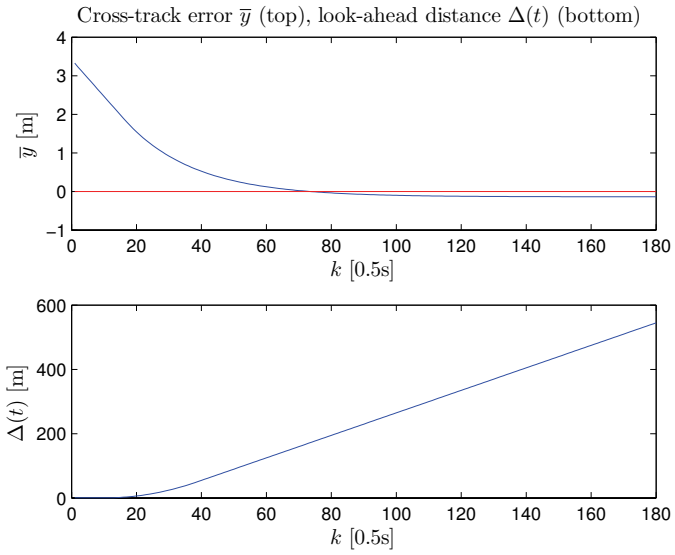

Figure 12: From simulation where $\Delta(t)$ starts increasing to soon and keeps increasing.

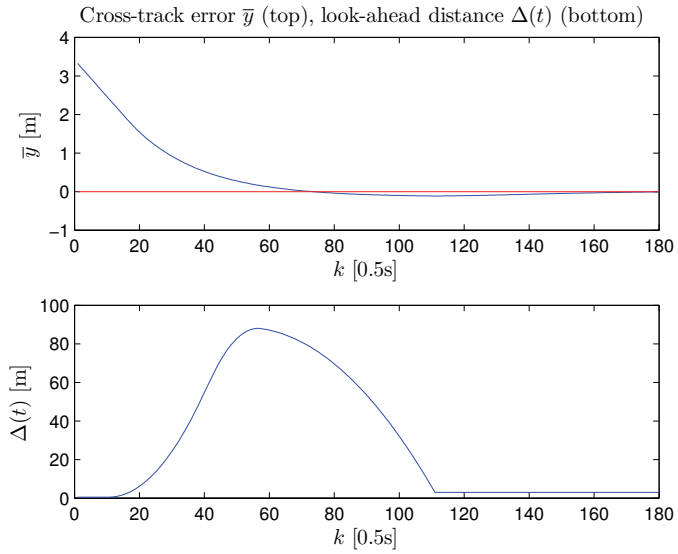

Figure 13: From simulation where $\Delta(t)$ increase to avoid over-shoot and decreases to ensure convergence.

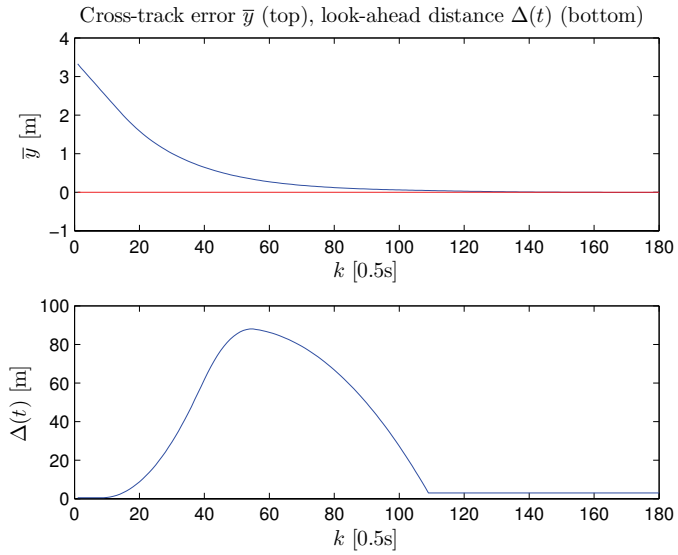

Figure 14: From simulation where $\Delta(t)$ starts to increase at time t_c to avoid over-shoot and decreases again to converge.

The vessel is simulated for one large and one small constant look-ahead distance, and for the time varying look-ahead distance from Figure 14, in order to get an idea of how to form a cost-function which has its minimum close to the desired behavior.

The states which can be used to form the cost-function are measured and investigated for the three cases. The goal is to find states with some properties which can be exploited when the cost-function is to be formulated. That is, if one or more states deviates less from zero for the case of desired behavior ($\Delta = \Delta(t)$), than for the cases of constant Δ, these states would be candidates for being included in the cost-function. Also, if a combination of states would result in a smaller cost for the case of desired behavior than for the other two cases, this can be exploited.

Note: *All candidate states should converge to zero since the reference frame is rotated such that the x axis is aligned with the path, the y coordinate equals the cross-track error and the rotated heading angle is zero when it equals the angle of the path.*

Note: *Since the main goal is to minimize the cross-track error, the cross-track error should appear in the cost function.*

Since the cost of performing the optimization, in terms of calculation time, increases with the prediction horizon and the input horizon, it is not desirable to have too long horizons. Thus the candidate states should have one of the following properties to be included in the cost-function

Property 1:

the sum over the prediction horizon of the squared state measurements should be less for a state measured from the simulated desired behavior, than for the same state measured from either one of the two simulations where the look-ahead distance is constant.

Property 2:

the sum over the prediction horizon of the squared state measurements should be less for two or more states measured from the simulated desired behavior, than for the same states measured from either one of the two simulations where the look-ahead distance is constant.

Expressed mathematically *Property 1 and 2* becomes

Property 1:

$$\begin{aligned}\sum_{i=k}^{k+H_p} k_x x_d^2(i) &< \sum_{i=k}^{k+H_p} k_x x_{c1}^2(i),\\ \sum_{i=k}^{k+H_p} k_x x_d^2(i) &< \sum_{i=k}^{k+H_p} k_x x_{c2}^2(i)\end{aligned} \tag{7.1.1}$$

Property 2:

$$\begin{aligned}\sum_{i=k}^{k+H_p} k_y y_d^2(i) + k_x x_d^2(i) &< \sum_{i=k}^{k+H_p} k_y y_{c1}^2(i) + k_x x_{c1}^2(i),\\ \sum_{i=k}^{k+H_p} k_y y_d^2(i) + k_x x_d^2(i) &< \sum_{i=k}^{k+H_p} k_y y_{c1}^2(i) + k_x x_{c1}^2(i)\end{aligned} \tag{7.1.2}$$

where i indicates sample i, k_j is the weight on state j, x_d and y_d are some state measurements measured from the case of desired behavior, while $x_{c1,c2}$ and $y_{c1,c2}$ are the same state measurements measured from the case of constant, large (c1) or small (c2), look-ahead distance.

Note: *In 7.1.2, Property 2 has been expressed for two states. If more than two states satisfies*

Property 2, or if property 2 is satisfied only for a combination of more than two states, the sum should include the square of all state measurements that satisfies Property 2.

This analysis is performed to get an idea of which states can be used in formulating the cost-function. However, even though the simulations may indicate that either *Property* 1 or 2 is satisfied for some states, this does not necessarily mean that a cost-function based on these states will always lead to desired behavior. This is because this analysis only considers simulations with two constant values for the look-ahead distance and one simulation for a time varying look-ahead distance, where only one set of initial conditions are used. There may be other time varying look-ahead distances which does not result in desired behavior, but satisfies *Property* 1 or 2 and results in a lower value of the cost-function. Further, states satisfying *Property* 1 or 2 under the initial conditions used in this analysis, may not satisfy *Property* 1 or 2 for other initial conditions. Finally, this analysis is performed assuming no environmental disturbances.

Each resulting cost function will have to be implemented and simulated. These simulations will have to be analyzed and compared to find which cost-function is best suited to describe the desired behavior.

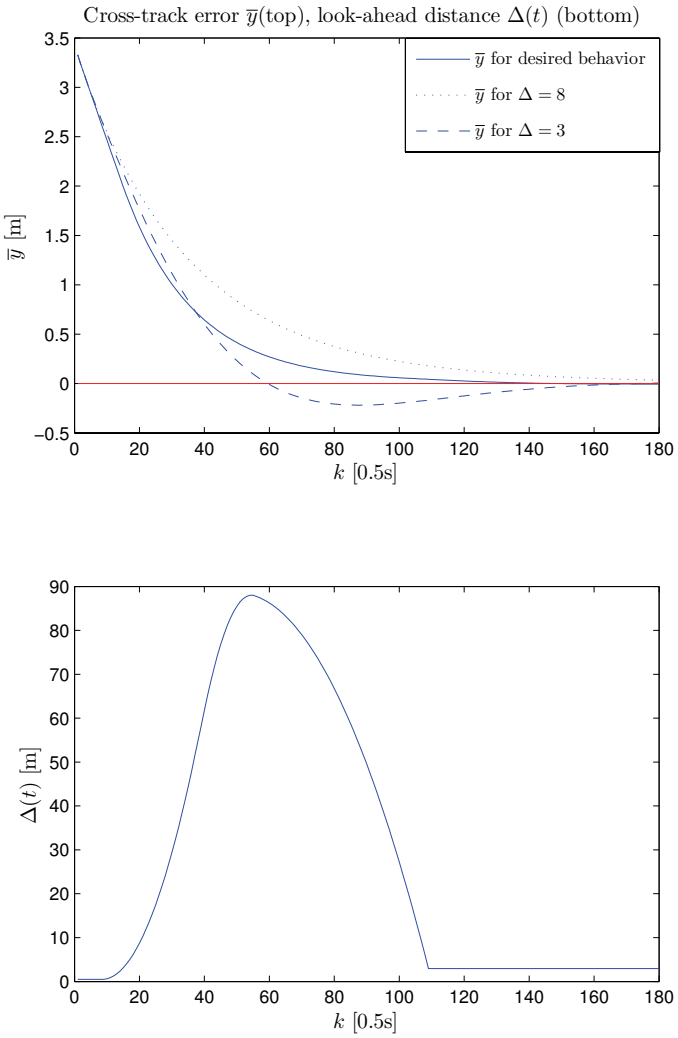

Figure 15: Cross-track error for the three cases and $\Delta(t)$.

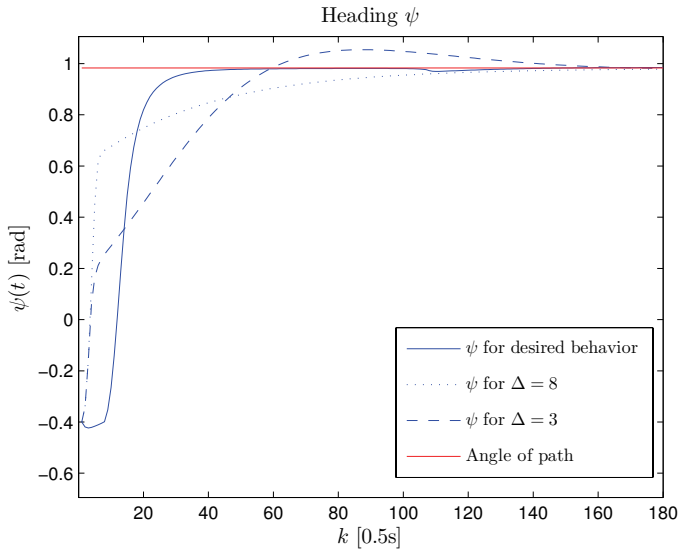

Figure 16: Heading for the three cases.

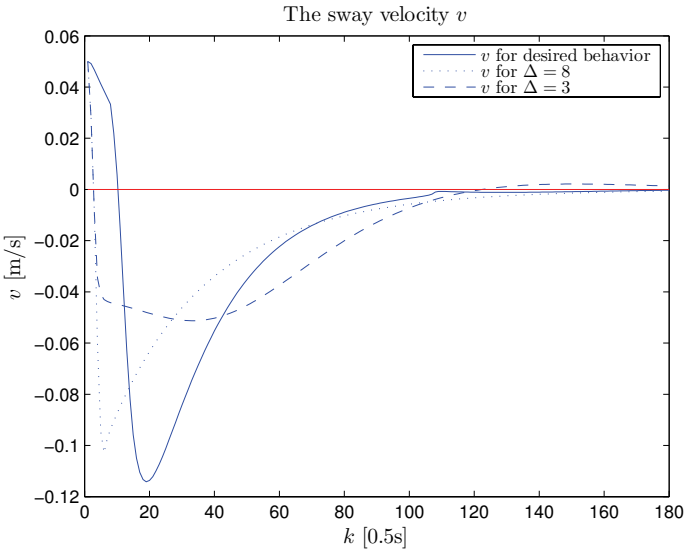

Figure 17: Sway velocity for the three cases.

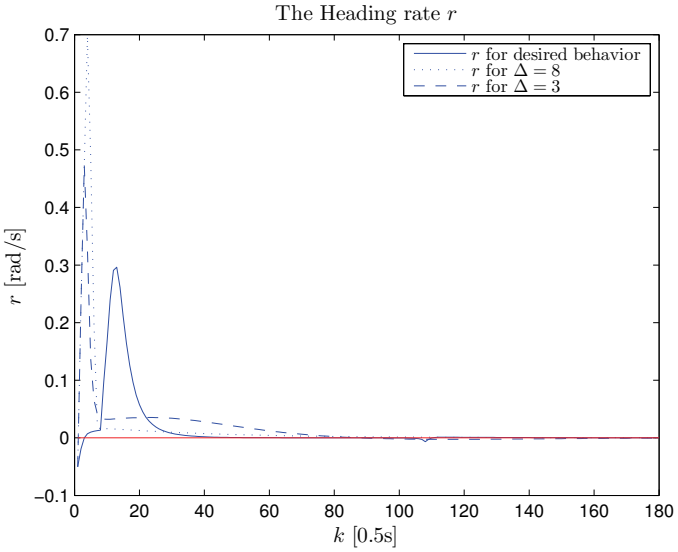

Figure 18: Heading rate for the three cases.

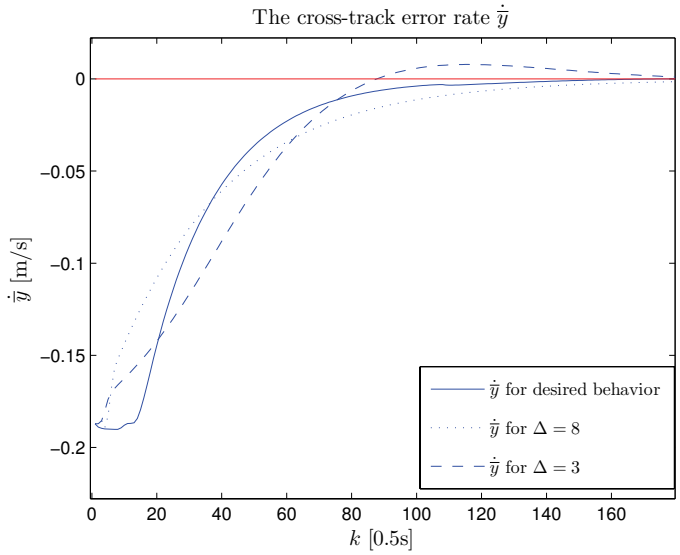

Figure 19: Cross-track error rate for the three cases.

7.2 Constant surge velocity

The optimization problems in this report will be divided into two main groups. The first group minimizes the cost-function with respect to the look-ahead distance where the surge velocity tracks a constant reference, while the second group minimizes the cost-functions with respect to the look-ahead distance and the desired surge velocity. The desired surge velocity is constant in the suggested cost-functions 1-3 .

7.2.1 Cost-function 1

It has already been stated that the cross-track error needs to be included in the cost-function. Figures 15-16 indicates that the cross-track error and the heading may satisfies *Property 2* for appropriate values of k_y and k_ψ. The cross-track error is smaller for the desired behavior than for both of the two constant look-ahead distances up to time $k \approx 35$ and from time $k \approx 75$, while the heading deviates less from the angle of the path for the desired behavior after time $k \approx 20$. Also notice that the heading rate in Figure 18 is smaller from time $k \approx 35$ for the desired behavior than for the constant look-ahead distances. This indicates that the heading rate combined with the cross-track error and the heading, may also satisfy *Property 2* for appropriate values of k_y, k_ψ and k_r. This motivates for the following cost function

$$\sum_{i=k}^{k+H_p} k_y y_i^2 + k_\psi \psi_i^2 + k_r r_i^2 \tag{7.2.1}$$

A too large weight k_ψ or k_r would cause minimization of the heading angle or the heading rate respectively, rather than the cross-track error. The product $k_y y^2$ should be larger than $k_\psi \psi^2$ up to some time before $k \approx 20$ since the contribution of the heading to the cost-function for desirable behavior is less than for non-desirable behavior only after $k \approx 20$. This is no guarantee of achieving desired behavior, but the heading should not contribute significantly to the cost function before the cross-track error is small since the contribution from $k_\psi \psi^2$ to the cost function is less for desirable behavior, than for undesirable behavior, only when the cross-track error is small.

The desirable behavior requires an increase in the look-ahead distance when the cross-track error is small and the cross-track error rate is large to avoid or reduce over-shoot. The effect of the increased look-ahead distance is that the cross-track error rate is reduced by rotating the vessel so that the difference between the angle of the path and the heading angle reduces. That is, the vessel rotates so that the velocity towards the path is mainly in the sway direction. Since the heading angle approaches the angle of the path when the cross-track error reduces and the look-ahead distance increases, it should be possible to find weights which causes the look-ahead distance to increase so that the cross-track error has only a small over-shoot. However, when the cross-track error and the cross-track error rate is small, it is desired to have the look-ahead distance reduce. Hence, the weight k_ψ must be large enough to cause an increase in the look-ahead distance when the cross-track error otherwise would over-shoot, but at the same time small enough so that the look-ahead distance will decrease when the cross-track error is small and reducing slowly. Thus, there might be a trade off between fast convergence and small-over shoot in the cross-track error.

The heading rate is smaller for the desired behavior than for un-desirable behavior only after $k \approx 35$. Hence, the weight k_r should be chosen such that $k_r r^2$ is small relative to $k_y y^2$ and $k_\psi \psi^2$ until the cross-track error and the heading is small. The effect of adding $k_r r^2$ to the cost-function is that fast changes in the heading will be more costly than slow changes. This can be exploited to reduce any oscillations in the heading. However, the desired behavior requires fast reduction of the heading to avoid or reduce over-shoot in the cross-track error. Thus, there might be a trade off between oscillations in the heading and the magnitude of the over-shoot in the cross-track error.

The heading reference is non-zero as long as the cross-track error is non-zero since the heading reference is set by the LOS-algorithm. The only exception is if the look-ahead distance diverges to infinity, which results in the heading reference approaching zero, regardless of the value of the cross-track error. Since the cost-function 7.2.1 includes the heading, it may occur that the heading will dominate the cost and cause the look-ahead distance to increase or stay large when the cross-track

error is small. It is desired that the look-ahead distance increases in the case where the cross-track error rate is large, but not when the cross-track error rate is small. This makes it hard to predict whether it is possible to find weights for cost-function 7.2.1 which achieves both fast convergence and small over-shoot in the cross-track error. Thus, another cost-function should be formulated, in case such weights can not be found.

7.2.2 Cost function 2

The cross track error is smaller for the case of desired behavior than for both un-desirable behaviors, except for the interval $k \approx 35 \rightarrow k \approx 75$. Thus, a state which is smaller in this interval for the case of desired behavior than for the un-desirable behaviors, needs to be found. Figure 19 indicates that the rate of the cross-track error has this property. The cross-track error rate ($\dot{y}$) in the case of desired behavior becomes smaller than that of the un-desirable behavior at time $k \approx 35$. At time $k \approx 75$, $\dot{y}$ becomes larger than that of the small constant look-ahead distance, but at time $k \approx 95$ $\dot{y}$ for the desired behavior is again the smallest. The interval $k \approx 75 \rightarrow k \approx 95$ is short, and the magnitude of $\dot{y}$ in this interval is small. If the prediction horizon is large enough, the sum of the squared samples of $\dot{y}$ for the desired behavior, will be smaller than the corresponding sum for either of the two un-desirable behaviors, for $k \geq k_c$ where k_c is some time before $k = 35$. The critical time k_c depends on the length of the horizon. This corresponds to every area between the curve of $\dot{y}$ and zero, starting at time k with length H_p, being smaller in the case of desired behavior than in the case of un-desirable behavior, for $k \geq k_c$ where k_c is dependent on H_p and H_p is large enough.

Based on this discussion, the following cost function is suggested

$$\sum_{i=k}^{k+H_p} k_y y_i^2 + k_{\dot{y}} \dot{y}_i^2 \tag{7.2.2}$$

The cost function suggested in 7.2.2 may seem counterintuitive. It has been stated that it is desired that the cross-track error should be reduced fast, so putting a weight on the cross-track error rate seems to work against this desire. However, it should be noted that the magnitude of the cross-track error rate is small relative to the magnitude of the cross-track error. Thus it may be possible to achieve a cost-function where a large cross-track error dominates the cost-function while a small cross track error results in the cross-track error rate dominating, by setting proper weights. That is, if the cross-track error is large, it will be reduced fast and as the cross-track error becomes small, the cross-track error rate will be reduced, which should reduce the magnitude of the over-shoot.

7.2.3 Cost function 3

The curve of the cross-track error in Figure 14 has the shape of an exponentially converging function, i.e. $x(t) = x_0 e^{-kt}$. Exponential convergence is a desirable property and is guaranteed when the control laws presented in *E. Fredriksen and K.Y Pettersen* [2] are used in the case of a constant look-ahead distance. However, it is not known if exponential convergence will follow for all time variant look-ahead distances. If the cost-function is defined so that it has its minimum when the cross-track error converges exponentially, the solution to the optimization problem should result in exponential convergence of the cross-track error. In particular, if the cost-function is defined so that it has its minimum where the cross-track error tracks an exponentially converging function, it should be possible to achieve fast convergence with small over-shoot. Such a cost-function can be achieved by weighing the deviation from a simple exponential curve.

Since a state defined by

$$\dot{x} = -kx$$
$$k > 0$$

has the solution

$$x(t) = x_0 e^{-kt}$$

the deviation from exponential convergence can be expressed as

$$d = \dot{x} + kx$$

If the cost function is defined as the sum of the squared deviations d, it will have its minimum in $d = 0$, or as close to $d = 0$ as the vessel dynamics allows. Thus, the cost function becomes

$$\sum_{i=k}^{k+H_p} (\dot{y}_i + k_y y_i)^2 \tag{7.2.3}$$

The cost function 7.2.3 will have its minimum for input values which results in behavior close to the desired behavior defined by k_y. Thus, the solution to the optimization problem should result in exponential convergence of the cross-track error. Increased value of k_y should result in increased convergence rate of the cross-track error. However, since all the states of the vessel are defined by their respective dynamics and constraints, all exponential curves defined by k_y can not possibly be followed. Thus, there is an upper limit for k_y, past this limit, the solution to the optimization problem can not be expected to result in desirable behavior. Further, it might be that there exists no exponential curve defined by k_y that can be followed exactly. However, if k_y is chosen so that the curve defined by k_y can be followed with small deviations, the cost function 7.2.3 should result in desired behavior.

When the LTV model is used for predictions, the cost function has to be in the form $x^T Q x$, where Q usually is diagonal. To implement the cost-function in equation 7.2.3, the weight matrix has to be

$$Q = K \begin{bmatrix} k_y^2 & 0 & 0 & 0 & 0 & 0 & 2k_y \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.2.4}$$

where K is a constant scalar grater than zero. The weight Q has to be on the form 7.2.4 because

$$(\dot{x} + k_y x)^2 = \dot{x}^2 + 2k_x x + k_x^2 x^2 = \begin{bmatrix} x & \dot{x} \end{bmatrix} \begin{bmatrix} k_x^2 & 2k_x \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \tag{7.2.5}$$

7.3 Time varying surge velocity

The cost-functions suggested so far all considers the look-ahead distance and its first and second derivatives as inputs, while the surge velocity is assumed to be constant. That is, perfect control of the surge velocity is assumed and the surge velocity reference is assumed constant. This assumption can be relaxed if the surge velocity reference is allowed to be time varying, which will allow the MPC algorithm to consider the surge velocity reference as an input. This extension of the optimization problem is done to allow the vessel to increase its velocity when it is far from the path, and decrease the velocity when it is close to the path.

Since this report considers optimal path following, and not the problem of generating the path and the velocity reference trajectory, it will be assumed that the desired surge velocity on the path is set by some external source and that it is constant. However, the MPC algorithm will be allowed to change the desired surge velocity when this will reduce the value of the cost-function.

7.3.1 Cost function 4-6

Relaxing the assumption of constant surge velocity reference increases the degrees of freedom for the optimization problem. This may increase the calculation time, but it might also reduce the deviation from the desired behavior. The cost-functions 7.2.1-7.2.3 will have to include a term which ensures that the surge velocity tracks the desired surge velocity on the path, set by the external source, when the vessel is on the path. This can be done by adding the following term to cost-functions 7.2.1-7.2.3

$$\sum_{i=k}^{k+H_p} k_{\overline{u}_d}\overline{u}_{d,i}^2 + k_{\dot{u}_d}\dot{u}_{d,i}^2 \tag{7.3.1}$$

where the last term is included to reduce rapid changes in the desired surge velocity u_d, and $\overline{u}_{d,i} = u_i - u_{dp}$. Where u_{dp} is the desired surge velocity on the path set by the external source and u_i is the surge velocity of the vessel at time i. The idea is that the MPC algorithm can change the surge velocity reference to reduce the deviation from the desired behavior, but at the cost of increasing the cost-function value. This will ensure that the surge velocity will not deviate too much from the desired surge velocity on the path.

The resulting three cost functions are then

$$\sum_{i=k}^{k+H_p} k_y y_i^2 + k_\psi \psi_i^2 + k_r r_i^2 + k_{\overline{u}_d}\overline{u}_{d,i}^2 + k_{\dot{u}_d}\dot{u}_{d,i}^2 \tag{7.3.2}$$

$$\sum_{i=k}^{k+H_p} k_y y_i^2 + k_{\dot{y}}\dot{y}_i^2 + k_{\overline{u}_d}\overline{u}_{d,i}^2 + k_{\dot{u}_d}\dot{u}_{d,i}^2 \tag{7.3.3}$$

$$\sum_{i=k}^{k+H_p} (\dot{y}_i + k_y y_i)^2 + k_{\overline{u}_d}\overline{u}_{d,i}^2 + k_{\dot{u}_d}\dot{u}_{d,i}^2 \tag{7.3.4}$$

When the LTV model is used the cost-function V(x) is on the form

$$V(x) = \sum_{i=0}^{H_p} x(k+i|k)^T Q x(k+i|k) + \sum_{i=0}^{H_u} \Delta U^T R \Delta U + \sum_{i=0}^{H_u} U^T S U \tag{7.3.5}$$

$$x_c = \begin{bmatrix} y \\ \psi \\ r \end{bmatrix} \tag{7.3.6}$$

where Q and R are scaling matrices with appropriate dimensions, x_c are the states that are to be controlled, and U contains the inputs.

8 Simulation Results

The six cost-functions suggested in the chapter *Problem Definition* will be implemented and used in simulations where both solvers are used, with the exception of *cost-function 4* which is not presented due to poor performance. The goals are to find which cost-function yields the best performance, investigate whether calculation time can be saved by using the LTV model instead of the nonlinear model, and to reveal possible problems with the different cost-functions.

All simulations are performed on the same vessel model, that is the model of *Cybership2*, with the same initial conditions when the same paths are tracked, in the case of no disturbance and in the case of a constant irrotational current. The simulations will also be performed on different desired paths. The weights will be tuned so that the vessel follows a straight line path, where the cross track error converges faster with less over-shoot than for a constant look-ahead distance. When these weights are found, the path will be extended to consist of several straight lines. The path consisting of several straight lines will be constructed so that it is likely that any constant look-ahead distance will result in either large over-shoot or slow convergence. A problem formulation is considered to be good if the time varying look-ahead distance, and the time varying surge velocity reference in the cases where it is not constant, results in better performance than for constant look-ahead distance and surge velocity reference.

There are many tuning parameters who plays a significant role in achieving good results. The most important relations will be explained for each problem in the section *Tuning rules*. The current which effects the vessel in some simulations is modeled by 4.1.7 using the following parameters

$$
\begin{aligned}
&\textit{current1}\\
&\qquad V_c = 0.01 m/s\\
&\qquad \beta_c = \pi/3\\
&\textit{current2}\\
&\qquad V_c = 0.01 m/s\\
&\qquad \beta_c = -\pi/3
\end{aligned}
$$

The system matrices for the vessel model, the constrains on the control force and moment and the hydrodynamic parameters used for simulations are

$$
\begin{aligned}
M &= \begin{bmatrix} 25.8 & 0 & 0 \\ 0 & 33.8 & 1.0115 \\ 0 & 1.0115 & 2.76 \end{bmatrix}\\
D &= \begin{bmatrix} 0.9257 & 0 & 0 \\ 0 & 2.8909 & -0.2601 \\ 0 & -0.2602 & 0.5 \end{bmatrix}\\
Tau_{u,max} &= 2N\\
Tau_{u,min} &= -2N\\
Tau_{\delta,max} &= 1.5Nm\\
Tau_{\delta,min} &= -1.5Nm\\
N_\delta &= 1\\
y_\delta &= -0.2
\end{aligned}
$$

It has been stated that since it is the cross-track error it is desired to optimize, it needs to be included in the cost-function. It has also been claimed that a cost function that minimizes the squared cross-track error alone, will not necessarily yield desired behavior. This is confirmed in Figure 20, where the LTV model and the cost-function $\sum k_y y_k^2 + \Delta U^T R \Delta U$ has been used. The weights were set to

$$k_y = 3 * 10^4 \tag{8.0.1}$$

$$R = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 3.45 \end{bmatrix} \tag{8.0.2}$$

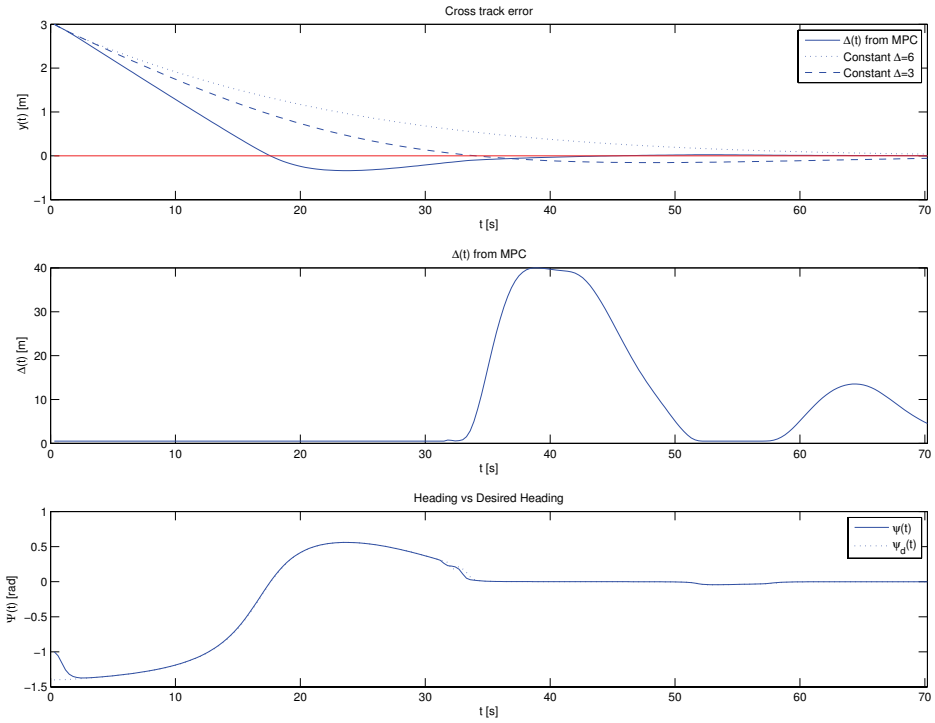

Figure 20: Simulation result for minimizing the squared cross-track error, $H_p = 80$.

where the weighting matrix R is a small weight. The initial states used when the results in section Optimal path-following for a straight line path and the result in Figure 20 were obtained are

$$X = \begin{bmatrix} y & \overline{y} & \psi & v & \overline{v} & r & u & \dot{y} \end{bmatrix}$$
$$X_{init} = \begin{bmatrix} 3.0000 & 2.9613 & -1.0000 & 0.0500 & 0.0477 & -0.0500 & 0.2 & 0 \end{bmatrix}$$

while for the path consisting of several straight lines

$$X_{init} = \begin{bmatrix} 3.0509 & 3.0126 & 0 & 0.0500 & 0.0477 & -0.0500 & 0.2 & 0 \end{bmatrix}$$

8.1 Optimal path-following for a straight line path

Cost functions 1-3 are defined assuming $u = 0.2$, thus the model where the surge velocity is constant is used. Six states are predicted for *cost-function 1* and seven for *cost-function 2 and 3*, while eight states are predicted for the remaining cost-functions. There are weights on the change in input to avoid fast changes, and to avoid that the look-ahead distance grows too large, in all simulations where the LTV model is used.

A problem which may occur for the LTV formulation when the Matlab function *quadprog* is used to solve the optimization problem, is that the constraints are violated by more than ϵ where ϵ is the maximum allowed constraint violation used by *quadprog*. The maximum allowed constraint violation is typically $\epsilon \leq 10^{-15}$. The violations occurs when either the look-ahead distance or the desired surge velocity is close to their maximum or minimum values. It has been suggested by the *Mathworks support* (see web page [15]) that such constraint violations may be caused by rounding errors. The *Mathworks support* also suggests a procedure for increasing ϵ as a workaround. However, it has not been possible to excess the necessary files to do this. Instead the inequality constraints on $U_{k+i},\ i = 1, 2, ...H_u$ has been tighten. That is, the constraints 6.6.1-6.6.4 are changed to

$$\begin{aligned} g_{1,k} &> g_{1,min} \\ g_{1,k+i} &> g_{1,min} + \kappa_{1,i} \end{aligned} \tag{8.1.1}$$

$$\begin{aligned} g_{1,k} &< g_{1,max} \\ g_{1,k+i} &< g_{1,max} - \kappa_{2,i} \end{aligned} \tag{8.1.2}$$

$$\begin{aligned} g_{4,k} &> g_{4,min} \\ g_{4,k+i} &> g_{4min} + \kappa_{3,i} \end{aligned} \tag{8.1.3}$$

$$\begin{aligned} g_{4,k} &< g_{4,max} \\ g_{4,k+i} &< g_{4,max} - \kappa_{4,i} \end{aligned} \tag{8.1.4}$$

where

$$\begin{aligned} &\epsilon \ll \kappa_{1,1} < \kappa_{1,2} ... < \kappa_{1,H_u} \ll g_{1,min} \\ &\epsilon \ll \kappa_{2,1} < \kappa_{2,2} ... < \kappa_{2,H_u} \ll g_{1,max} \\ &\epsilon \ll \kappa_{3,1} < \kappa_{3,2} ... < \kappa_{3,H_u} \ll g_{4,min} \\ &\epsilon \ll \kappa_{4,1} < \kappa_{4,2} ... < \kappa_{4,H_u} \ll g_{4,max} \\ &i = 1, 2, ...H_u \end{aligned}$$

where $g_{1,min} = \Delta_{min} = 0.5$ meters, $g_{4,min} = u_{d,min} = 0.08$ m/s and $g_{4,max} = u_{d,max} = 0.3$ m/s. The maximum allowed value of the look-ahead distance $g_{1,max} = \Delta_{max}$ is set to several different values which will be accounted for.

The constraint violations are avoided when the constraints 8.1.1-8.1.4 are used instead of the constraints 6.6.1-6.6.4. This is because the initial inputs $g_{i,k+i}$ $(i = 0, 1, 2...H_u - 1)$ is set equal to inputs $g_{i,k+1+i}$ from the previous iteration, where the previously found inputs $g_{i,k+1+i}$ satisfies tighter constraints than the inputs $g_{i,k+i}$ has to satisfy.

NOTE *The lower and upper bounds on the change in inputs must be chosen with care, these can cause problems with constraint violations if they are too tight. These bounds should not be necessary, but can be added to avoid too large inputs or too large changes in the inputs. However, a better way to deal with such problems is to increase the input weights.*

8.1.1 Cost function 1

In the simulations of cost-function 1, where the vessel is to track a straight-line path, the following parameters and weights are used

$$
\begin{aligned}
H_p &= 80 \\
H_p &= 180 \\
H_u &= 25 \\
T_s &= 0.3 \\
Q_1 &= \begin{bmatrix} 3*10^4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10^4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\
Q_2 &= \begin{bmatrix} 3*10^4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4*10^4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\
Q_3 &= \begin{bmatrix} 3*10^4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10^4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \\
R_1 &= \begin{bmatrix} 10 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 1000 \end{bmatrix}
\end{aligned}
$$

where H_p is the prediction horizon, H_u is the input horizon, T_s is the time between samples. In most simulations, the simulation time is 70 seconds. The sample time T_s has been chosen to be small enough to give good estimates of the system approximated by

$$x_{k+1} = \Phi_k x_k + \Gamma_k g_k$$

The weighting matrix R is the same in all simulations where *cost-function 1* is used, these values for R are the values found to give the best results for *cost-function 1*. That is, there was not much to gain by changing R for the different weights Q.

The cross-track error in Figure 21 reduces fast, but over-shoots. The look-ahead distance grows large before the over-shoot, which is desirable, but it does not reduce fast enough after the over-shoot. This is a problem, since, as seen in Figure 21, it causes the cross-track error to converge very slow to zero. This is because the numerical value of the cross-track error is small, and reducing the look-ahead distance comes at the cost of increased heading error, and increased cost from the change in inputs.

It is desired that the look-ahead distance should be small when the cross-track error is large and grow when the cross-track error reduces, so that the cross-track error reduces fast, only has a small over-shoot and the heading does not have oscillations. This is not true in Figure 21 since the cross-track error over-shoots, and since the look-ahead distance starts to increase at time $t \approx 5$ seconds, decreases at time $t \approx 8$ seconds and increases again at time $t \approx 14$ seconds. The result is that the heading rotates towards the angle of the path, then away from the angle of the path, and towards the angle of the path again. The source of this undesirable behavior is that the prediction horizon moves along the time-axis, hence the optimal inputs at time $k+i$ found at time k, may not be the same as the optimal inputs found at time $k+1$. Thus, the increase in the look-ahead distance at

time $t \approx 5$ seconds results from the optimizer not being aware of the future beyond $T_s * H_p$. This is confirmed in Figure 22 where the prediction horizon is increased to $H_p = 180$. The look-ahead distance in Figure 22 does not have the increase at time $t \approx 5$ seconds as it has in Figure 21, and the cross-track error in Figure 22 converges fast and does not have over-shoot.

However, the increased prediction horizon results in much longer calculation time of the inputs. This is because the predictions has to be made further into the future, and because the LTV model becomes larger and takes longer to calculate, which again causes the calculation time of the Hessian H to increase. All this leads to increased calculation time, which is a major draw-back since each calculated input needs to be calculated within one sampling interval. This real time requirement is not even satisfied for $H_p = 80$, as in Figure 21, unless the MPC algorithm runs on a powerful computer. The simulations performed on the authors computer (see *Appendix B* for details) with $H_p = 80$ and $T_s = 0.3$ seconds, where the simulation time is 1 minute and 10 seconds, takes approximately 3 minutes to perform.

The weight on the heading can be increased to make the look-ahead distance start increasing earlier, and to reduce the over-shoot. This has been done in Figure 23. The look-ahead distance in Figure 23 converges to zero without over-shoot. However, the look-ahead distance still has a top at time $t \approx 7$ seconds, and the look-ahead distance does not reduce when the cross-track error is small. The latter might be a problem even if the cross-track error converges perfectly to zero since environmental disturbances such as wind, waves and ocean currents can move the vessel from the path. When the look-ahead distance is large (≥ 6 ship lengths or ≈ 7.5 meter), such disturbances will not be effectively suppressed since the heading will stay close to the angle of the path, even when the cross-track error is non-zero. This is seen in Figure 24 where the vessel is subjected to *current1* and the convergence of the cross-track error is slow. This is because the look-ahead distance reduces to slow from time $t \approx 25$ seconds. The heading has oscillations from time $t \approx 70$ which are undesirable. Again, these are caused by the short prediction horizon.

Increasing the prediction horizon and using the weights Q_1 and R_1 when the vessel is subjected to *current1* results in fast convergence of the cross-track error, and less oscillations in the heading than in Figure 24, as seen in Figure 25. However, the heading do have some oscillations after time $t \approx 25$. These oscillations can also be seen in the look-ahead distance in Figure 25. It may be that these can be removed by increasing the prediction horizon further, but this will increase the calculation time even more.

Notice that the simulation time in Figures 24, 26, 28 and 29 is 110 seconds, instead of 70 seconds as in most of the figures in this section.

The cross track error converges to approximately zero for $H_p = 80$, in the presence of current, when the heading rate is weighed in addition to the heading and the cross-track error, see Figure 26. However, the look-ahead distance still has a top at time $t \approx 7$ seconds, the look-ahead distance stays large too long which causes an increase in the cross-track error at time $t \approx 60$ seconds, and the heading has some oscillations after time $t \approx 80$ seconds. The oscillations and the top are again caused by too short prediction horizon. The look-ahead distance does not have the top at time $t \approx 7$ seconds, and the oscillations are reduced when the prediction horizon is increased, as seen in Figure 27. That is, the oscillations in the heading are not completely removed by adding a weight on the heading rate.

The current in Figure 27 is in a direction which damps the over-shoot in the cross-track error. If the current is set to the opposite direction, that is if *current2* is used, the over-shoot in the cross-track error will increase if the look-ahead distance does not become small immediately after the over-shoot. Since including the heading in the cost-function results in increased look-ahead distance when the cross-track error is about to over-shoot, the look-ahead distance is large when the cross-track error over-shoots in Figures 28 and 29. This results in a large over-shoot, that is the cross-track errors from simulations with both a large and a small constant look-ahead distances have smaller over-shoots than the cross-track error for weights Q_1 and Q_3 in Figures 28 and 29. However, the cross-track errors does converge to approximately zero as opposed to the two cases of constant look-ahead distances. The large over-shoot in the cross-track errors in Figures 28 and 29

results from the weight on the heading and the heading rate. If this over-shoot is to be avoided, the heading would have to increase fast at time $t \approx 6$ seconds, as in Figure 28, but it would also have to increase faster at time $t \approx 40$ seconds. In effect, the look-ahead distance will have to reduce fast at time $t \approx 40$ seconds, which results in increased heading rate, increased deviation from zero in the heading, and increased cost from the changes in inputs. Thus, good performance in both the case of *current1* and *current2* is not achievable for either of the weights found in this section. This means that the cost-functions *cost-function 1*, does not perform well for the same weight under different conditions. Which is a major draw-back of *cost-function 1* since it is desirable to have an MPC controller which works well under all conditions, or at the very least most conditions, for a given set of control parameters. Thus, *cost-function 1* will not be investigated further. Since *cost-function 4* is much the same as *cost-function 1*, where the only difference is that the desired surge velocity is time variant, *cost-function 4* will not be investigated in this report.

Note: *simulations where the cross-track error and the heading rate were weighed, but not the heading, have not given any good results. This is because minimizing the cross-track error alone causes a large over-shoot which needs to be compensated for by increasing the look-ahead distance when the cross-track error becomes small. This is achieved when the heading is included in the cost-function, because as the cross-track error reduces, the deviation from zero in the heading becomes significant to the cost-function, which results in the heading being reduced by increasing the look-ahead distance. This is seen by noting that the heading reference approaches zero when the look-ahead distance is increased. However, if the heading rate and the cross-track error is minimized without the heading, the increase in the look-ahead distance to avoid over-shoot is not achieved, because increasing the look-ahead distance is the same as changing the heading, which results in increased cost from the headin rate.*

Note: *The fast changes in the look-ahead distance sometimes causes the desired heading to change faster than the heading can track, i.e. at time $t \approx 15$ seconds in Figure 21. This is a result of saturation in the actuators (τ) which the optimization algorithm is unaware of. The limitations of the actuators should be added to the constraints, especially since the MPC formulation is well suited for such constraint handling. However, this requires expressing τ in terms of the change in inputs and has not been done as part of this thesis.*

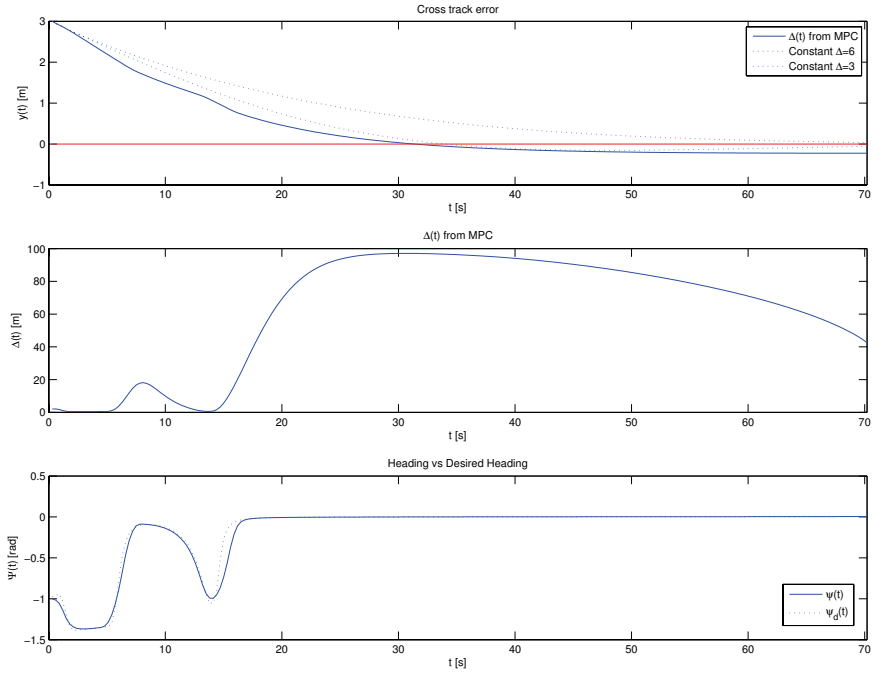

Figure 21: Simulation result for Q_1 and R_1, no current, $H_p = 80$.

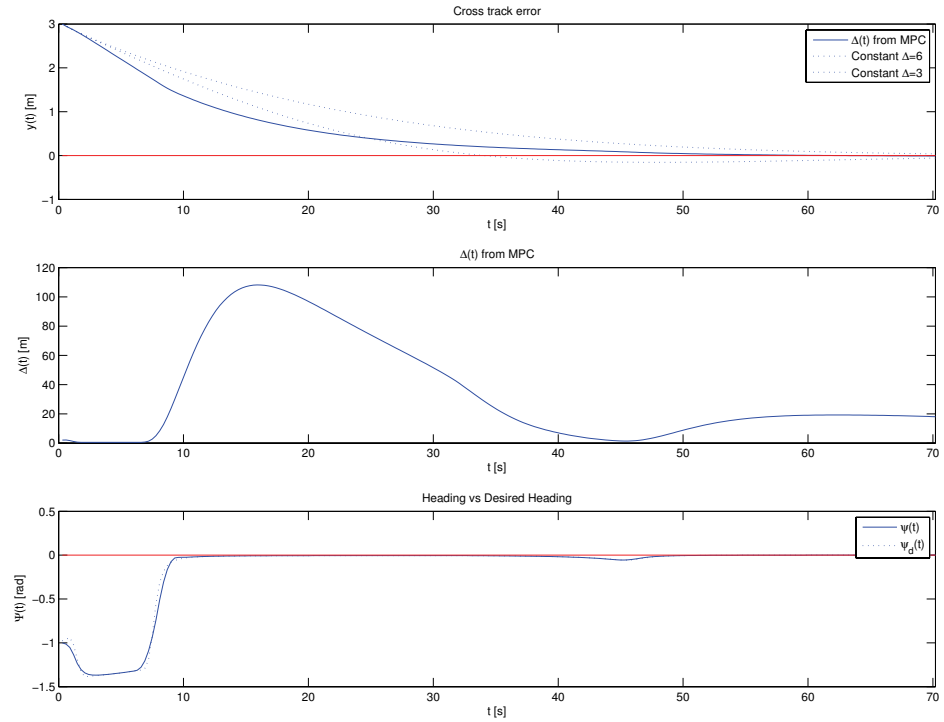

Figure 22: Simulation result for Q_1 and R_1, no current, $H_p = 180$.

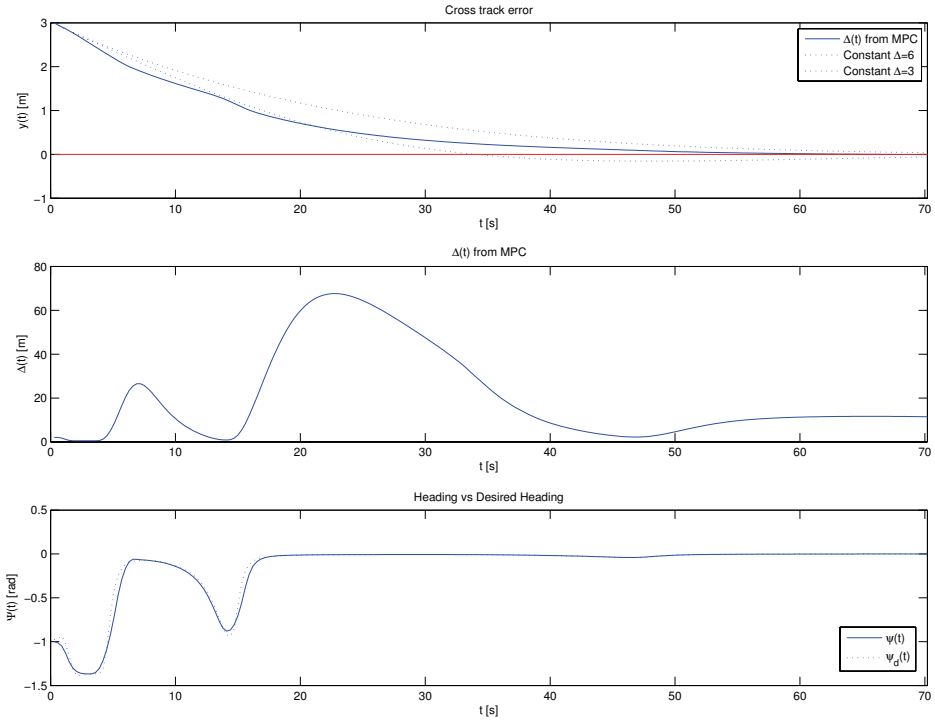

Figure 23: Simulation result for Q_2 and R_1, no current, $H_p = 80$.

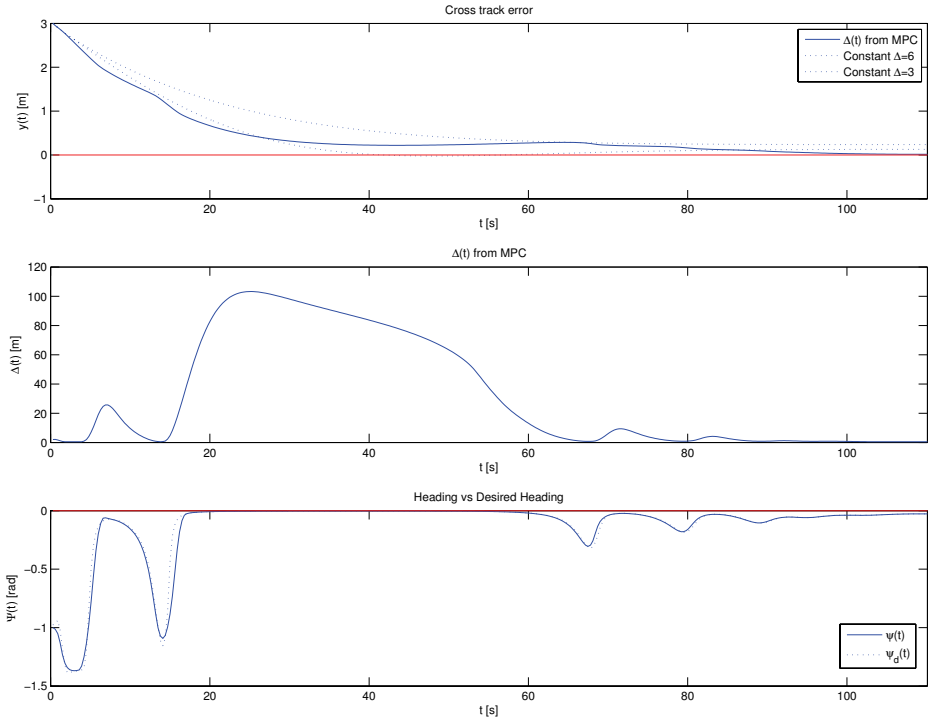

Figure 24: Simulation result for Q_2 and R_1, with current1, $H_p = 80$.

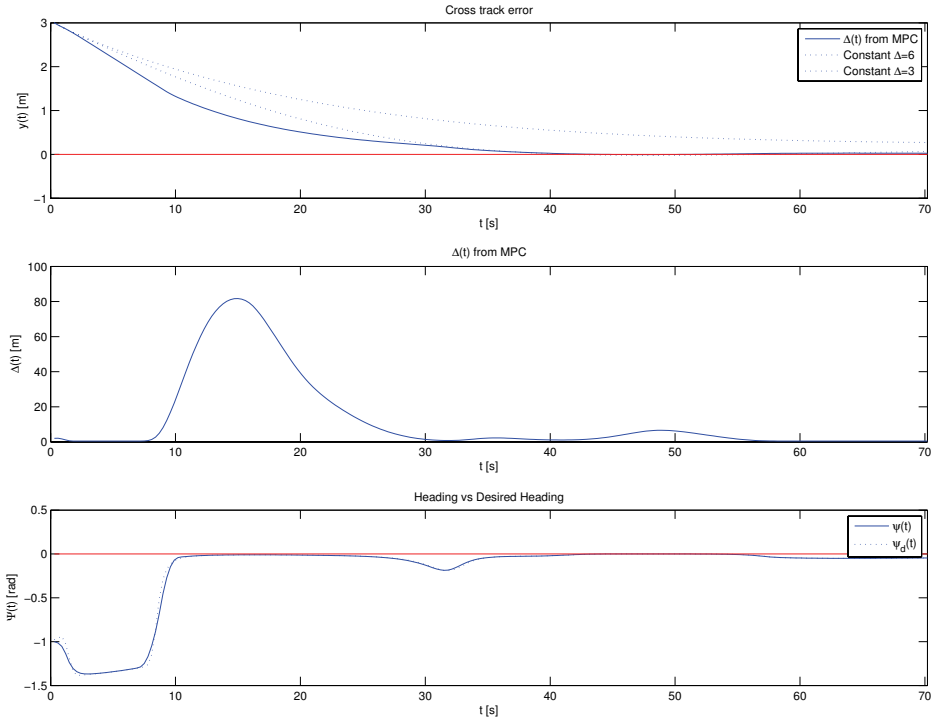

Figure 25: Simulation result for Q_1 and R_1, with current1, $H_p = 180$.

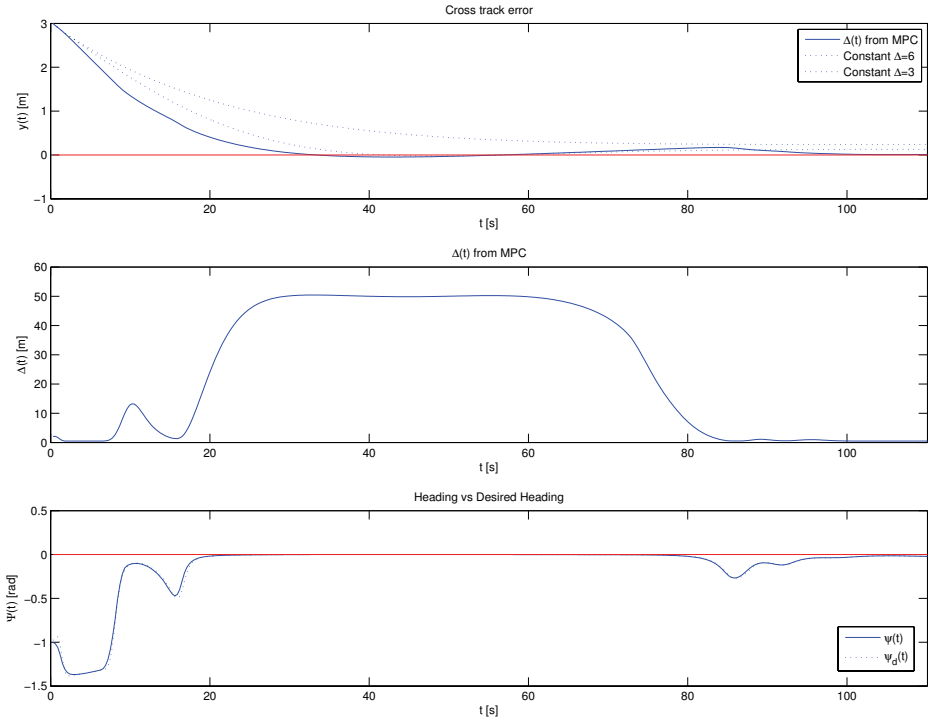

Figure 26: Simulation result for Q_3 and R_1, with current1, $H_p = 80$.

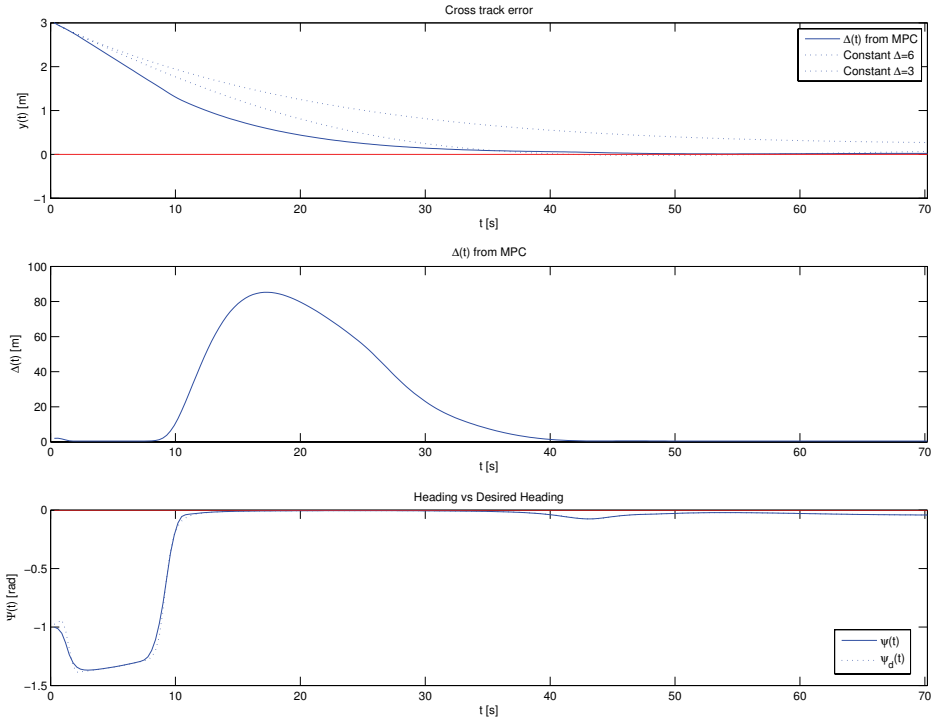

Figure 27: Simulation result for Q_3 and R_1, with current1, $H_p = 180$.

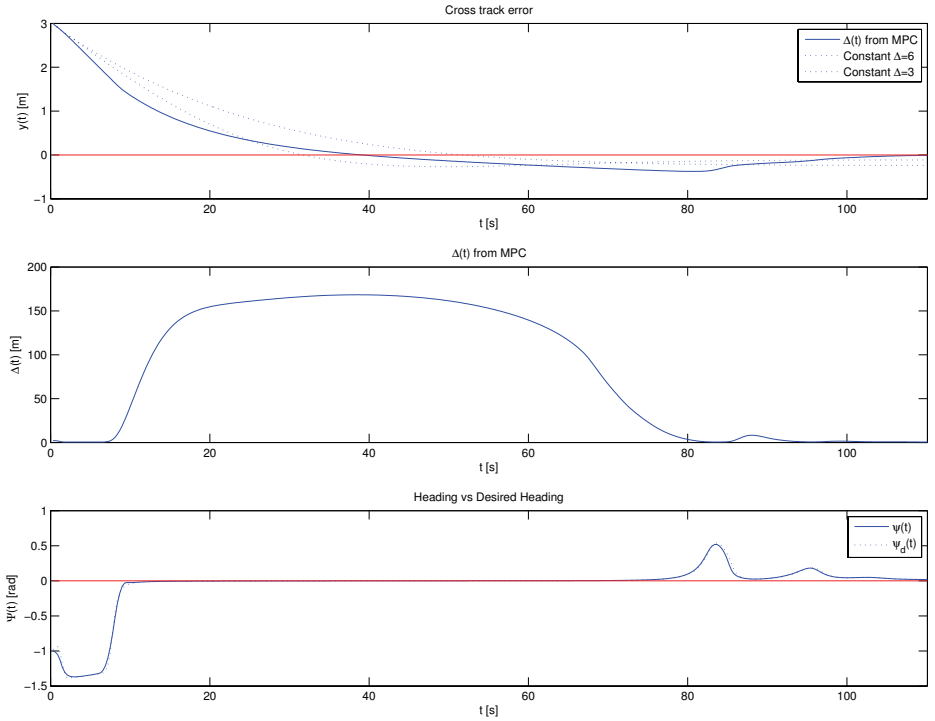

Figure 28: Simulation result for Q_3 and R_1, with current2, $H_p = 180$.

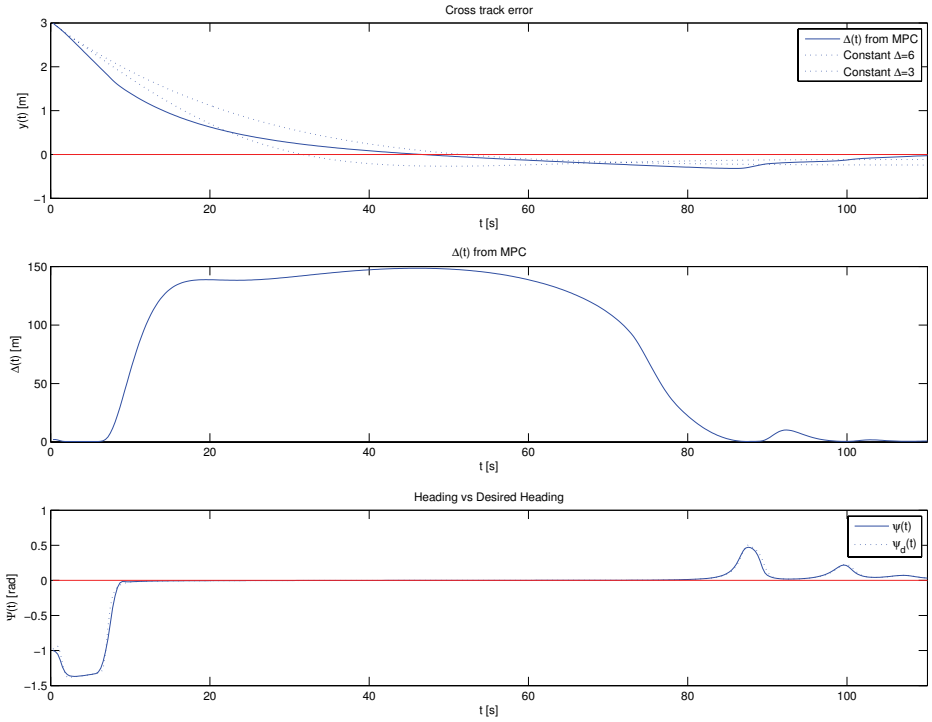

Figure 29: Simulation result for Q_1 and R_1, with current2, $H_p = 180$.

8.2 Cost function 2

In order to implement *cost function 2*, the predictions needs to be extended to included $\dot{y}$. The following parameters and weights are used for the simulations of *cost-function 2*

$$
\begin{aligned}
H_p &= 80 \\
H_u &= 25 \\
T_{s,LTV} &= 0.3 \\
T_{s,nonl} &= 0.5 \\
Q_4 &= \begin{bmatrix} 3*10^4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5*10^4 \end{bmatrix} \\
Q_5 &= \begin{bmatrix} 3*10^4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 16*10^5 \end{bmatrix} \\
Q_6 &= \begin{bmatrix} 3*10^4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2*10^6 \end{bmatrix} \\
R_2 &= \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 3.45*10^2 \end{bmatrix} \\
R_3 &= \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 200 \end{bmatrix} \\
R_4 &= \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 300 \end{bmatrix} \\
R_5 &= \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 750 \end{bmatrix} \\
k_{g_3,1} &= 0.001 \\
k_{y,1} &= 3 \\
k_{\dot{y},1} &= 160 \\
k_{\dot{y},2} &= 200
\end{aligned}
$$

where $T_{s,LTV}$ is the step length between samples when the LTV model is used, while $T_{s,nonl}$ is the length between samples when the nonlinear solver is used.

The results from minimizing the cost-function $\sum k_y y_k^2 + \Delta U^T R \Delta U$ can be found in Figure 20, notice that the cross-track error reduces fast and has a large over-shoot. This over-shoot can be reduced by adding the squared cross-track error rate to the cost-function, including a weight on both states. However, the weight on the cross-track error rate has to be large enough relative to the

weight on the cross-track error to achieve a reduction of the over-shoot. This can be seen in Figure 30 where the cross-track error has the same over-shoot as in Figure 20. The effect of adding the cross-track error rate to the cost-function is not seen until time $t \approx 30$. This is because the magnitude of the cross-track error is large relative to the magnitude of the cross-track error rate, and the weight on the cross-track error rate is too small to have an effect on the look-ahead distance until time $t \approx 30$. The look-ahead distance in Figure 30 starts to increase sooner after the over-shoot, and stays large for a longer time, than in Figure 20, which results in smaller cross-track error rate as the cross-track error converges.

When the weight on the cross-track error rate is large enough, the over-shoot is reduced. This is seen in Figure 31. However, the increased weight on the cross-track error rate also results in oscillations in the look-ahead distance, which leads to oscillations in the desired heading and the heading. The oscillations in the look-ahead distance implies large changes in the acceleration of the look-ahead distance. When the weight on the change in the acceleration of the look-ahead distance is increased, the oscillations are reduced but not removed, as seen in Figure 32. There is not much to gain by increasing the weight on the change in the acceleration of the look-ahead distance further, since at some point before the oscillations are removed, the over-shoot starts to increase, see Figure 33. This increase in over shoot can be reduced by increasing the weight on the cross-track error rate, but this does not remove the oscillations. The results in Figure 34 are the best results achieved for the LTV approach with *cost-function 2*. The look-ahead distance in Figure 34 has some oscillations up to time $t \approx 7$ seconds.

The oscillations in the look-ahead distance results from including the cross-track error rate in the cost-function, why is this? As explained in the section *MPC*, the tail of the optimal solutions U_k found at time k is not necessarily equal to the optimal solutions U_{k+1} found at time *k+1* unless the prediction and input horizons are infinite, the prediction model is perfect, and all measurements including disturbances are prefect[4]. However, the inputs $g_{i,k+1}$ predicted at time $k+1$ should be close to inputs $g_{i,k+1}$ found at time k, for large enough finite prediction and input horizons. Thus, the oscillations in the look-ahead distance should reduce when the prediction and input horizons increases, unless the prediction model is too inaccurate. As can be seen in Figure 35, the oscillations up to time $t \approx 10$ seconds increase relative to Figure 34, even though the prediction and input horizons are larger in Figure 35. Thus, the oscillations in the look-ahead distance indicates that the predicted optimal solutions $g_{i,k+j}, j = 1, 2...H_u$ are inaccurate, which indicates that the predictions made by the LTV model are inaccurate. The LTV model is an approximation of the system and will always suffer from errors introduced by assuming linear behavior between samples, however the predictions made by the LTV model appeared to be accurate enough during the simulations of *cost-function 1*.

When the matrices Γ_{k+j} are investigated, it becomes clear that these matrices are the root of the inaccuracy of the LTV model. The effect on the cross-track error rate of the inputs are estimated by $\frac{\partial \ddot{y}}{\partial g_i} g_i, i = 1, 2, 3$. That is, a negative value of $\frac{\partial \ddot{y}}{\partial g_1}$ implies that the cross-track error rate reduces if the look-ahead distance increases.

In a situation A where the cross-track error is positive, and the cross-track error rate is negative, the magnitude of the cross-track error rate should decrease, that is the cross-track error rate should increase, when the look-ahead distance increases. Thus, $\frac{\partial \ddot{y}}{\partial g_1}$ should be positive in situation A. Situation A corresponds to the situation in the simulations from time $t = 0$ up to the time where the cross-track error over-shoots. When the matrices Γ_{k+j} are investigated for situation A, it turns out that $\frac{\partial \ddot{y}}{\partial g_1}$ actually is negative, which is wrong since increasing the look-ahead distance does not increases the magnitude of the cross-track error rate in situation A. The reason for this error can be seen by studying the system equations which are descretized according to the discrete model by

[4]The simulations which have resulted in Figures 30-35 are all performed without any disturbances under the assumptions of perfect state measurements.

Thor I. Fossen [11]

$$\tau_{k+1} = f(\nu_k, \eta_k, g_{i,k}) \tag{8.2.1}$$

$$\nu_{k+1} = \nu_k + T_s[M^{-1}(-C(\nu_k)\nu_k - D\nu_k + \tau_{k+1})] \tag{8.2.2}$$

$$\eta_{k+1} = \eta_k + T_s R(\psi_k)\nu_{k+1} \tag{8.2.3}$$

$$y_{k+1} = y_k + T_s(u_{k+1}\sin\psi_k + v_{k+1}\cos\psi_k) \tag{8.2.4}$$

$$\Downarrow$$

$$\dot{y}_k = u_{k+1}\sin\psi_k + v_{k+1}\cos\psi_k \tag{8.2.5}$$

where T_s is the step length between samples. The velocities ν_{k+1} depends on the inputs $g_{i,k}$ through τ_{k+1}, while η_{k+1} depends on the inputs $g_{i,k}$ through ν_{k+1}. Since the LTV model assumes linear behavior between samples, and the sates η_{k+1} depends on ν_{k+1} and η_k, the effect of the inputs $g_{i,k}$ on the heading does not appear until time *k+1*. Thus, the effect of $g_{i,k}$ on the cross-track error rate at time k, is only through v_{k+1}, even though the heading ψ_{k+1} is effected by $g_{i,k}$. Hence, the negative value of $\frac{\partial \ddot{y}}{\partial g_1}$ in situation *A*. However, at the next time step *k+1*, the effect of inputs $g_{i,k}$ on the system is described by $\Phi_{k+1}\Gamma_k$, which results in the cross-track error rate $\dot{y}_{k+1}$ also depending on $g_{i,k}$ through ψ_{k+1}. For situation *A*, this results in $\frac{\partial \ddot{y}_k}{\partial g_{1,k}} < 0$ and $\frac{\partial \ddot{y}_{k+1}}{\partial g_{1,k}} > 0$. While at the next time step *k+1*, $\frac{\partial \ddot{y}_{k+1}}{\partial g_{1,k+1}} < 0$, which means that the effect of the look-ahead distance on the cross-track error rate $\dot{y}_{k+1}$ at time k appears to be opposite of the effect of the look-ahead distance at time *k+1* on $\dot{y}_{k+1}$. Further more, the effect of the look-ahead distance $g_{1,k}$ on the cross-track error rate $\dot{y}_k$ appears to be the opposite of the effect of the look-ahead distance $g_{1,k}$ on the cross-track error rate $\dot{y}_{k+1}$. Hence, the optimal solution for input $g_{1,k}$ at time k is not likely to be equal to the optimal solutions $g_{1,k+1}$ found at time *k+1*. This motivates for solving the optimization problems which have cost-functions that are dependent on the cross-track error rate with the nonlinear solver described in section *Nonlinear solver*. This solver uses predictions based on numerical integration of the system where the step length in the numerical integration of the descritized model in equations 8.2.2 and 8.2.3 is considerably smaller than the step length between samples. Thus, $\dot{y}_{k+1}$ depends on $g_{i,k}$ through the heading as well as the sway velocity.

The draw-back to the nonlinear approach is increased calculation time relative to the LTV approach when the time between samples, prediction horizon and input horizon is identical. However, the time between samples can be increased when the nonlinear solver is used because the accuracy of the predictions is independent of the time between samples. This is because the predictions are made by numerical integration of the nonlinear system equations with the sampled states as initial values. Increased time between samples only means increased time between new inputs, assuming the predictions are made for the same length time interval. When the calculation time of the nonlinear approach is to be compared with the calculation time of the LTV approach, the optimization should be performed over the same length time intervals. That is $H_{p,LTV}T_{s,LTV} = H_{p,nonlin}T_{s,nonlin}$ and $H_{u,LTV}T_{s,LTV} = H_{u,nonlin}T_{s,nonlin}$ should hold. Thus, the prediction and input horizons, $H_{p,nonlin}$ and $H_{u,nonlin}$, can be reduced, which results in reduced calculation time. However, the LTV approach is generally faster.

The simulation results for the nonlinear solver, with weights corresponding to the weights in Figure 31, can be found in Figure 36. The oscillations in the desired heading are removed, even when the weight on the acceleration is small. The cross-track error reduces fast, but has a over-shoot, and converges to approximately zero. The look-ahead distance reduces fast to compensate for the over-shoot and stays small. This is a very desirable property. The increased heading at time $t \approx 35$ seconds results from the small-look ahead distance, and is necessary to reduce the cross-track error quickly after the over-shoot. Notice that the look-ahead distance increases at time $t \approx 50$ seconds, this reduces the cross-track error rate and a second over-shoot is avoided.

The over shoot is reduced when the weight on the cross-track error rate is increased, this is seen in Figure 37 where the cross-track error converges to approximately zero with only a slight over-shoot. However, this is at the cost of slower convergence. Hence, there is a trade off between fast convergence and small over-shoot.

The weights leading to Figure 36 are used when the vessel is subjected to *current1* and *current2*. The cross-track error in Figure 38, where *current1* is used, converges fast to approximately zero, with only a small over-shoot. This is because the look-ahead distance increases fast to reduce the over-shoot, and because *current1* acts to damp the over-shoot. The small over-shoot is compensated for fast, since the look-ahead distance is small after the over-shoot. The cross-track error in Figure 39, where *current2* is used, converges fast to approximately zero, but has larger over-shoot than in Figure 38. The larger over-shoot is caused by *current2* which acts to increase the over-shoot. However, the cross-track error in both Figures 38 and 39 converges faster, to a smaller value and with smaller over-shoot than for both constant look-ahead distances.

The simulations performed with LTV-model based optimization, has reveal a weakness in the LTV-model. That is, the LTV approximation of the cross-track error rate dynamic is not accurate enough. This has lead to the use of a non-linear optimization solver, which in general spends more time calculating the optimal inputs than the LTV-model based optimization solver. The simulations performed with this cost-function last for approximately 300 seconds for the nonlinear solver, and approximately 190 seconds for the LTV approach, while the time simulated is 70 seconds. Thus, work should be put into finding a better LTV-model of the system, since this would reduce the calculation time of the inputs. The simulations performed with the non-linear solver, with and without current, has shown that *cost-function 2* performs well for proper weights, and the chosen initial values of the states. Fast convergence of the cross-track error with little or no over-shoot has been achieved.

Note *Though the look-ahead distance may appear to be very close to zero in some figures, it is never smaller than 0.5 meters, which is the lower boundary used by the optimization solvers.*

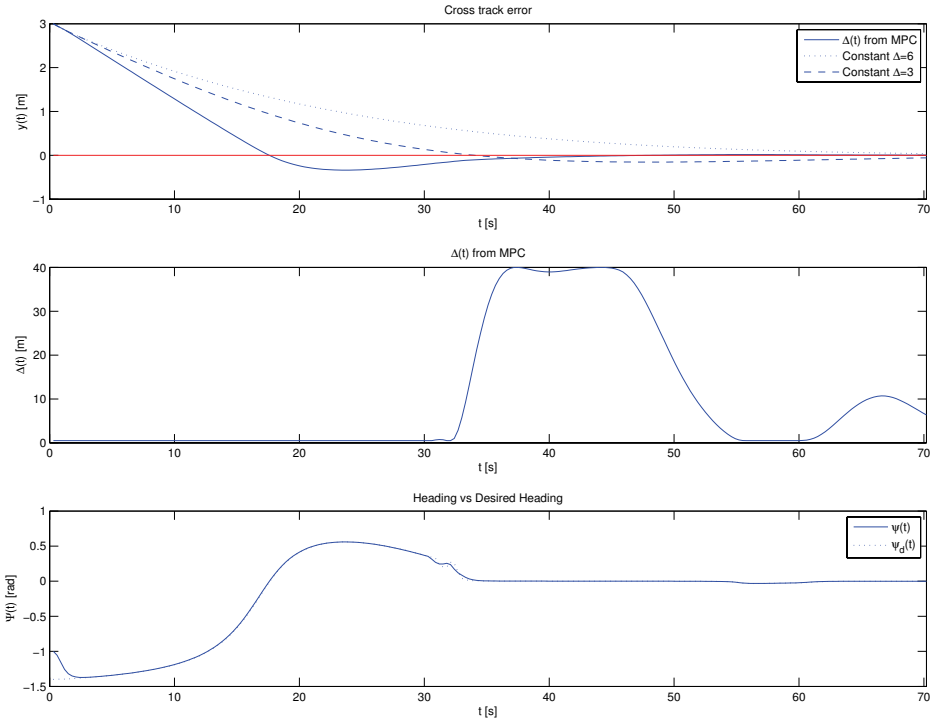

Figure 30: Simulation result for Q_4 and R_2, $H_p = 80$.

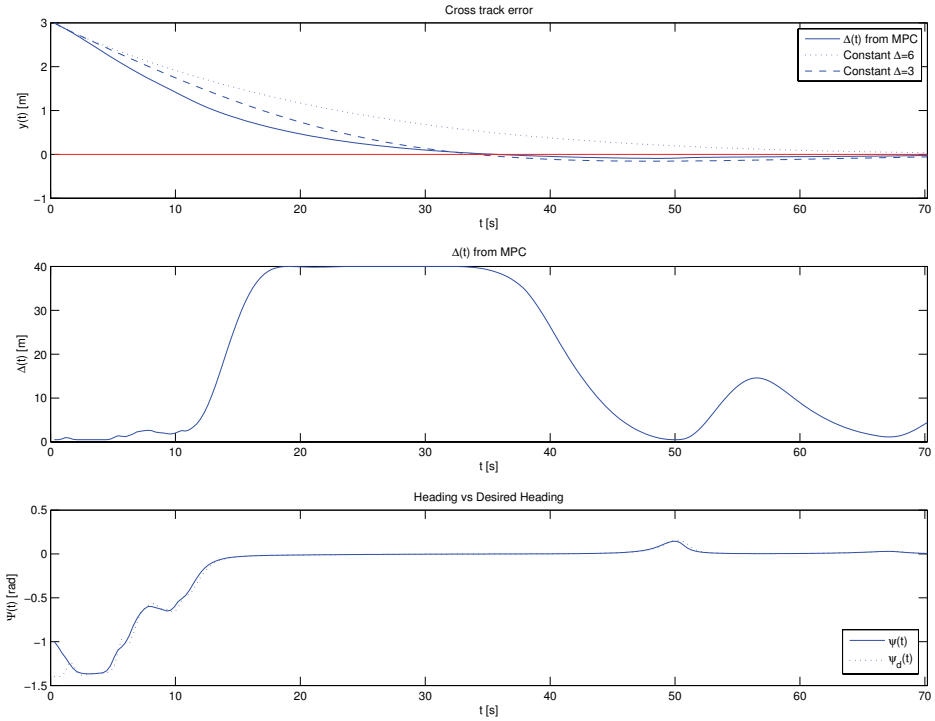

Figure 31: Simulation result for Q_5 and R_2, $H_p = 80$.

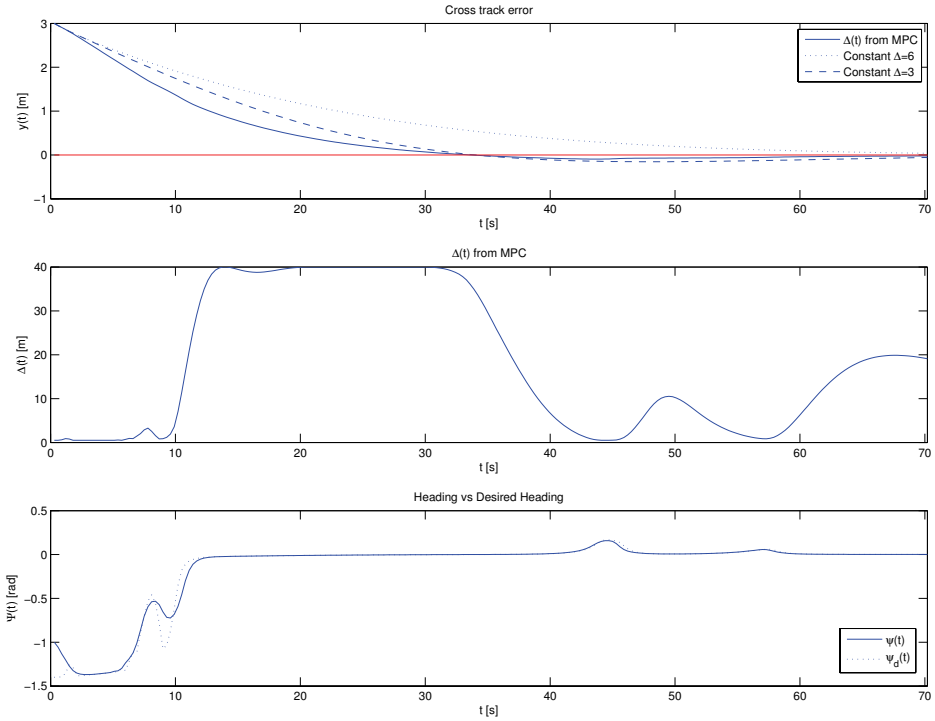

Figure 32: Simulation result for Q_5 and R_3, $H_p = 80$.

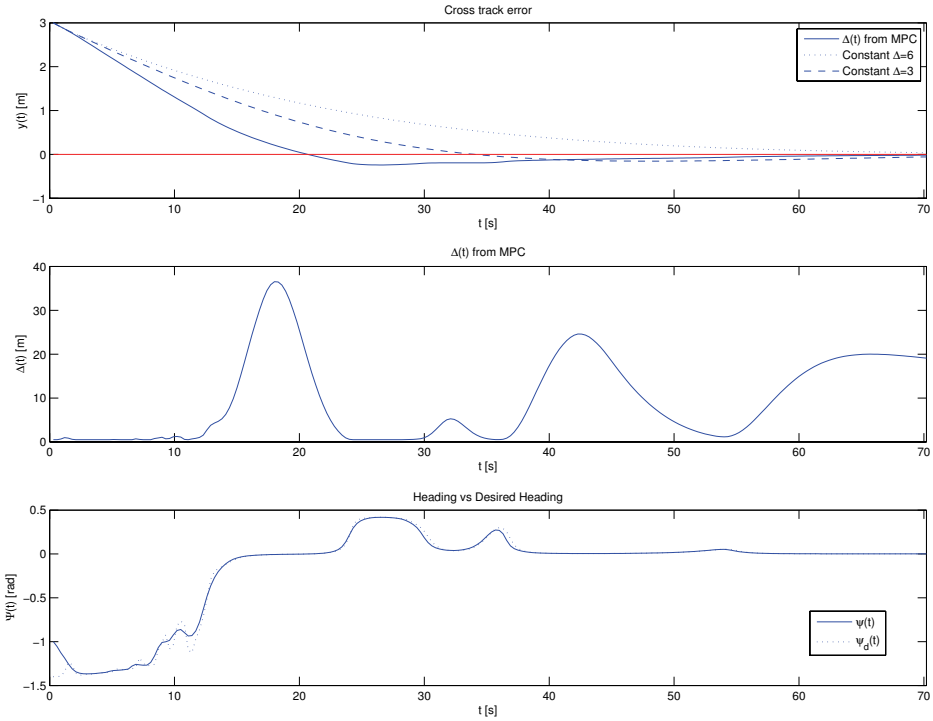

Figure 33: Simulation result for Q_5 and R_4, $H_p = 80$.

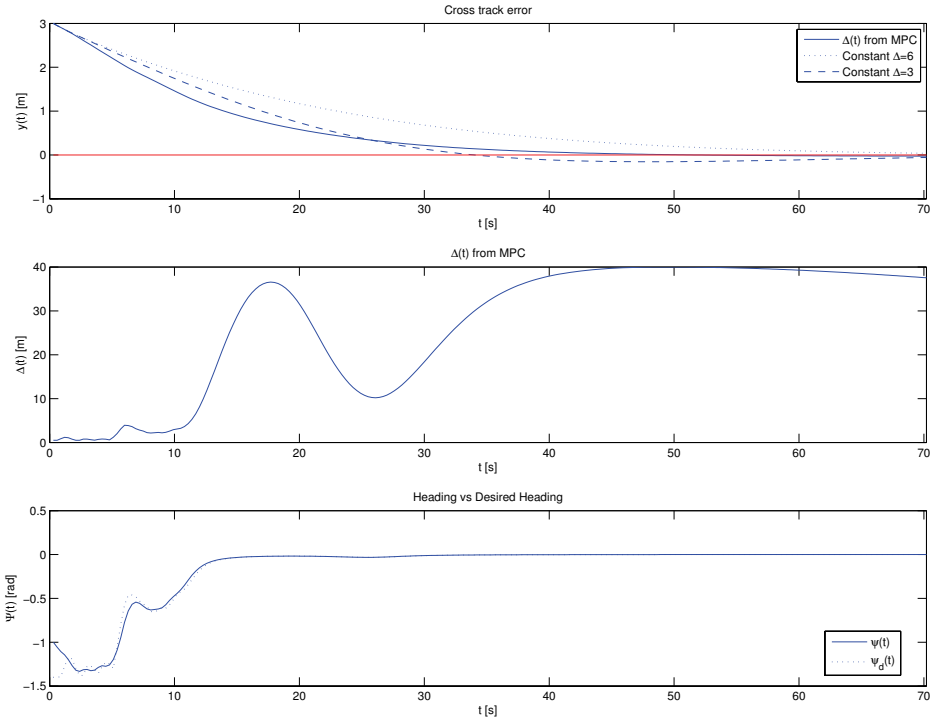

Figure 34: Simulation result for Q_6 and R_5, no current $H_p = 80$.

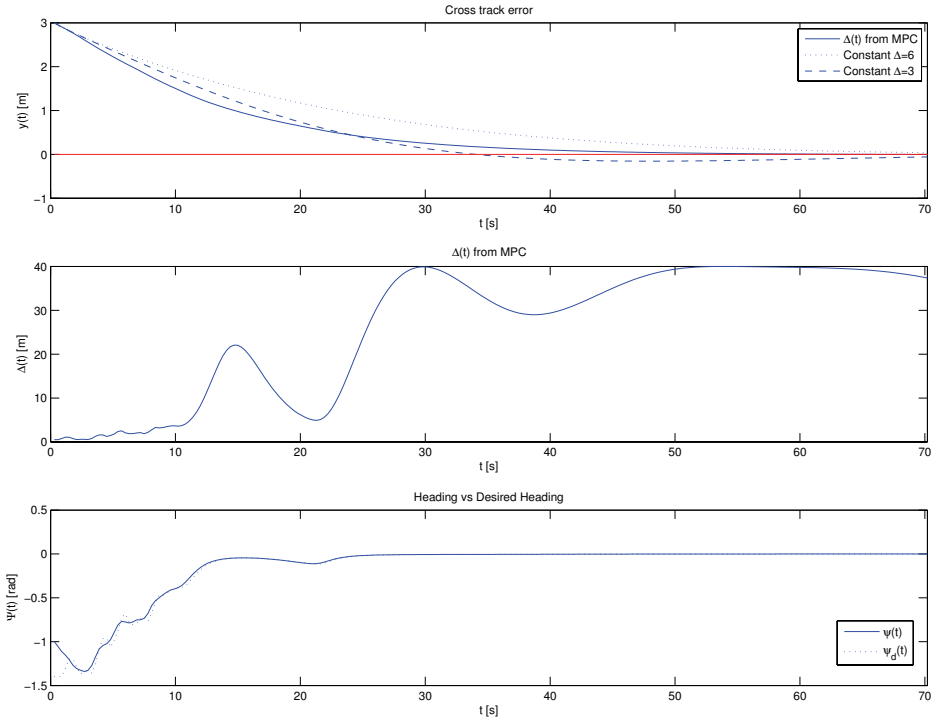

Figure 35: Simulation result for Q_6 and R_5, no current $H_p = 180$ and $H_u = 50$.

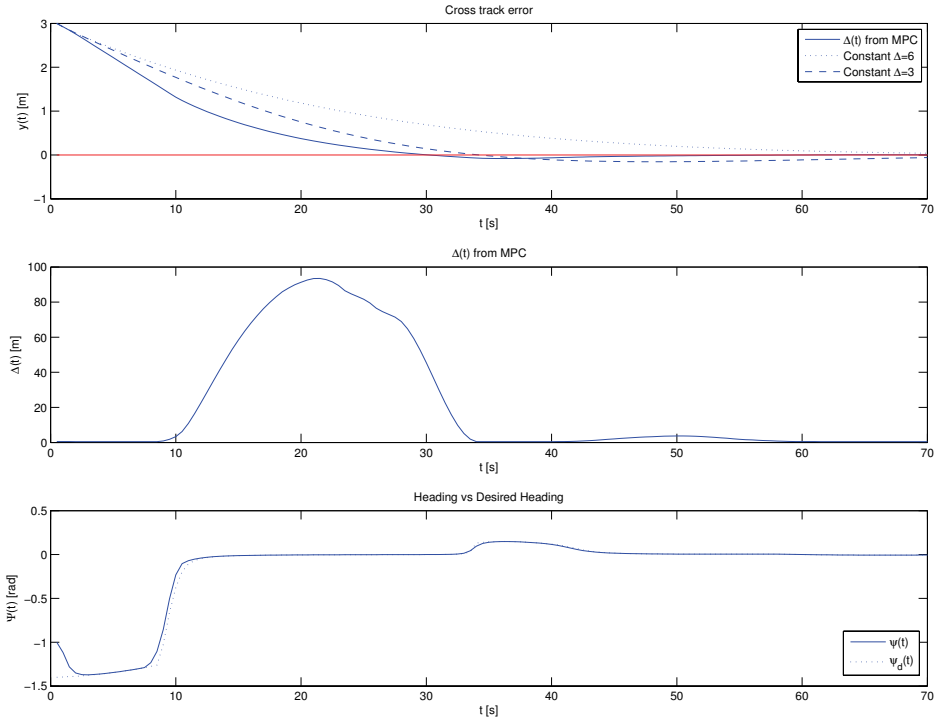

Figure 36: Simulation result nonlinear optimizer, no current, $k_{y,1}$, $k_{\dot{y},1}$, $k_{g_3,1}$, $H_p = 48$ and $H_u = 15$.

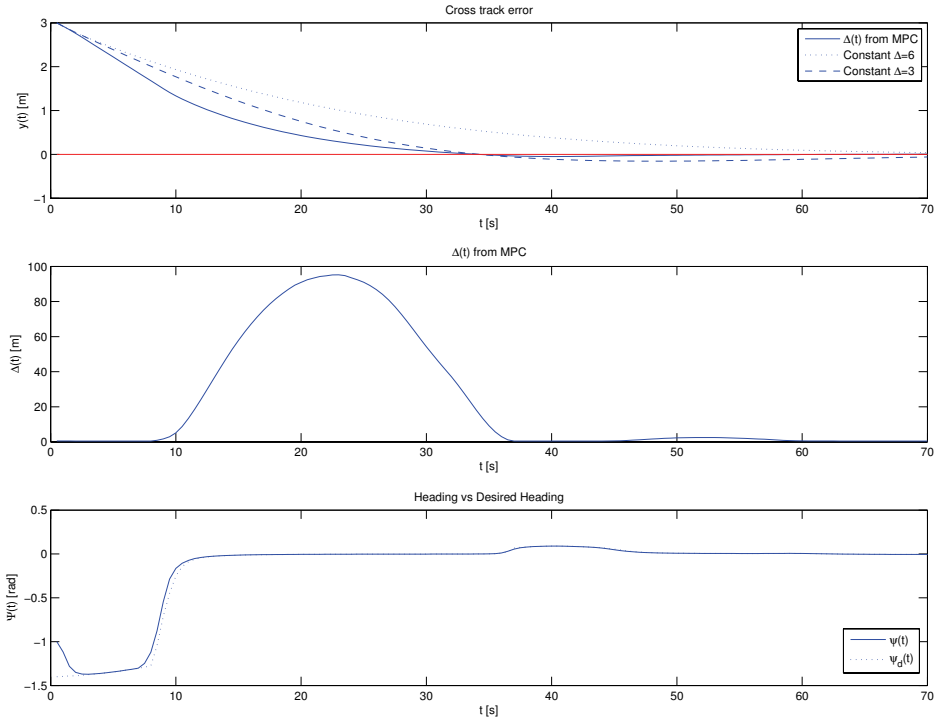

Figure 37: Simulation result nonlinear optimizer, no current, $k_{y,1}$, $k_{\dot{y},2}$, $k_{g_3,1}$, $H_p = 48$ and $H_u = 15$.

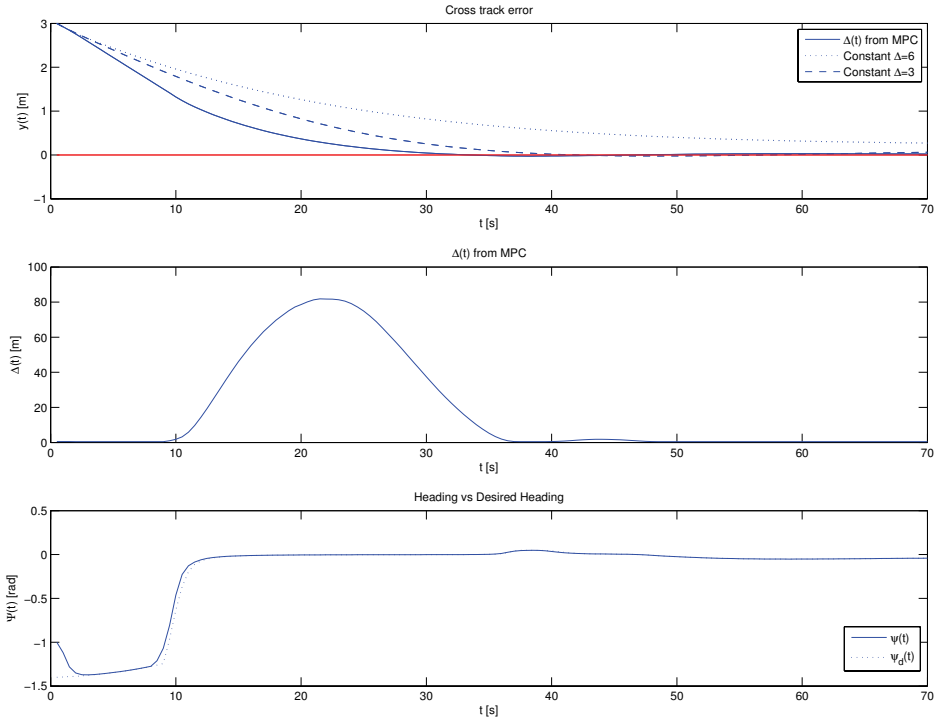

Figure 38: Simulation result nonlinear optimizer, current1, $k_{y,1}$, $k_{\dot{y},1}$, $k_{g_3,1}$, $H_p = 48$ and $H_u = 15$.

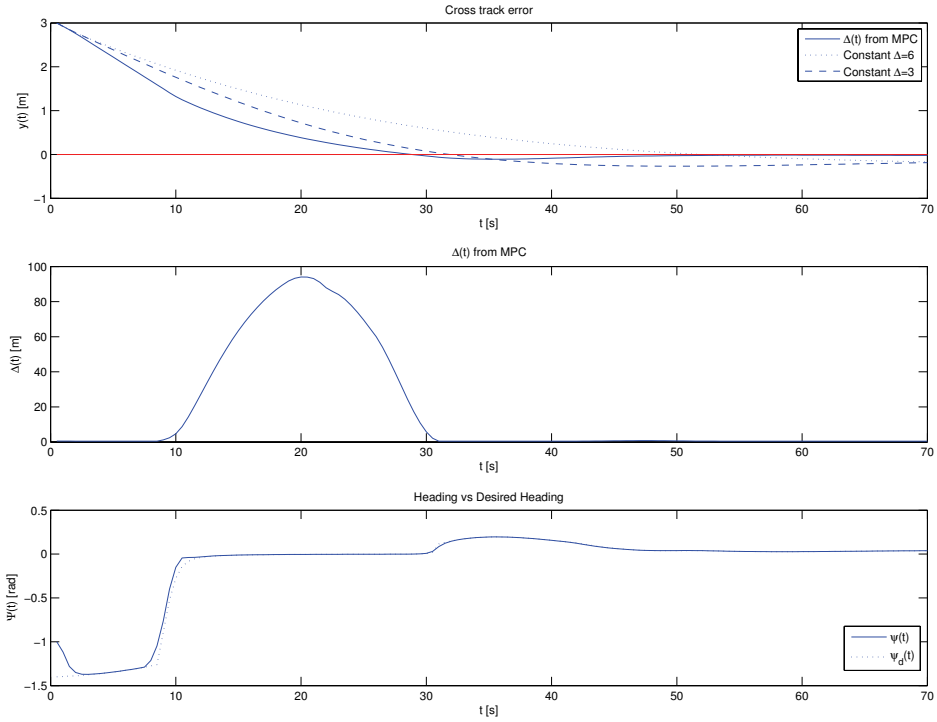

Figure 39: Simulation result nonlinear optimizer, current2, $k_{y,1}$, $k_{\dot{y},1}$, $k_{g_3,1}$, $H_p = 48$ and $H_u = 15$.

8.2.1 Cost function 3

Due to the problems with the LTV model found in the previous section, this cost-function will be minimized using the nonlinear solver. The cost-function is on the form

$$V(x,g) = \sum k_e(k_y yk + \dot{y}_k)^2 + k_g g_{3,k}$$

where the weight k_e weighs the deviation from exponential convergence. The following parameters and weights have been used in the simulations in this section

$$\begin{aligned} H_p &= 48 \\ H_u &= 15 \\ T_s &= 0.5 \\ k_{y,2} &= 0.04 \\ k_{y,3} &= 0.08 \\ k_{g,3} &= 0.0001 \\ k_e &= 4 \end{aligned}$$

The weight k_y corresponds to $1/T_c$ where T_c is the time constant of the desired cross-track error $y_0 e^{-t/T_c}$. Thus it is expected that increasing k_y results in faster reduction of the cross-track error. Figure 40 shows the results of a simulation for k_y small. The cross-track error in Figure 40 does converge faster than for the two constant look-ahead distances, however faster convergence was obtained for *cost-function 2*, which means that k_y is to small and should be increased.

The convergence rate of the cross-track error is increased when k_y is increased. This is seen in Figure 41, where the cross-track error converges faster than in Figure 40, however the over-shoot in the cross-track error in Figure 41 is larger than in Figure 40. The look-ahead distance has a slight increase at time $t \approx 42$, which reduces the cross-track error rate so that the cross-track error converges to approximately zero at time $t \approx 52$ seconds, without any significant over-shoot. The look-ahead distances in Figure 41 is small when the cross-track error is large, increases rapidly to avoid over-shooting, and reduces rapidly and stays small when the cross-track error is small. This behavior is as previously discussed desirable since it results in fast reduction of the cross-track error, small over-shoot, and suppression of environmental forces such as currents.

Since a large value of k_y results in faster convergence at the cost of increased over-shoot, there is a trade off between fast convergence and small over-shoot. In the case of a constant current, the effect of the current is suppressed fast when the weight $k_{y,3}$ is used. This can be seen in Figures 42 and 43, where Figure 42 is the result of simulation with *current1*, while Figure 43 is the result of simulation with *current2*. As for *cost-function 2*, *current2* causes a larger over-shoot than *current1*.

This cost-function formulation results in good performance in the presence of both *current1* and *current2*, that is the cross-track error reduces fast, has little or no over-shoot and converges to approximately zero significantly faster than for any of the two constant look-ahead distances. The performance in terms of over-shoot and convergence rate is much the same as for *cost-function 2*. However, *cost-function 3* seems to have the property that the predicted solutions at time k is very close to the solutions at time *k+1* because after a few iterations, the calculation time reduces so that the total simulation time is close to the time simulated. When 70 seconds are simulated the simulation last for approximately 65-77 seconds. That is, the simulations where cost-function 3 is minimized by the nonlinear solver runs faster than for the LTV approach where *cost-function 1* is minimized which runs in approximately three times the time simulated. This is not the case when *cost-function 2* is minimized by the nonlinear solver. However, the calculations in the initial iterations lasts for more than one sample interval so not all time limits are met, still this is a promising result.

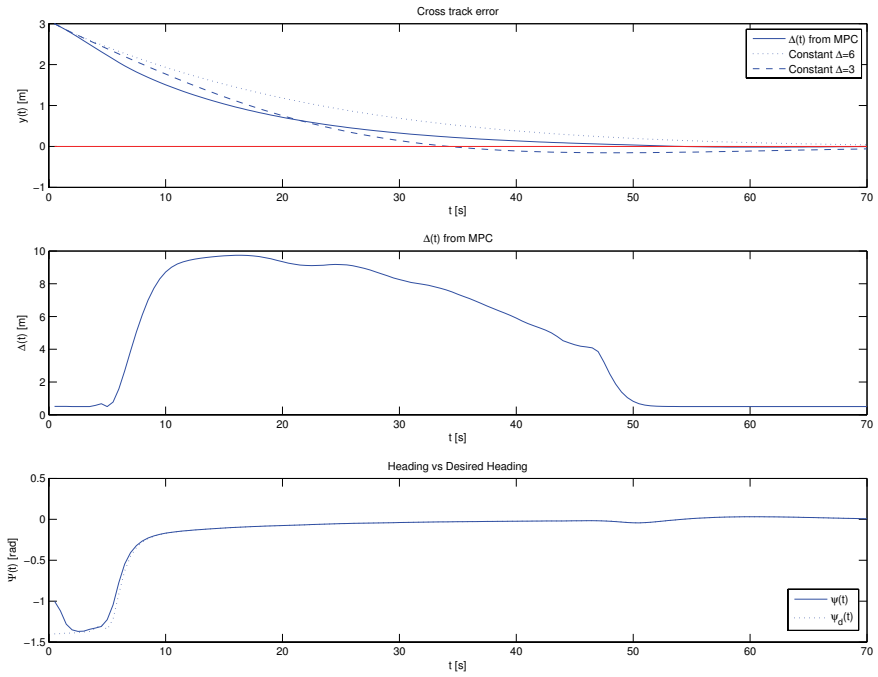

Figure 40: Simulation result for $k_{y,2}$ and $k_{g,3}$, no current.

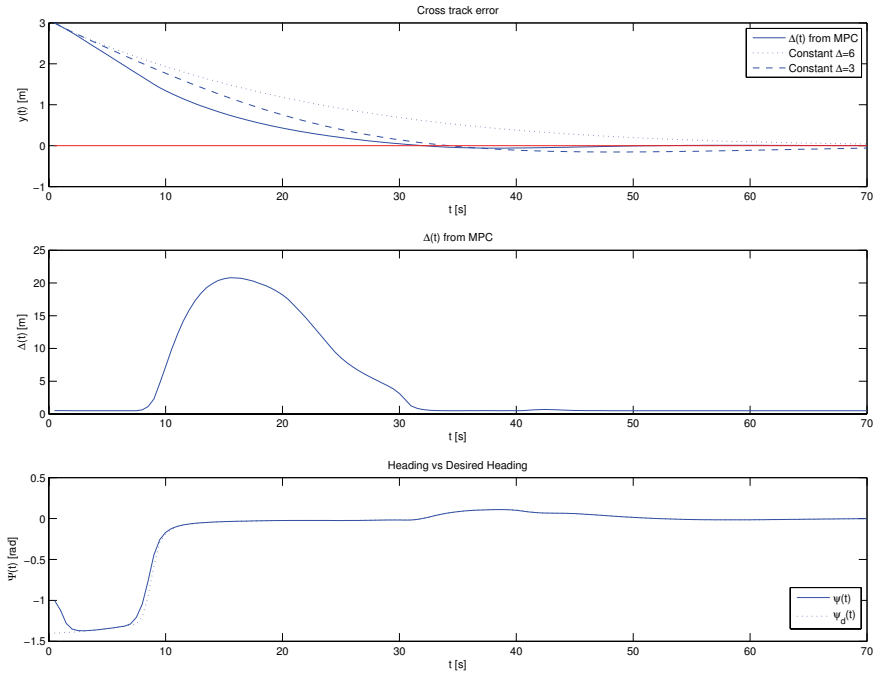

Figure 41: Simulation result for $k_{y,3}$ and $k_{g,3}$, no current.

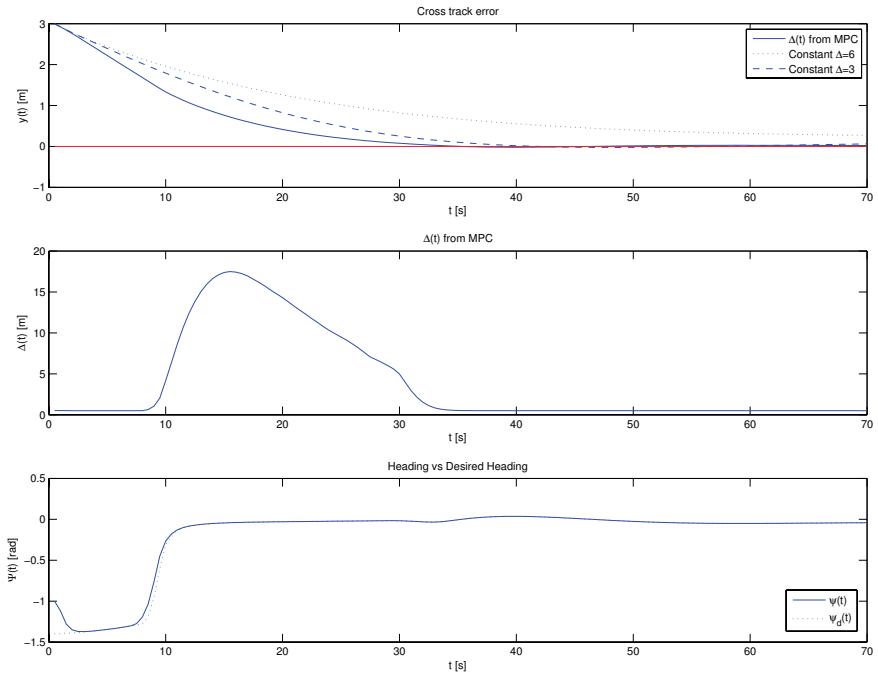

Figure 42: Simulation result for $k_{y,3}$ and $k_{g,3}$, current1.

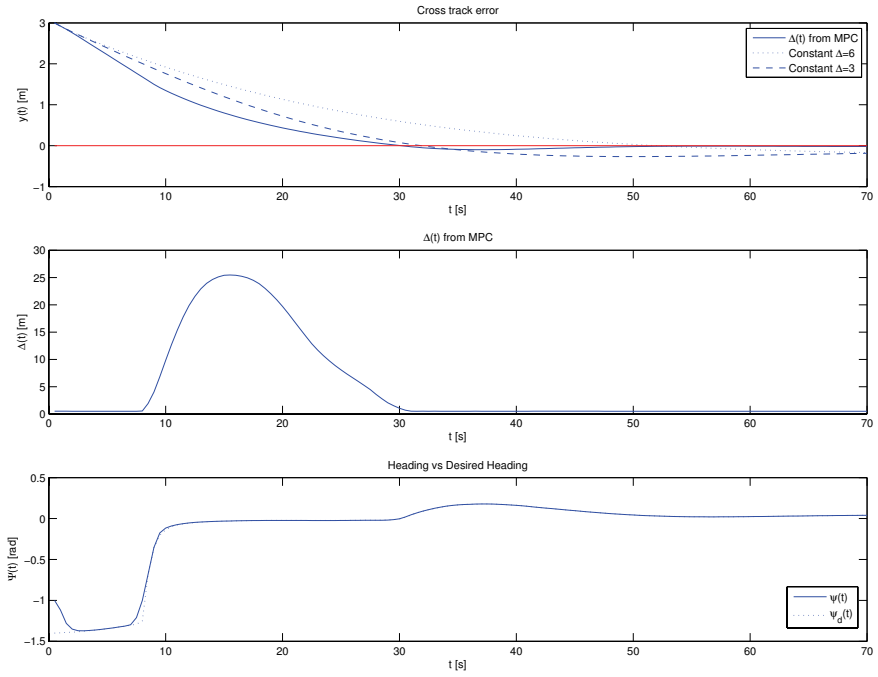

Figure 43: Simulation result for $k_{y,3}$ and $k_{g,3}$, current2.

8.2.2 Cost function 6

The weights which gave good results for *cost-function 3* will be used as an initial guess of the weights for this cost-function since this cost-function is the same as *cost-function3* where weights on the desired surge-velocities deviation from the desired surge-velocity on the path ($\overline{u}_d$) and the desired surge velocity rate ($\dot{u}_d$) are added. The added weights should be chosen such that the desired surge-velocity is allowed to deviate from the desired surge-velocity on the path only when this increases the convergence rate of the cross-track error. That is, the desired surge-velocity should converge to the desired surge-velocity on the path when the cross-track error converge to zero.

The parameters and weights used in the simulations with this cost-function are

$$H_p = 48 \tag{8.2.6}$$
$$H_u = 15 \tag{8.2.7}$$
$$T_s = 0.5 \tag{8.2.8}$$
$$k_{e,1} = 4 \tag{8.2.9}$$
$$k_{\overline{u}_d,1} = 20 \tag{8.2.10}$$
$$k_{\overline{u}_d,2} = 10 \tag{8.2.11}$$
$$k_{\dot{u}_d,1} = 3 \tag{8.2.12}$$
$$k_{\dot{u}_d,2} = 4 \tag{8.2.13}$$
$$k_{g_3,1} = 0.0001 \tag{8.2.14}$$
$$k_{y,5} = 0.12 \tag{8.2.15}$$

Figure 45 shows the results from a simulation with the initial choice of weights. Notice that the look-ahead distance does not reduce after the cross-track error over-shoots. When the cost-function is investigated by reducing the look-ahead distance and inserting the initial values corresponding to the state of the system at time $t = 70$ seconds, it becomes clear that the cost-function does reduce when the look-ahead distance decreases. Thus, the look-ahead distance should be decreasing. Close inspection of the look-ahead distance shows that it does reduce slowly, but not to the values of the look-ahead distance which minimizes the cost-function. When the cost of $(k_y y_k + \dot{y}_k)^2$ is plotted with respect to increasing values of the look-ahead distance, the root of the problem becomes clear. As can be seen in Figure 44, the term $e^2 = (k_y y_k + \dot{y}_k)^2$ is not convex for values of the look-ahead distance greater than $\Delta \approx 4$ meter, when the system is in the state at time $t = 70$ seconds. It should be noted at this point that Figure 44 has been produced under the assumption of u, v, $\overline{y}$ and y constant and equal to their respective values at time $t = 70$ and by evaluating the following equations for increasing Δ

$$\overline{y}_k = y_k + \varepsilon \sin \psi_{k-1} \tag{8.2.16}$$
$$\psi_{d,k} = -atan(\frac{\overline{y}_k}{\Delta}) \tag{8.2.17}$$
$$\psi_k = \psi_{d,k} \tag{8.2.18}$$
$$\dot{y}_k = u_k \sin \psi_k + v_k \cos \psi_k \tag{8.2.19}$$
$$e_k^2 = (k_y y_k + \dot{y}_k)^2 \tag{8.2.20}$$

The cost e^2 appares to have only one minimum, however this does not imply that the cost-function does not have two or more minima. Even if the cost-function has only one minimum and is not convex, the optimization-problem solver may terminate with a non-optimal solution. This is because the gain in reducing the look-ahead distance appears to be smaller than it is, due to the small magnitude of the directional derivative in the search direction, which is one of the criteria that terminates *fmincon*. The directional derivative is [9]

$$D(f(x); p) = \nabla f(x)^T p$$

where we have that

$$f(x + p) = f(x) + \nabla f(x + \alpha p)^T p$$

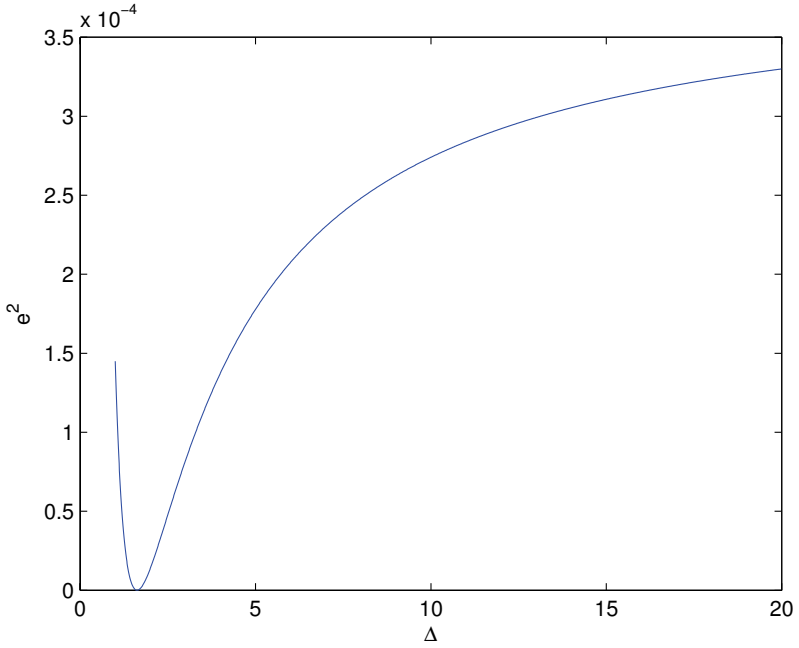

Figure 44: The term $e^2 = (k_y y_k + \dot{y}_k)^2$ as a result of increasing Δ

for some $\alpha \in (0, 1)$. That is $\nabla f(x)^T p$ is the instantaneous rate of change of *f(x)* moving through x in the direction p. When $\nabla f(x)^T p$ is small, *f(x+p)* is close to the minimum if *f(x)* is convex. The optimization terminates before the optimal solution is reach because the directional derivative satisfies the termination criterion. As the printout from *fmincon* also states: *magnitude of directional derivative in search direction less than 2*options.TolFun*[5]. Note that this is the most common criteria which causes *fmincon* to terminate in the simulations performed in this thesis, which is no problem as long as the look-ahead distance is in the convex region.

The find of only partial convexity of *cost-function 6* implies that *cost-function 3* is also only partially convex. Which brings up the question, why did not this problem surface for *cost-function 3*? The answer is also the solution to the problem in Figure 45. There is a difference between the implementations used for simulations with *cost-function 3* and *cost-function 6*. Which is that the initial guess of the optimal look-ahead distance is chosen differently. This is just a coincident, however fortunate. The initial guess used for *cost-function 3*, U_1, puts the last inputs to zero, including the look-ahead distance Δ_{k+H_u}. This breaks the constraint on the look-ahead distance, which the solver *fmincon* handles by finding a feasible value for Δ_{k+H_u}, which also is within the convex-region. This is because $\Delta = 0$ breaks the lower bound $\Delta_{min} = 0.5$ and *fmincon* choose a feasible value of Δ close to the lower bound. However, since the documentation of *fmincon* does not guarantee that this will always be true, it would be better to set the last Δ in the initial guess of the optimal look-ahead distance to Δ_{min}.

The significance of having the last initial guess Δ_{k+H_u} small, is that this ensures that the cost-function will increase if Δ_{k+H_u} is increased, further, if Δ_{k+H_u-1} is large, the acceleration $\ddot{\Delta}_{k+H_u-2}$ and the rate $\dot{\Delta}_{k+H_u-1}$ will have to be large to satisfy the equality constraints. The acceleration $\ddot{\Delta}$ is weighed, hence, the cost-function will reduce if the acceleration is reduced. Reducing the acceleration by reducing Δ_{k+H_u-1} will be cheaper than reducing the acceleration by increasing Δ_{k+H_u}, hence look-ahead distance Δ_{k+H_u-1} will be reduced. Since look-ahead distances prior to Δ_{k+H_u-1} also are constrained to ensure that the look-ahead distance is continuous, the effect may spread. At the next time step, the optimal look-ahead distances will be shifted once in time and used as the initial guess. This has the effect of moving the small look-ahead distances at the end of the solution

[5]For more information on this termination criterion, see the *Matlab* support website [16].

vetor forward in time, and another small look-ahead distance will be inserted at the last place in the vector. Thus any solution outside of the convex region, will approach the convex region.

Figure 46 shows the results for a simulation where the initial guess of the look-ahead distance is taken to be the previous solution shifted once in time and where $\Delta_{k+H_u} = \Delta_{min}$, the weights are as in Figure 45. The cross-track error in Figure 46 converges fast to approximately zero, and has a small over-shoot. The surge-velocity increases when the cross-track error is large, and decreases as the cross-track error reduces, which, in combination with the increase in the look-ahead distance, results in only a small over-shoot. The look-ahead distance begins to decrease before the over-shoot, and is small when the cross-track error over-shoots. The heading increases at time $t \approx 37$ seconds, as a result of the small look-ahead distance, and causes the cross-track error to converge fast to approximately zero. The increase in the look-ahead distance at time $t \approx 50$ seconds acts to reduce the cross-track error rate so that the cross-track error does not over-shoot a second time. The surge-velocity has a small peak at time $t \approx 10$ seconds, which is shortly after the look-ahead distance begins to increase. This peak might be caused by a too low weight on the desired surge velocity rate, or by the lack of estimation and thus the lack of opportunity to weigh the desired surge velocity acceleration.

Notice that the surge velocity in Figure 46 does not deviate much from the desired velocity on the path. If the weight $k_{\overline{u}_d}$ is reduced, the surge velocity can vary more in order to achieve faster convergence. The results in Figure 47 were obtained by reducing the weight $k_{\overline{u}_d}$ and increasing the weight $k_{\dot{u}_d}$. The surge velocity in Figure 47 does not have the peak found in the surge velocity in Figure 46, however it still converges fast to the desired velocity on the path. This is desired, though it would be better if the surge velocity reduced more to avoid the over-shoot. However, the over-shoot is so small that the cost of reducing the desired surge velocity is too large, so it is not much to gain if the weight on $k_{\overline{u}_d}$ is reduces further. The cross-track error in Figure 47 converges slightly faster and the over-shoot is slightly smaller than in Figure 46.

The results from subjecting the vessel to *current1* and *current2* can be found in Figures 48 and 49 respectively. The cross-track error in Figure 48 converges fast and has virtually no over-shoot. The look-ahead distance is small when the cross-track error is large, increases as the cross-track error reduces, and reduces as the cross-track error becomes small. There are some slight oscillations in the look-ahead distance rate at time $t \approx 33$ to $t \approx 40$ seconds. These oscillations can be seen as the flat spot in the look-ahead distance at time $t \approx 33$ seconds which is followed by fast reduction of the look-ahead distance at time $t \approx 35$ seconds. However, the oscillations in the look-ahead distance rate does not cause oscillations in the heading since the reducing cross-track error causes $-atan(\frac{\overline{y}}{\Delta})$, where $\overline{y} = y + \varepsilon \sin\psi$, to increase , that is the magnitude decreases when the heading is negative, until the cross-track error is approximately zero. The cross-track error in Figure 49 converges fast, but has larger over-shoot than in Figure 48. However, this is expected since the current in Figure 48 acts du damp the over-shoot while the current in Figure 49 acts to increase the over-shoot. Again the look-ahead distance and the surge velocity behaves as desired.

The look-ahead distance resulting from *cost-function 6* gives fast convergence of the cross-track error to a value close to zero and only small over-shoot, for both *current1* and *current2*. This is in contrast to the cross-track errors from the two constant look-ahead distances. The heading does not have oscillations, and the surge velocity converges to the desired velocity on the path. However, it should be noted that the inclusion of the surge velocity to the problem comes at the cost of increased calculation time relative to the calculation time when *cost-function 3* was used, however, the calculation time when *cost-function 6* is used is much less than when *cost-function 5* is used. However, there is still much to gain by optimization of the code, reducing the time between samples and by reducing the input and prediction horizons. Further work should be put into identifying the convex-region of *cost-function 3 and 6*, so that a measure to stay in the convex-region other than the one suggested in this section can be identified, or so that the measure taken in this section can be proven to hold. This is because the measure taken in this section, and section *cost-function 2, 3 and 5* is based on intuitive arguments and not on any mathematical proof.

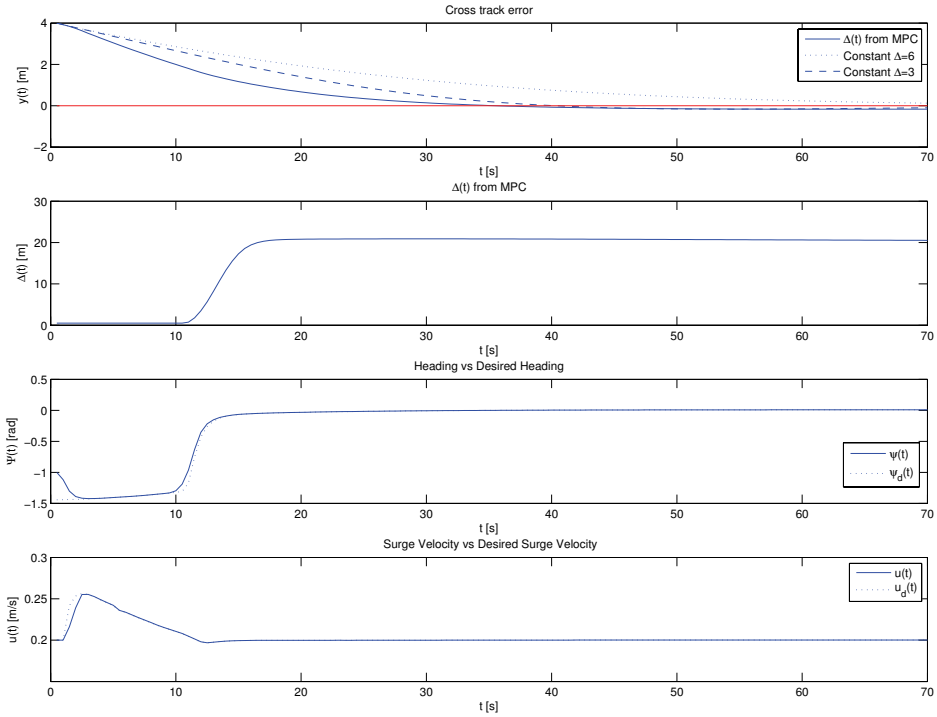

Figure 45: Simulation result for $k_{y,5}$, $k_{\overline{u}_d,1}$, $k_{\dot{u}_d,1}$, $k_{g_3,1}$ no current, initial guess U1.

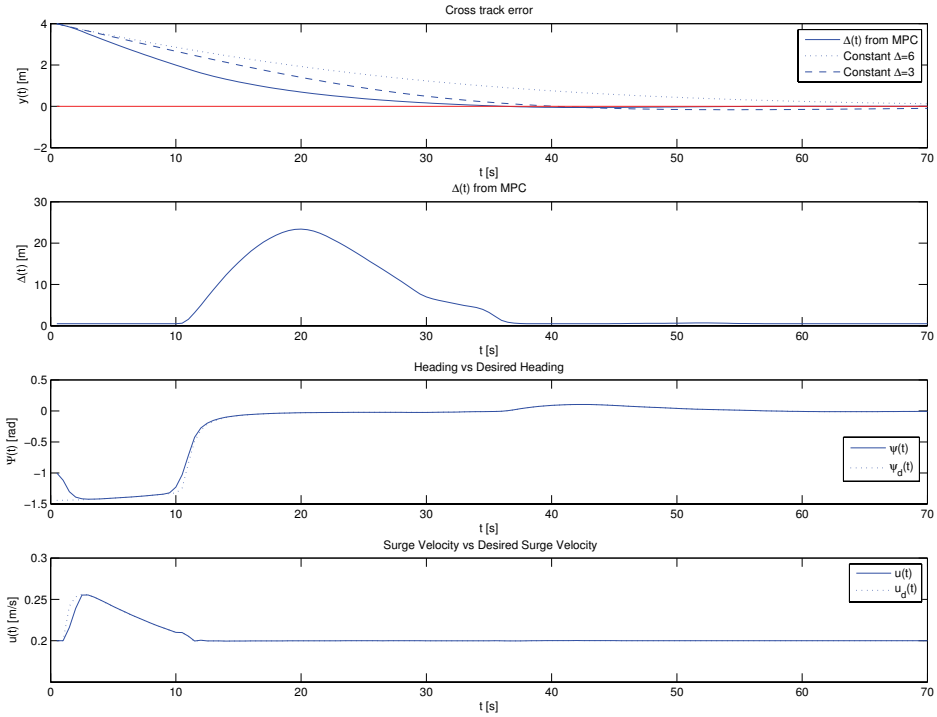

Figure 46: Simulation result for $k_{y,5}$, $k_{\overline{u}_d,1}$, $k_{\dot{u}_d,1}$, $k_{g_3,1}$ no current, initial guess U2.

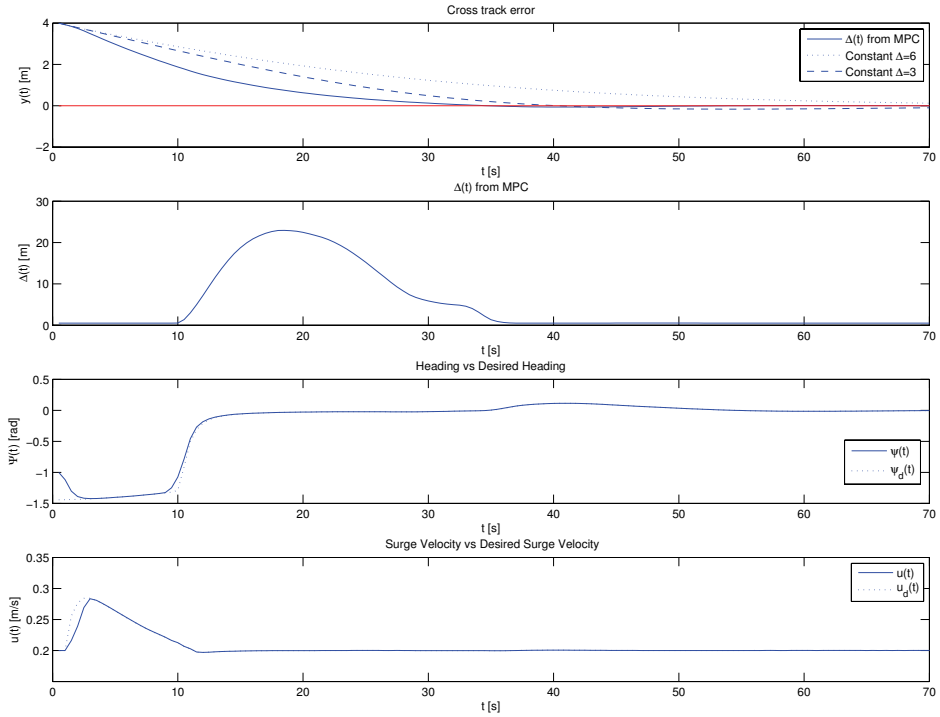

Figure 47: Simulation result for $k_{y,5}$, $k_{\overline{u}_d,2}$, $k_{\dot{u}_d,2}$, $k_{g_3,1}$ no current, initial guess U2.

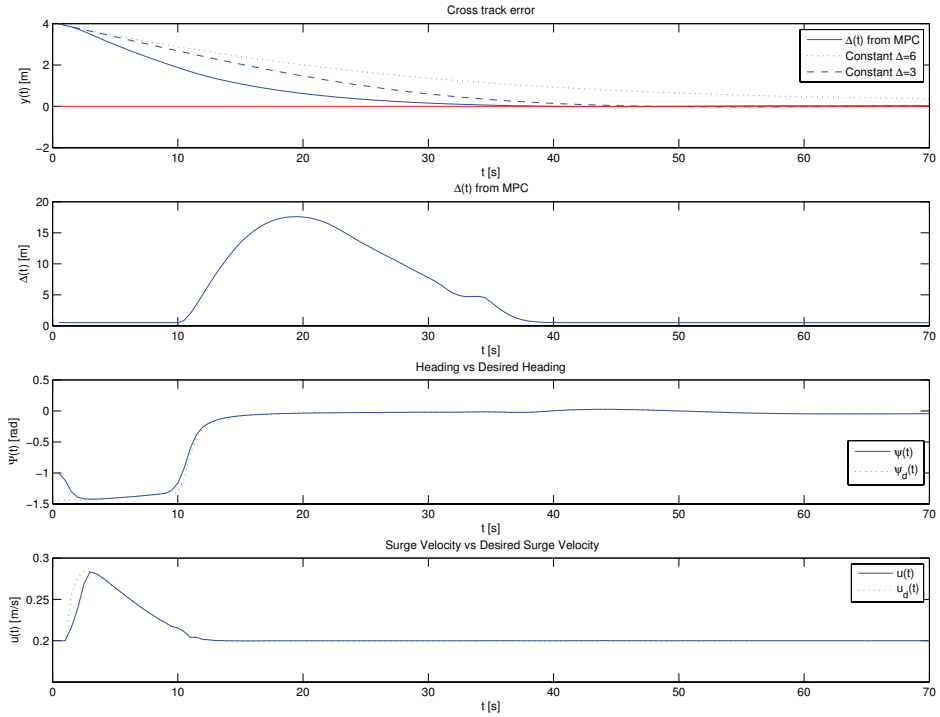

Figure 48: Simulation result for $k_{y,5}$, $k_{\overline{u}_d,2}$, $k_{\dot{u}_d,2}$, $k_{g_3,1}$ current1, initial guess U2.

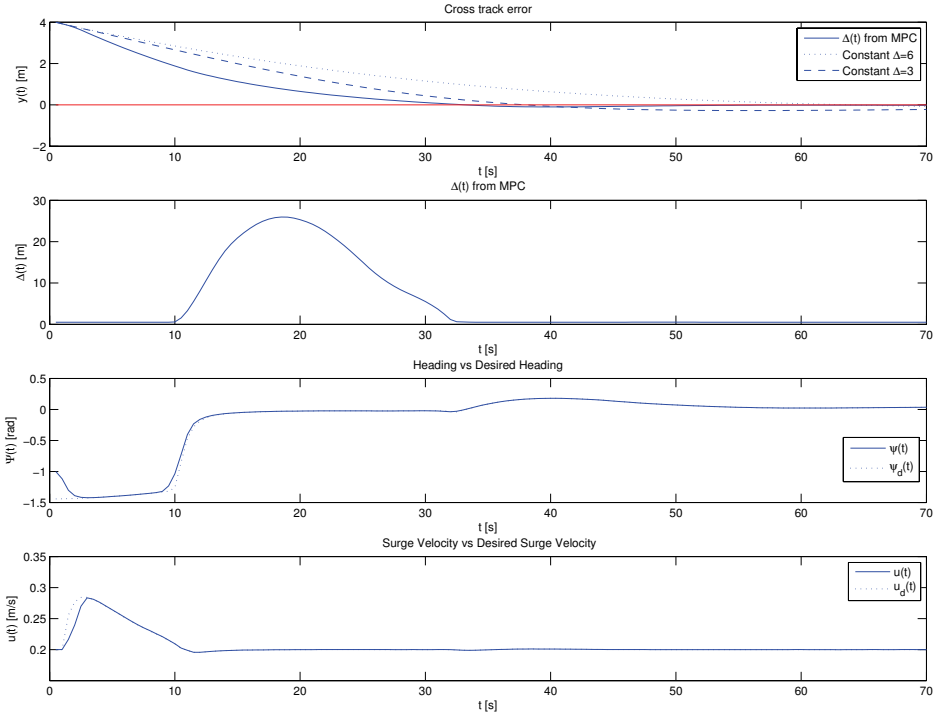

Figure 49: Simulation result for $k_{y,5}$, $k_{\overline{u}_d,2}$, $k_{\dot{u}_d,2}$, $k_{g_3,1}$ current2, initial guess U2.

8.2.3 Cost function 5

The weights k_y and $k_{\dot{y}}$ which gave the best performance when *cost-function 2* was minimized is a natural initial guess since *cost-function 5* is much the same as *cost-function 2*. The weights $k_{\overline{u}_d}$ and $k_{\dot{u}_d}$ which gave good results for *cost-function 6* will be used as a initial guess of these weights.

The parameters and weights used for the simulations of this cost-function are

$$\begin{aligned} H_p &= 48 \\ H_u &= 15 \\ T_s &= 0.5 \\ k_{y,5} &= 3 \\ k_{\dot{y},3} &= 160 \\ k_{\overline{u}_d,1} &= 20 \\ k_{\overline{u}_d,2} &= 10 \\ k_{\dot{u}_d,1} &= 3 \\ k_{\dot{u}_d,2} &= 15 \\ k_{g_3,4} &= 0.0001 \\ k_{g_3,5} &= 0.008 \\ g_{1,max_1} &= 100 \\ g_{1,max_2} &= 50 \end{aligned}$$

The results in Figure 50 indicates that this problem formulation is also non-convex, that is at least not convex for all values of the look-ahead distance. This is seen from the quite large non-zero value of the cross-track error after the over-shoot, and from the look-ahead distance staying large. When the cost-function is evaluated for the state of the system at time $t = 70$ seconds, the cost-function has a slight increase for a small reduction of the look-ahead distance, while it decreases for a large reduction of the look-ahead distance. Thus it is not convex for large values of the look-ahead distance in the current state. The non-convexity of *cost-function 5* is likely the result of increased cost from $\dot{y}^2$ and reduced cost from y^2 for decreasing values of the look-ahead distance. The cost-function was inspected by executing the following commands in *Matlab* after the simulation had ended

$$\begin{aligned} cost &= objectiveF3(Uopt) \\ &= 6.7627 \\ Uoptt(1:3:3H_u) &= Uopt(1:3:3H_u) - 0.1 \\ cost &= objectiveF3(Uoptt) \\ &= 6.7663 \\ Uoptt(1:3:3H_u) &= Uopt(1:3:3H_u) - 1 \\ cost &= objectiveF3(Uoptt) \\ &= 6.7645 \\ Uoptt(1:3:3H_u) &= Uopt(1:3:3H_u) - 10 \\ cost &= objectiveF3(Uoptt) \\ &= 6.7444 \\ Uoptt(1:3:3H_u) &= Uopt(1:3:3H_u) - 50 \\ cost &= objectiveF3(Uoptt) \\ &= 6.5696 \end{aligned}$$

The commands listed above evaluates the cost-function for the optimal inputs returned by *fmincon*, and for decreasing values of the look-ahead distance. Optimal inputs $Uopt(1:1:3H_u)$ are the optimal look-ahead distances. All optimal inputs are subtracted by the same constant. The cost-function *objectiveF3* access the current state through a global variable, predicts the future states based on *Uopt/Uoptt*, evaluates the cost-function and returns the value.

Constraining the look-ahead distance to a smaller value did not move *cost-function 6* from the non-convex region. As seen in Figure 51, *cost-function 5* seems to avoid the non-convex region when the look-ahead distance is constrained to a smaller value, under the same conditions as in Figure 50. However, since the convex region has not been identified, it would be safer to chose the initial guess of the optimal look-ahead distance as in *cost-function 6* in addition to constraining the look-ahead distance. Notice that the heading in Figure 51 is not able to track its reference from time $t \approx 30$ seconds to $t \approx 33$ seconds. This is partly the result of the fast increase in the heading at time $t \approx 28$ seconds combined with the reduction of the surge velocity at time $t \approx 29$ seconds. This leads to a large heading rate which can not be reduced immediately due to the reduced surge velocity. Further more, the heading controller includes acceleration feed-forward and the oscillations in the heading reference causes this feed-forward term to be large. Inspection of the control input in yaw shows that it oscillates from the maximum positive value to the minimum negative value. The heading dynamics acts as a first order low-past filter to fast changes in the heading rate, while the heading rate dynamics acts as a first order low-pass filter to the control input. These things in combination results in the heading not being able to track its reference from time $t \approx 30$ seconds to $t \approx 33$ seconds.

Note: *The oscillations in the control input in yaw from one saturation limit to the other are not feasible. This shows that the dynamics of the control inputs should also be modeled. For an implementation on a real vessel of the MPC controllers suggested in this thesis, the control input rate should also be weighed to reduce ware and tear.*

The result of using the initial guess of the optimal look-ahead distance as in section *cost-function 6*, U_2, and constraining the look-ahead distance can be found in Figure 52. The weights are the same as in Figure 51, but the results are different. This shows that setting a smaller upper bound on the look-ahead distance alone, does not guarantee that the look-ahead distance stays in the convex-region, and that the region is different for different states of the system. Further more, it shows that the look-ahead distance in the two figures have converged to different optima. The look-ahead distance in Figure 52 has some rapid oscillations at time $t \approx 30$ seconds, while there are rapid oscillations in the surge velocity from time $t \approx 15$ to $t \approx 35$ seconds. These oscillations are undesirable and can be suppresed by increasing the weight on the look-ahead distance acceleration and the desired surge-velocity rate.

Figure 53 is the result of increased weights $k_{\dot{u}_d}$ and k_{g_3}. The oscillations in the look-ahead distance and the desired-surge velocity are removed. The cross-track error in Figure 53 reduces fast when it is large, which results from the small look-ahead distance and the large surge-velocity. The look-ahead distance increases and the surge-velocity reduces as the cross-track error reduces. This results in only a small over-shoot. This over-shoot is compensated for by reduced surge velocity prior to the over-shoot and increased surge velocity after the over-shoot, and by the fast reducing look-ahead distance.

Figure 54 results from using the same weights as in Figure 53, and by subjecting the vessel to *current1*. The cross-track error reduces fast, has a small over-shoot and converges fast to approximately zero. As the cross-error approach zero, $t \approx 37$ seconds, the look-ahead distance increases which reduces the cross-track error rate. This helps avoiding a second over-shoot since the current acts to move the vessel in positive cross-track error direction.

Figure 55 is the result from simulations where the vessel is subjected to *current2*. Notice that the surge velocity in Figure 55 reduces before the over-shoot and increases after the over-shoot, this, in combination with the small look-ahead distance, results in only a small over-shoot and fast convergence. The cross-track error in both Figures 54 and 55 converges faster and to a smaller cross-track error, than for both constant look-ahead distances.

This cost-function results in fast convergence of the cross-track error and only small over-shoots. However, the simulation time is up to 1148 seconds when 70 seconds are simulated, that is approximately sixteen times the time simulated. Which is a major draw-back of this cost-function.

NOTE *The solver is not always able to find the minimum within the maximum number of iterations for this cost-function. Which renders the solution non-optimal and results in vast calculation times. The calculation time in this section is generally much grater than for any of the simulations in the previous sections.*

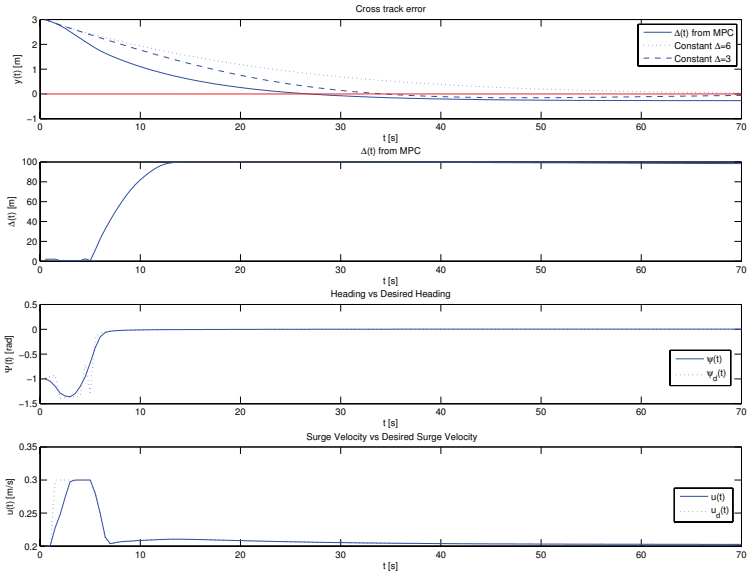

Figure 50: Simulation result for $k_{y,5}$, $k_{\dot{y},3}$, $k_{\overline{u}_d,1}$, $k_{\dot{u}_d,1}$, $k_{g_3,4}$ and g_{1,max_1}, initial guess U1.

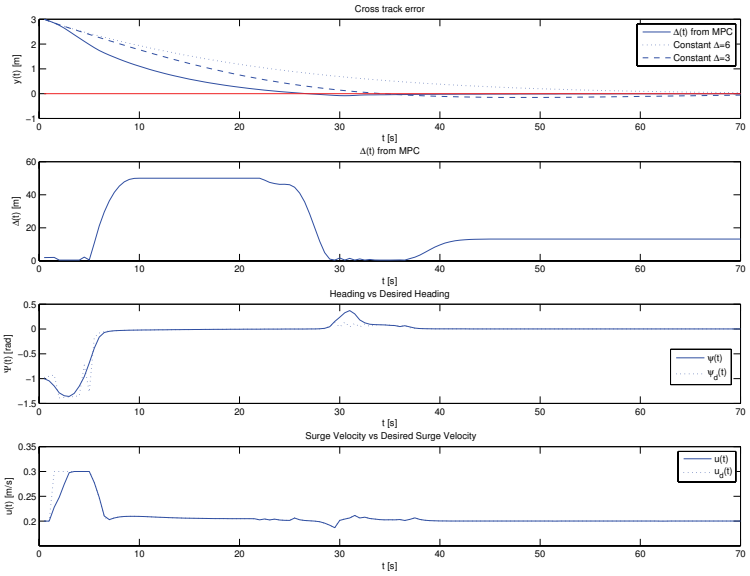

Figure 51: Simulation result for $k_{y,5}$, $k_{\dot{y},3}$, $k_{\overline{u}_d,1}$, $k_{\dot{u}_d,1}$, $k_{g_3,4}$ and g_{1,max_2}, initial guess U1.

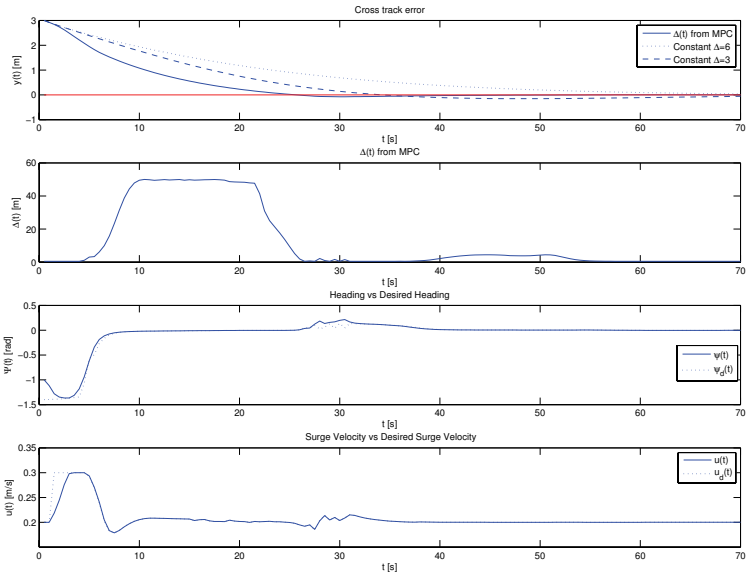

Figure 52: Simulation result for $k_{y,5}$, $k_{\dot{y},3}$, $k_{\overline{u}_d,1}$, $k_{\dot{u}_d,1}$, $k_{g_3,4}$ and g_{1,max_2}, initial guess U2.

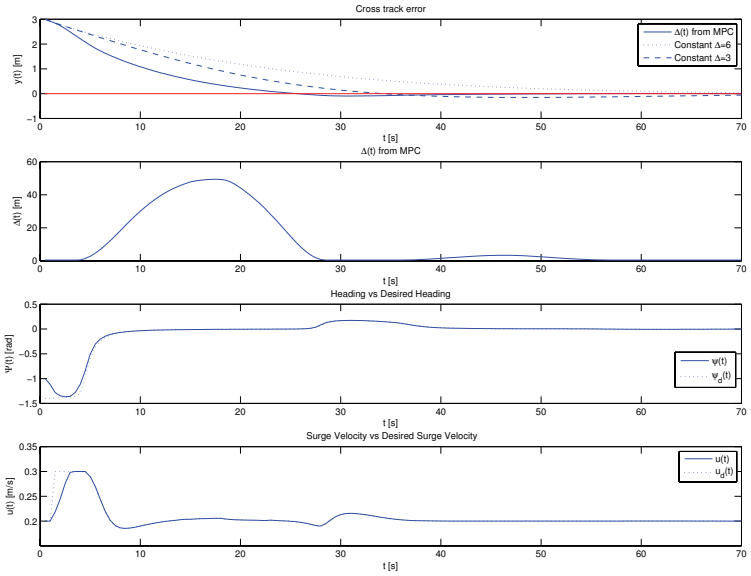

Figure 53: Simulation result for $k_{y,5}$, $k_{\dot{y},3}$, $k_{\overline{u}_d,1}$, $k_{\dot{u}_d,2}$, $k_{g_3,5}$ and g_{1,max_2}, initial guess U2.

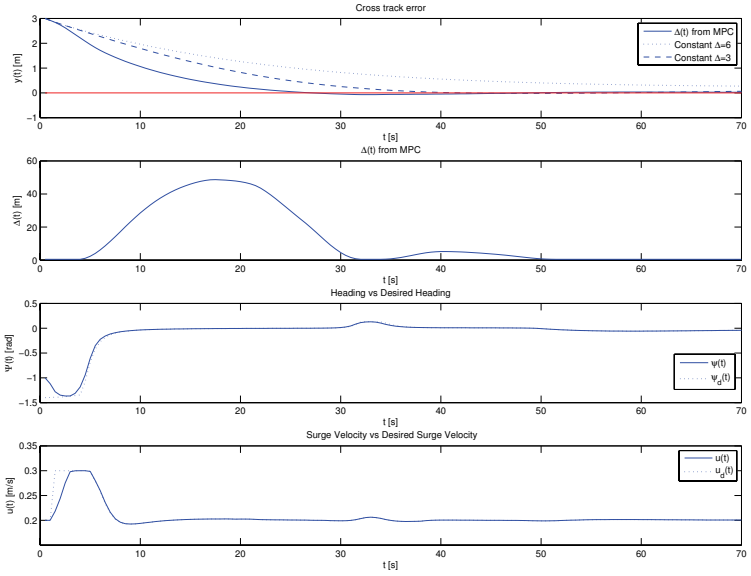

Figure 54: Simulation result for $k_{y,5}$, $k_{\dot{y},3}$, $k_{\overline{u}_d,1}$, $k_{\dot{u}_d,2}$, $k_{g_3,5}$ and g_{1,max_2}, current1, initial guess U2

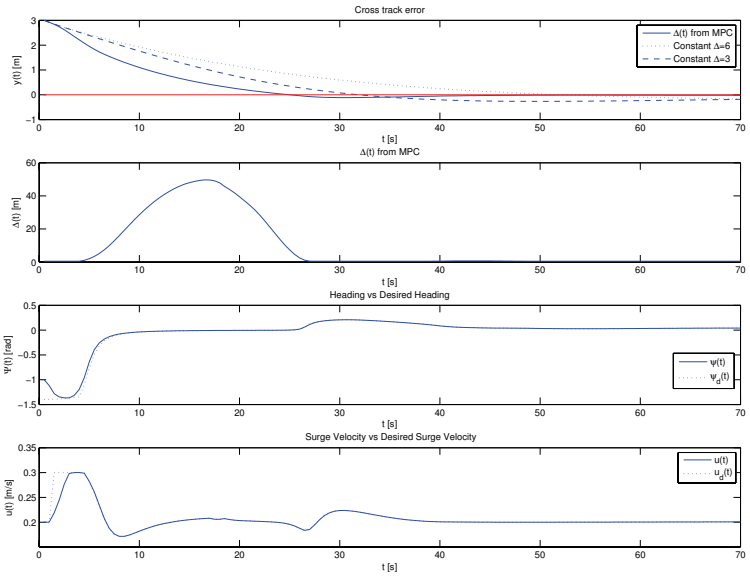

Figure 55: Simulation result for $k_{y,5}$, $k_{\dot{y},3}$, $k_{\overline{u}_d,1}$, $k_{\dot{u}_d,2}$, $k_{g_3,5}$ and g_{1,max_2}, current2, initial guess U2

8.3 Optimal path-following of several straight lines.

The main goal of this thesis was to formulate an MPC problem for path-following, incorporating existing controllers based on the LOS algorithm. This has been achieved for the two cost-functions *cost-function 2* and *cost-function 3*, and for the extended cost-functions *cost-function 5* and *cost-function 6*. The main difference in performance for these cost-functions, when they are used to track one straight line, is calculation time. If one of the MPC problems derived in this thesis is to be implemented on a marine vehicle, it has to perform better than the existing LOS algorithm where the look-ahead distance is constant, when the vehicle tracks a path made up by several straight lines. Thus, the cost-functions *2, 3, 5* and *6* will be used in simulations where the underactuated vessel tracks the path in Figure 56. The vessel is subjected to *current1* in this section. The results from subjecting the vessel to *current2* can be found in *Appendix D*.

The simulations are performed using the following parameters

$$\begin{aligned} S_t &= 660 \\ H_u &= 15 \\ H_{p,1} &= 48 \\ H_{p,2} &= 45 \\ T_s &= 0.5 \end{aligned}$$

where S_t is the time simulated. In this section the positions of the vessel resulting from the look-ahead distance returned by the optimizer can be found as the solid blue lines, while the positions of the vessel resulting from the two constant look-ahead distances can be found as the dashed lines and the dotted lines.

The way-points used to construct the path are

$$w_p = \begin{bmatrix} 0 & 0 \\ 12 & 18 \\ 27 & 27 \\ 27 & 48 \\ 21 & 54 \\ 12 & 54 \\ 0 & 48 \\ -3 & 39 \\ -3 & 30 \\ 3 & 21 \\ 12 & 21 \\ 21 & 27 \end{bmatrix}$$

Note *The steps in the heading and the cross-track error in the figures in this section occurs when a way-point is reach and a new line is tracked. This step does not occur in the physical heading. The heading seen in the figures is the heading in the rotated reference frame, thus the heading angle in the figures is relative to the angle of the path. The cross-track error in the figures is not the y-coordinate of the inertial reference frame but the y-coordinate of the rotated reference frame. Hence, no states contains step changes. However, the heading generally does increase rapidly after a new way-point is accepted, though the increase appears steeper than it is due to the long simulations time in the figures.*

Note *The initial choice U_2 is used for simulations of* cost-function 2 *and* cost-function 3 *in this section, since this choice is safer than the choice U_1 made in previous simulations with these cost-functions. The upper limit for the look-ahead distance is set to 100 meters instead of 50 meters.*

8.3.1 Cost-function 2

This cost-function resulted in better path-following than for the two constant look-ahead distances in the case of a single straight line path, both in the presence of a constant irrotational current, and without disturbances. The simulation time was approximately 300 seconds when 70 seconds

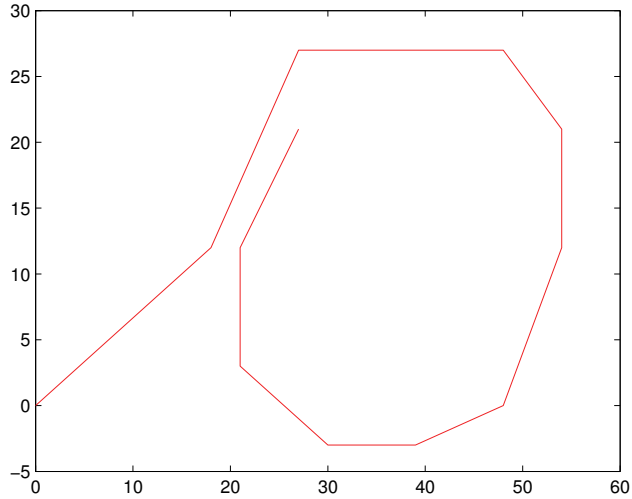

Figure 56: Path used to investigate the performance of the different cost-functions.

was simulated. The time simulated when the vessel tracks the path in figure 56 is 660 seconds, while the resulting simulation time when this cost-function is used is approximately 2000 seconds. Thus, the time-limits for finding input k are not met. However, since the simulation time can be reduced as discussed in section *Further work*, it may be possible to reduce the calculation time so that time-limits are met.

A good measure for how close the vessel tracks the path is the sum of the squared cross-track errors. The resulting squared sums for the simulation where *current1* was used are listed below

$$\sum_{k=1}^{\frac{S_t}{T_s}} y_{1,k}^2 = 362.7535 \tag{8.3.1}$$

$$\sum_{k=1}^{\frac{S_t}{T_s}} y_{2,k}^2 = 304.7897 \tag{8.3.2}$$

$$\sum_{k=1}^{\frac{S_t}{T_s}} y_{3,k}^2 = 254.9836 \tag{8.3.3}$$

where $y_{1,k}$ results from the constant look-ahead distance $\Delta = 6$ meters, $y_{2,k}$ results from the constant look-ahead distance $\Delta = 3$ meters, and $y_{3,k}$ results from the look-ahead distance returned from the optimizer. The sum 8.3.3 shows that the vessel is closer to the path for the look-ahead distance returned by the optimizer than for the two constant look-ahead distances. This is also seen in Figures 57-59. The results have been divided into the two Figures 58 and 59 so that the results can be seen better. Notice that the vessel avoids or reduces the over-shoot by cutting the corners between two successive lines of the path in Figure 57. This is due to the fast increase of the look-ahead distance after a way-point is accepted, as seen in Figures 58 and 59. The cross-track error in Figures 58 and 59 converges faster to a smaller value for the look-ahead distance returned by the optimizer than for the two constant look-ahead distances.

The sums of the squared cross-track errors for *current2* are

$$\sum_{k=1}^{\frac{S_t}{T_s}} y_{4,k}^2 = 365.7408 \tag{8.3.4}$$

$$\sum_{k=1}^{\frac{S_t}{T_s}} y_{5,k}^2 = 302.7394 \tag{8.3.5}$$

$$\sum_{k=1}^{\frac{S_t}{T_s}} y_{6,k}^2 = 248.6044 \tag{8.3.6}$$

where $y_{4,k}$ results from the constant look-ahead distance $\Delta = 6$ meters, $y_{5,k}$ results from the constant look-ahead distance $\Delta = 3$ meters, and $y_{6,k}$ results from the look-ahead distance returned from the optimizer. The sum 8.3.6 shows that the vessel is closer to the path for the look-ahead distance returned by the optimizer than for the two constant look-ahead distances, and is slightly smaller than the sum 8.3.3.

This cost-function results in better path-following of the path in Figure 56, for a pre-tuned set of constant weights, than achieved by using constant look-ahead distances.

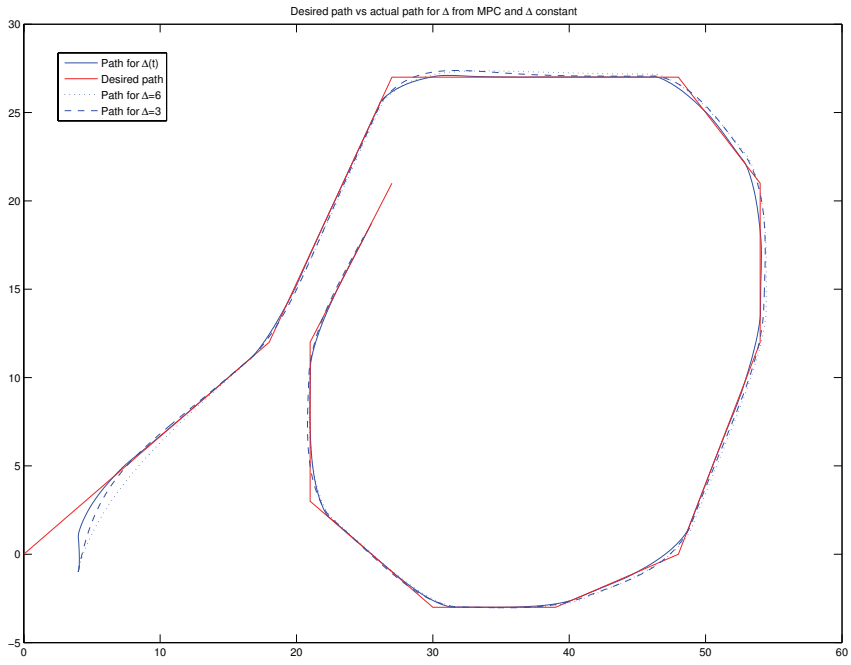

Figure 57: Position of the vessel for *cost-function 2* when the vessel is subjected to *current1*.

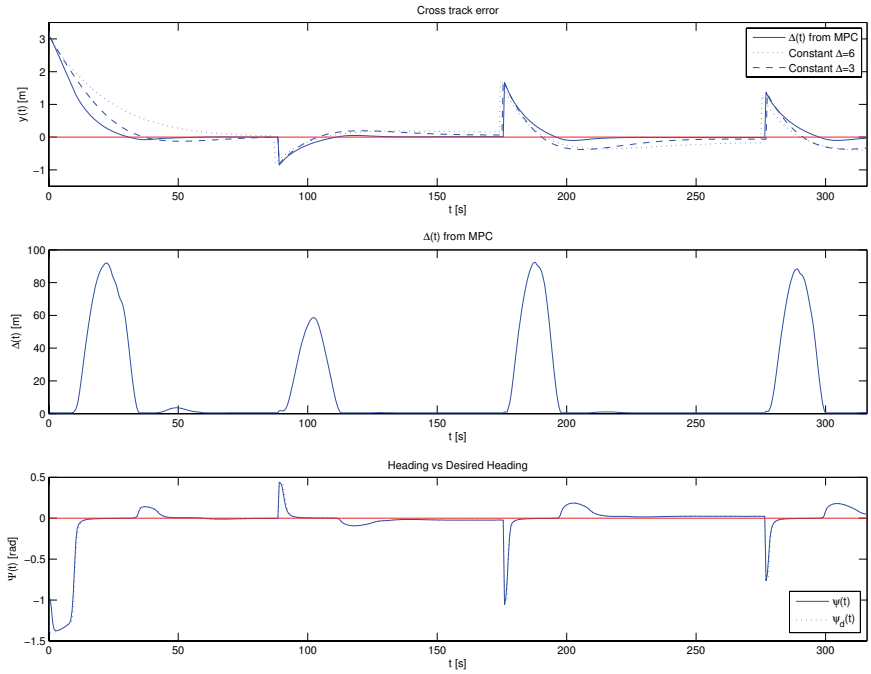

Figure 58: States from time $t = 0$ to $t = 316$ seconds, *cost-function 2*, the vessel is subjected to *current1*.

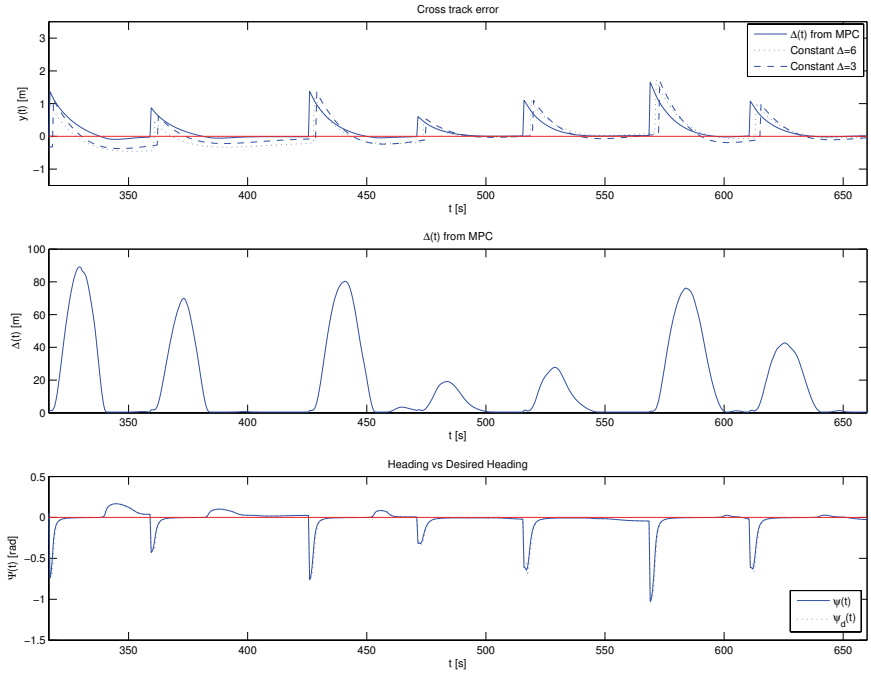

Figure 59: States from time $t = 316.5$ to $t = 660$ seconds, *cost-function 2*, the vessel is subjected to *current1*.

8.3.2 Cost-function 3

This cost-function resulted in the shortest calculation time when it was used to track a single straight line. When this cost-function is used to track a path consisting of several straight lines, the simulation runs faster than the actual time simulated. That is, the vessel is simulated for 660 seconds while the simulation lasts 463.26 seconds when the vessel is subjected to *current1*. The time from the simulation starts to the simulation terminates is measured by calling the *Mathlab* functions *tic* and *toc*. This cost-function also resulted in better path-following than for the two constant look-ahead distances in the case of a single straight line path, both in the presence of a constant irrotational current, and without disturbances.

The sum of the squared cross-track errors for this cost-function where *current1* was used is

$$\sum_{k=1}^{\frac{S_t}{T_s}} y_{7,k}^2 = 258.7872 \tag{8.3.7}$$

where $y_{7,k}$ results from the look-ahead distance returned from the optimizer. As seen from the sums 8.3.1-8.3.2 and 8.3.7, the look-ahead distance returned from the optimization problem results in smaller deviation from the path than for the two constant look-ahead distances. This is also seen in Figures 60-62 where the cross-track error converges faster and stays closer to the path for every sub-path of the path in Figure 56. Notice that the over-shoots in the cross-track error causes fast reduction of the look-ahead distance, which causes the heading to have one top for each sub-path. This is necessary to achieve fast convergence to the path. Also notice the step in the look-ahead distance when a way-point is excepted, i.e. at time $t \approx 610$ seconds in Figure 62. These steps are caused by a function which is implemented to ensure that the heading reference does not have a step change in the transition between way-points, as discussed in section *Way-points*. Also notice that the look-ahead distance resulting from *cost-function 2* increases to a larger value after way-points are accepted than the look-ahead distance resulting from *cost-function 3*, i.e at time $t \approx 325$ seconds in Figures 59 and 62, while the cross-track error seems to be quite similar in terms of over-shoots. It is likely that this is because the look-ahead distance increases and decreases faster for *cost-function 3* than for *cost-function 2*, and because the reduction of $\dot{y}$ is smaller for larger values of the look-ahead distance when the look-ahead distance is large than when it is small.

Since this cost-function performs good and the simulation runs faster than the simulated time for *current1*, it would be interesting to see whether this also is true for *current2*. The resulting simulation time when the vessel is subjected to *current2* is 537.68 seconds. Which is greater than for *current1* but well within the simulated time which is 660 seconds.

The sum of the squared cross-track errors for the simulation where *current2* was used is

$$\sum_{k=1}^{\frac{S_t}{T_s}} y_{8,k}^2 = 259.7788 \tag{8.3.8}$$

where $y_{8,k}$ results from the look-ahead distance returned from the optimizer. The sum resulting from the look-ahead distance returned by the optimizer, sum 8.3.9, is smaller than the sums resulting from the two constant look-ahead distances, sums 8.3.4 and 8.3.5, only slightly larger than the sum 8.3.7, and larger than the sum 8.3.6.

This cost-function results in better path-following of the path in Figure 56, for a pre-tuned set of constant weights, than achieved by using constant look-ahead distances. The sum of the squared cross-track errors is slightly larger for this cost-function than for *cost-function 2*, however the simulation time for this cost-function is within the time simulated. Which is a very desirable property.

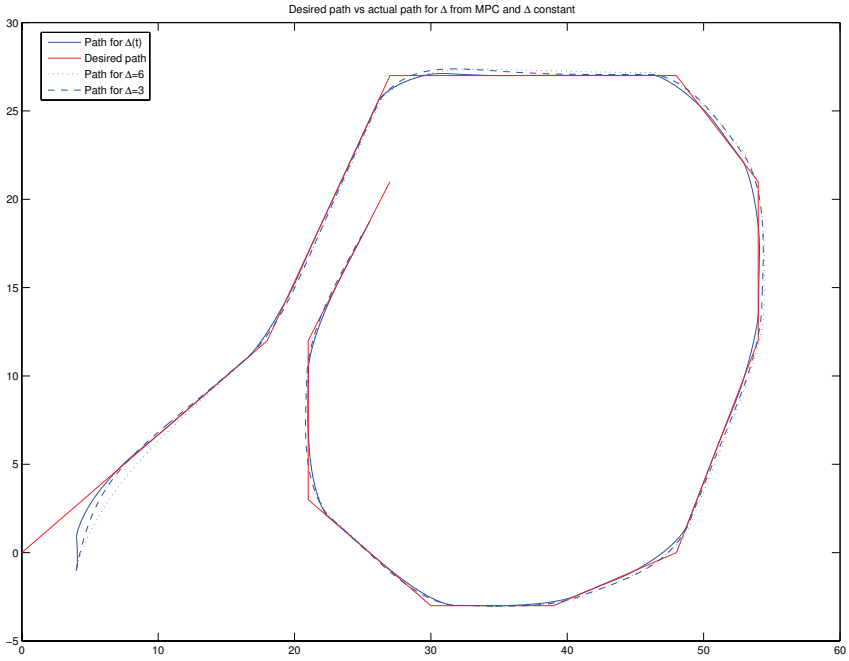

Figure 60: Position of the vessel for *cost-function 3* when the vessel is subjected to *current1*.

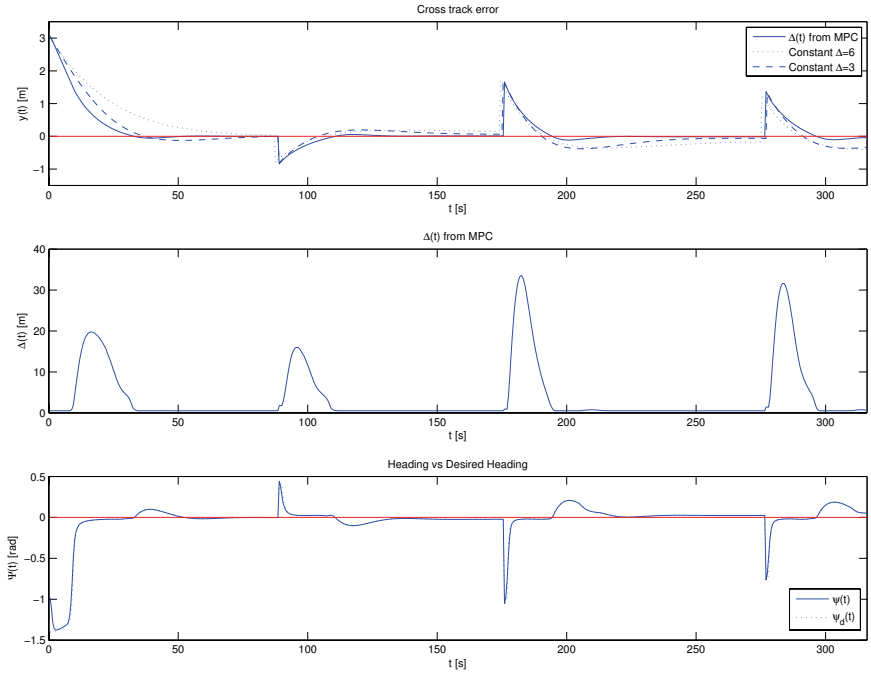

Figure 61: States from time $t = 0$ to $t = 316$ seconds, *cost-function 3*, the vessel is subjected to *current1*.

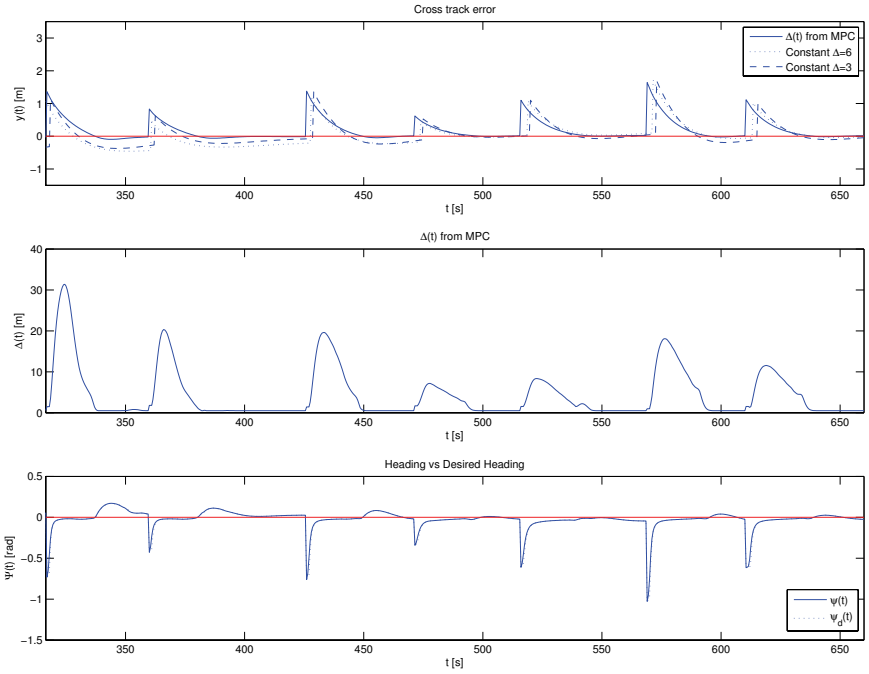

Figure 62: States from time $t = 316.5$ to $t = 660$ seconds, *cost-function 3*, the vessel is subjected to *current1*.

8.3.3 Cost-function 5

This cost function has the advantage relative to *cost-function 2* that the surge velocity can be varied. However, the calculation time is the longest of the suggested cost-functions. The simulation time is greater than 10000 seconds, when the time simulated is 660 seconds. That is, it takes almost 3 hours to simulate 11 minutes. This is a major drawback to *cost-function 5*. The vessel does track the path with small cross-track errors for this cost-function. Which can be seen in Figures 75-77, and from the sum of the squared cross-track errors resulting from the optimizer for *current1*

$$\sum_{k=1}^{\frac{S_t}{T_s}} y_{9,k}^2 = 233.6862 \tag{8.3.9}$$

Notice that the over-shoots in the cross-track error in Figures 76 and 77 are very small. This is also seen by comparing the path of the vessel at approximately coordinates [30 26] in Figure 75 with i.e. Figure 60. The smaller over-shoots in the cross-track error for this cost-function, relative to *cost-function 2* is a result of the reduced surge velocity prior to the over-shoots. The surge velocity in Figures 76 does converge to the velocity desired on the path, however the surge velocity has some fast changes, i.e at time $t \approx 515$ in Figure 77. These fast changes may not be feasible, since the rate of change in the control input is constrained on a real vessel while it is not in the simulation.

The sum of the squared cross-track errors when *current2* is used ($y_{10,k}^2$) is almost the same as for *current 1*

$$\sum_{k=1}^{\frac{S_t}{T_s}} y_{10,k}^2 = 233.9605 \tag{8.3.10}$$

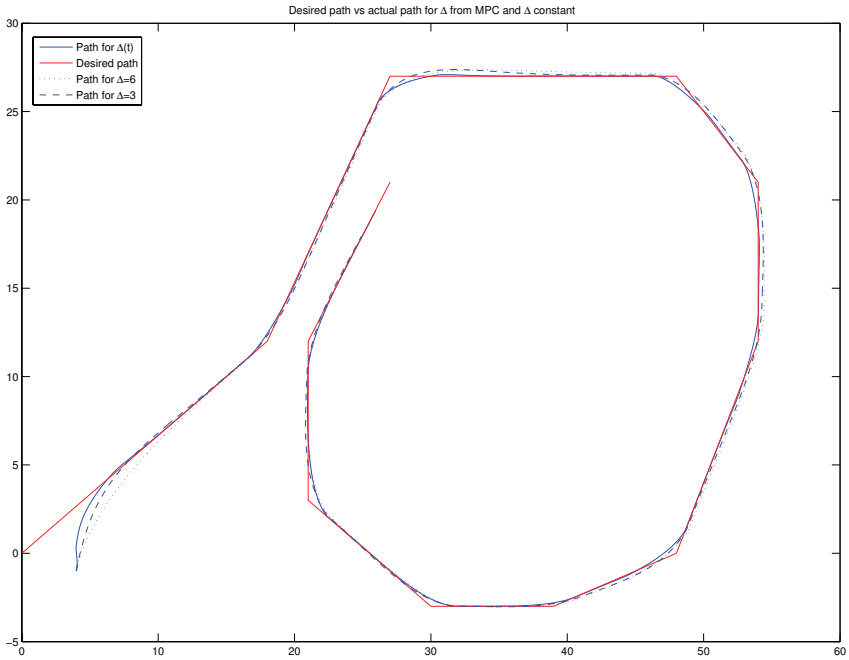

Figure 63: Position of the vessel for *cost-function 5* when the vessel is subjected to *current1*.

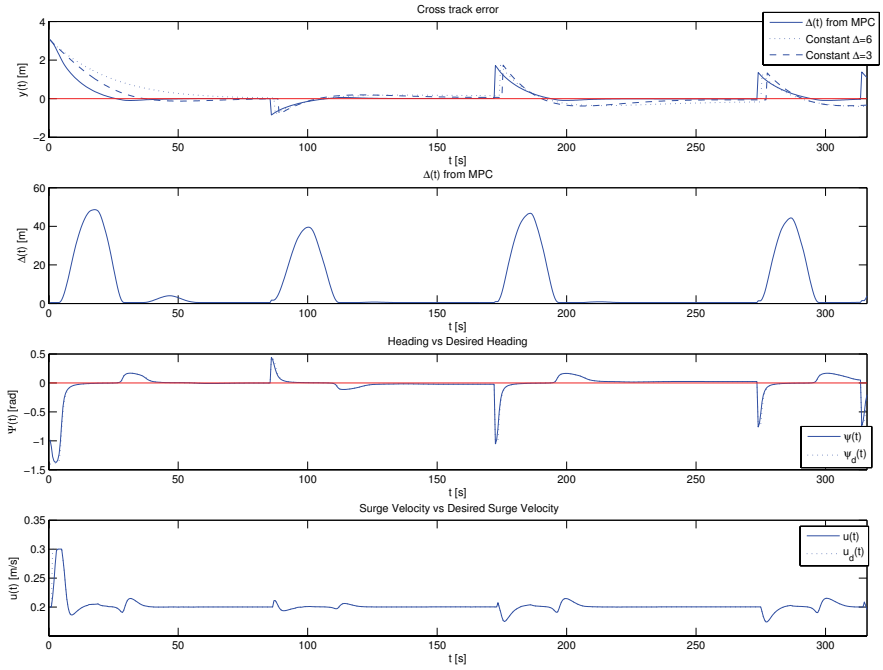

Figure 64: States from time $t = 0$ to $t = 316$ seconds, *cost-function 5*, the vessel is subjected to *current1*.

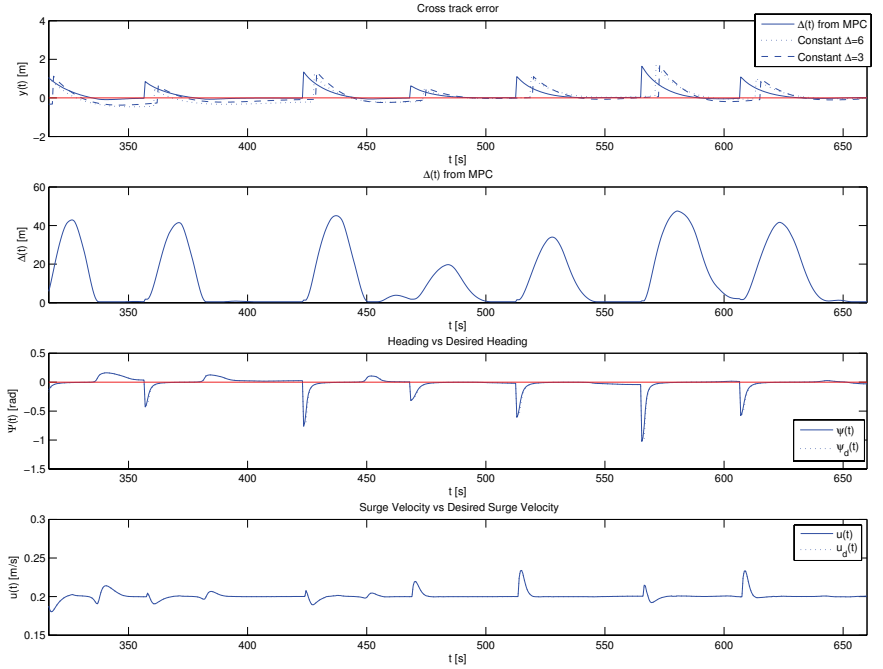

Figure 65: States from time $t = 316.5$ to $t = 660$ seconds, *cost-function 5*, the vessel is subjected to *current1*.

8.3.4 Cost-function 6

This cost-function has the advantage, relative to *cost-function 3*, of letting the desired surge velocity vary. However this comes at the cost of grater calculation time. The calculation time when this cost-function is used is approximately 3500 seconds, which is approximately 33% of the calculation time when *cost-function 5* is used, approximately nine times the calculation time when *cost-function 3* is used, and a little less than 2 times the calculation time when *cost-function 2* is used. This is also the case when these cost-functions are used when the vessel tracks the straight line path. When the vessel is subjected to *current1*, the sum of the squared cross-track errors is

$$\sum y_{10,k}^2 = 247.6981 \tag{8.3.11}$$

As seen from 8.3.11 the sum of the squared cross-track errors is smaller for this cost-function than for *cost-function 3*, when the vessel is subjected to *current1*, but larger than for *cost-function 5*. The position of the vessel can be found in Figure 66. Notice that the over-shoot at position [31 26] is smaller in Figure 66 than in Figure 60. This is again a result of the ability to reduce the surge velocity to reduce or avoid over-shoot. The surge velocity in Figures 67 and 68 stays closer to the desired surge velocity on the path than the surge velocity resulting from *cost-function 5*, which may be the reason why the vessel stays closer to the path for *cost-function 5* than for *cost-function 6*. It is possible that the performance of *cost-function 6* may be improved for choices of the weights $k_{\overline{u}_d}$ and $k_{\dot{u}_d}$ which allows the surge velocity to vary more. However, since *cost-function 2* resulted in smaller cross-track errors than *cost-function 3*, it is likely that *cost-function 5* will perform better in terms of the cross-track error than *cost-function 6* even for better choices of the weights $k_{\overline{u}_d}$ and $k_{\dot{u}_d}$. The cross-track error converges fast with little or no over-shoot. The over-shoots are smaller than for *cost-function 3* but similar to those for *cost-function 5*. The sum of the squared cross-track errors when *current2* is used is

$$\sum y_{11,k}^2 = 256.5299 \tag{8.3.12}$$

Again the sum 8.3.12 shows that the vessel is closer to the path when the look-ahead distance returned from MPC is used instead of the constant look-ahead distances. Notice that the sum of the squared cross-track errors for *current2* for *cost-function 6* is closer to the corresponding sum for *cost-function 3* than when *current1* is used. This indicates that the gain in extending *cost-function 3* to include time variant desired surge velocity might not be that great, considering the vast increase in calculation time. However, as discussed, it is possible that the results for *cost-function 6* may be improved.

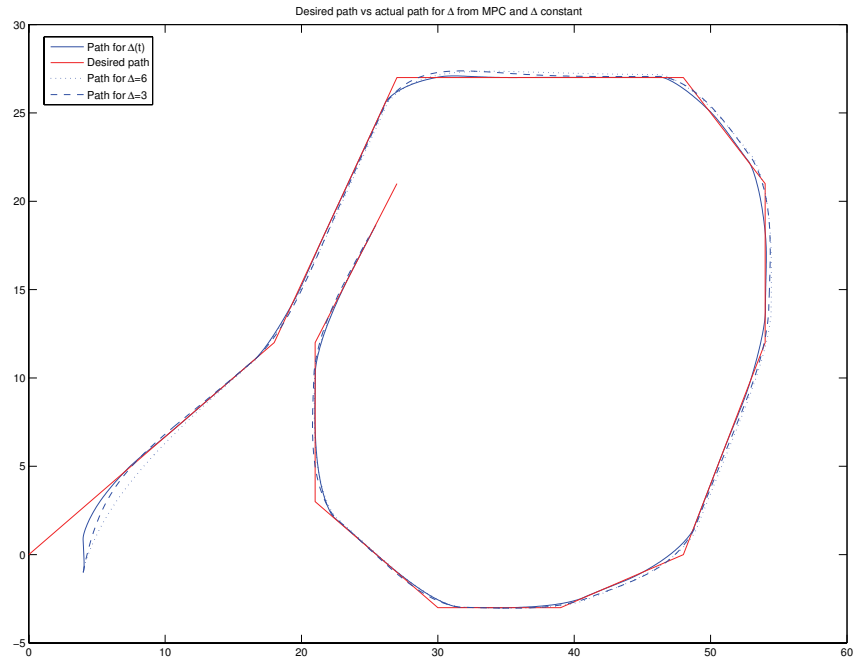

Figure 66: Position of the vessel for *cost-function 6* when the vessel is subjected to *current1*.

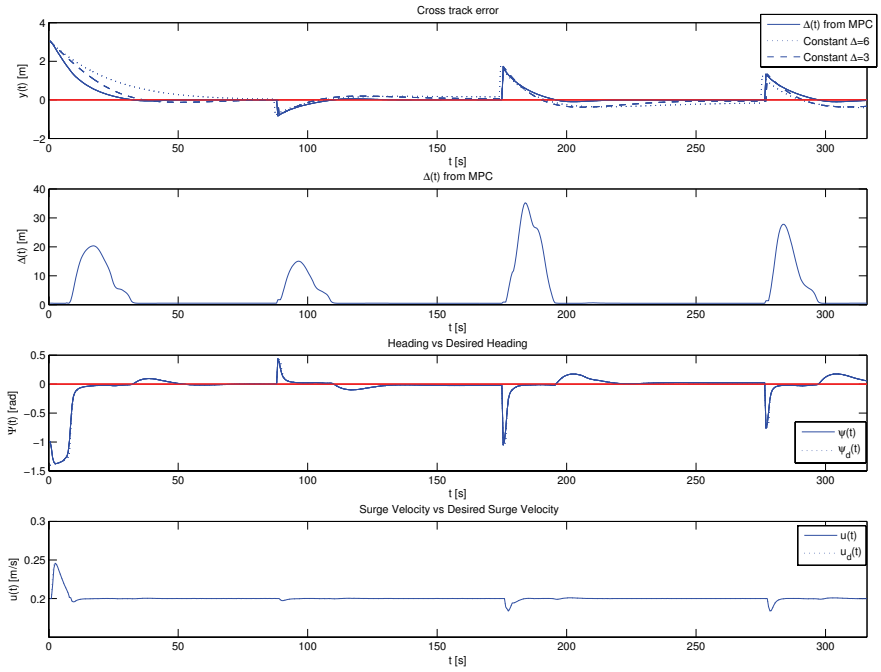

Figure 67: States from time $t = 0$ to $t = 316$ seconds, *cost-function 6*, the vessel is subjected to *current1*.

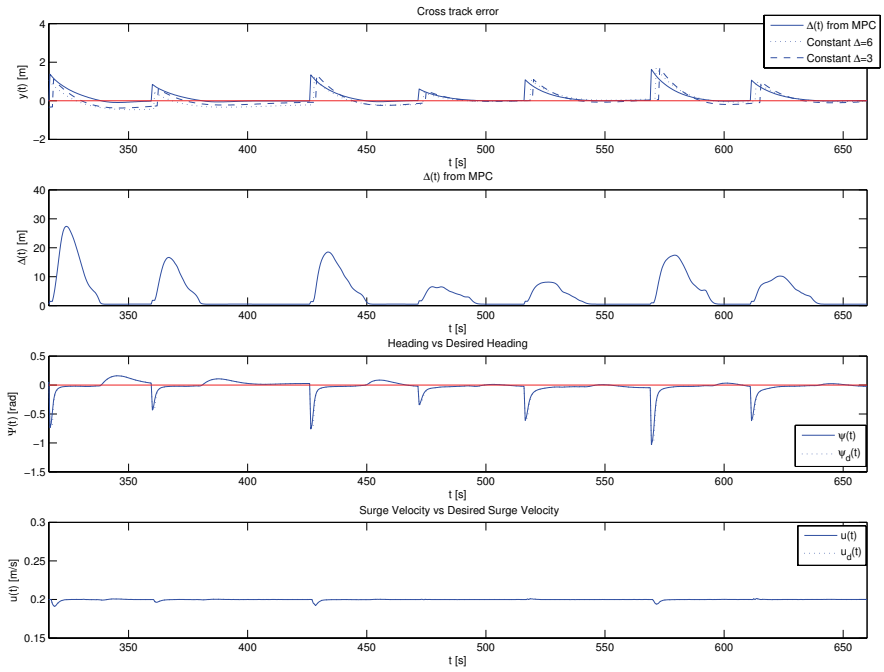

Figure 68: States from time $t = 316.5$ to $t = 660$ seconds, *cost-function 6*, the vessel is subjected to *current1*.

8.3.5 Tuning rules

In general one should expect that increasing the weight on one state-deviation should result in faster convergence of this state. However, it is important to keep in mind that some of the cost-functions are formulated as the sum of squared state deviations. This does not mean that the lower the contribution of one state deviation to the cost-function is, the nicer this state converges. This is because the lowest cost may be where a state reduces as fast as the dynamic of the state allows, but over-shoots significantly. This is the case for the cross-track error, which is the state we want to converge fast and exponentially to zero. Figure 20 shows the results from minimizing $\sum k_y y_k^2$. It is therefore necessary to find weights which causes the cost-function to have its minimum close to desired behavior of the cross-track error.

The simulations leading to the weights presented in this report are not all included since presenting every simulation performed to find proper weights would take too much space. However, some tips to find proper weights will be presented in the following.

If the look-ahead distance becomes too large when it increases to avoid over-shoot in the cross-track error, the result is that the look-ahead distance does not become small soon enough after the over-shoot. If this problem is experienced, reduce the upper limit of the look-ahead distance and/or increase the weights on the changes in the look-ahead distance and its derivatives. The latter may not always result in good performance since it will also increase the cost of reducing the look-ahead distance.

It is generally a good idea to start with small weights on the change in inputs, or on the derivatives of the inputs if the nonlinear solver is used, relative to the weights on the states. This is because the weights on the inputs should be used to reduce oscillations in the inputs, except for the weight $k_{\overline{u}_d}$ which is used to ensure that the surge velocity converges to the desired surge velocity on the path. In cost-function formulation *cost-function 1* the following rules for tuning can be used

- *Over-shoot can be reduced by increasing the weight on the heading relative to the weight on the cross-track error.*
- *Increased convergence rate of the cross-track error can be achieved by reducing the weight on the heading relative to the weight on the cross-track error, possibly at the cost of over-shoot.*
- *Unwanted oscillations in the look-ahead distance and the heading can be reduced by increasing the prediction horizon, or by adding a weight on the heading rate. The best effect is from a combination of both increasing the prediction horizon and including a weight on the heading rate.*

Note *Good performance in the general case for the same set of constant weights has not been achieved for* cost-function 1*, i.e. good performance in the presence of one specific current does not imply good performance in the presence of a current heading in the opposite direction.*

Note *The LTV model approach was only successful for* cost-function 1*.*

In cost-function formulation *cost-function 2* the following rules for tuning can be used

- *Over-shoot in the cross-track error can be reduced by increasing the weight on the cross-track error rate relative to the weight on the cross-track error.*
- *Increased convergence rate of the cross-track error can be achieved by reducing the weight on the cross-track error rate relative to the weight on the cross-track error, possibly at the cost of increased over-shoot.*
- *Unwanted oscillations in the look-ahead distance can be reduced by increasing the weight on the look-ahead distance acceleration.*

In cost-function formulation *cost-function 3* the following rules for tuning can be used

- *Over-shoot in the cross-track error can be reduced by reducing the parameter* k_y
- *Increased convergence rate of the cross-track error can be achieved by increasing* k_y.
- *Unwanted oscillations in the look-ahead distance can be reduced by increasing the weight on the look-ahead distance acceleration.*

Tuning the weights for *cost-function 3* is easier than for the other cost-functions since the tuning of the two parameters can be done quite independent.

If *cost-function 5* or *cost-function 6* is to be tuned after weights for *cost-function 2* or *cost-function 3* have been obtained, these weights should be used as a starting points. The tuning is performed using the same rules as for *cost-function 2* or *cost-function 3*. The weight $k_{\overline{u}_d}$ is increased if the surge velocity is not close enough to the desired velocity on the path, while it is reduced if the surge velocity does not deviate from the path when this is desirable. Oscillations in u_d are reduced by increasing $k_{\dot{u}_d}$, while reducing $k_{\dot{u}_d}$ allows faster changes in the surge velocity. If the weight $k_{\overline{u}_d}$ is small and the surge velocity still stays at the desired velocity on the path, try reducing $k_{\dot{y}_d}$.

9 Conclusion

Six cost-functions have been suggested to find the optimal look-ahead distance, and in some cases the optimal surge velocity reference. The general performance in terms of path following, relative to the existing constant look-ahead distance approach, is improved for all suggested cost-functions except for *cost-function 1* and *cost-function4*. That is, all of the suggested cost-functions 2, 3, 5 and 6 results in path following with faster convergence of the cross-track error, smaller over-shots in the cross-track error and with smaller cross-track errors, than for the constant look-ahead distances. This is true in the case of no disturbances and in the presence of a constant irrotational current. Both when the vessel tracks a straight line path and when the vessel tracks a path consisting of several straight lines for constant pre-tuned weights. This is a very desirable property. That is, weights tuned to achieve good performance when the vessel tracks a single straight line, also results in good performance when the vessel tracks a path consisting of several straight lines. This is a necessary property if a controller based on one of the suggested cost-functions is to be implemented on a real vessel since the controller has to be able to track any path after being tuned.

A good measure of the performance, in terms of the cross-track error, of the cost-functions and the constant look-ahead distances, is the sum of the squared cross-track errors. The sums resulting from cost-functions 2, 3, 5 and 6 are all smaller than the sums resulting from the constant look-ahead distances. When the vessel tracks a path consisting of several straight lines, the smallest sum for the constant look-ahead distances is approximately 303, while the largest sums for cost-functions 2, 3, 5 and 6 are approximately 255, 260, 234 and 256 respectively. That is, *cost-function 5* results in the smallest cross-track errors. However, only *cost-function 3* results in calculation times close to the time limits for producing the next inputs, while *cost-function 5* results in the largest calculation times, approximately 15 times the time limits. The tuning process of *cost-function 3* is quite easy.

The main trade-offs that has to be made are between calculation time and convergence rate of the cross-track error, and between fast reduction of the cross-track error and over-shoot.

The LTV model developed in this thesis is accurate enough for *cost-function 1*. However, this is not the case when the cross-track error rate is included in the LTV model. Hence *cost-function 1* is the only cost-function where using the LTV model gives good results. That is, weights which yield good performance in certain cases can be found, however the same set of weights does not yield good performance in the general case. The inaccuracy of the LTV model motivates for solving the optimization problems with a non-linear optimization problem solver. The gain in this approach is increased accuracy, however this is at the cost of increased calculation times, and no guarantee of convexity.

The cost-functions 2, 3, 5 and 6 are not convex for all values of the look-ahead distance, which may cause problems. However, a measure to stay in the convex region is presented, that is, a certain choice of starting points for the optimizer resolves this problem. None of the problems caused by the non-convexity of the cost-functions are experienced when this measure is taken.

9.1 Further work

An improved LTV model for the cross-track error rate $\dot{y}$ is needed since the LTV model used for this state in this thesis is too inaccurate. Using an LTV model for predictions is desired because the optimization problem is convex when the system is modeled linear time variant, and because the calculation time is in general shorter. The calculation time when *cost-function 5* and *cost-function 6* is minimized has to be reduced to make the use of these cost-functions feasible. It is desirable to use these cost-functions because they have the desirable property of being able to vary the desired surge velocity.

The assumptions of perfect knowledge of the vessel model and of unconstrained control input rates has to be relaxed since these assumptions will not hold. Thus, the performance in the case of an

imperfect prediction model should be investigated by inserting reasonable inaccuracies in the prediction model and including the actuator dynamics in the vessel dynamics used for simulations. The performance in the case of an imperfect vessel model can also be investigated by implementing the suggested MPC approach on a real vessel such as *Cybership 2*.

The physical constraints on actuators and states should be included in the optimization problem since MPC is well suited for such constraint handling.

As little use of the actuators as possible is generally desirable. This is because excessive use of the actuators causes unnecessary wear and tear, and increased fuel consumption. Thus, the possibility of including the control inputs and the control input rate in the cost-functions should be investigated.

Work should be put into proving stability for the time variant look-ahead distance and desired surge velocity. Any constraints for stability to hold should be identified.

The convex region of the cost-functions should be identified so that a measure to guarantee convergence to the global optimum can be identified.

10 References

References

[1] Ivar-André F. Ihle, Roger Skjetne and Thor I. Fossen *Output Feedback Control for Maneuvering Systems Using Observer Backstepping*
Proceedings off the 13th Mediterranean Conference on Control and Automation Limassol, Cypros, June 27-20, 2005

[2] E. Fredriksen, K.Y. Pettersen *Global κ-exponential way-point maneuvering of ships:Theory and experiments*

[3] Even Børhaug and Kristin Y. Pettersen *Cross-track control for underactuated autonomous vehicles*
Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005 Seville, Spain, December 12-15, 2005

[4] Even Børhaug, Kristin Y. Pettersen and Alexey Pavlov *An optimal guidence scheme for cross-track control of underactuated underwater vehicles*

[5] W. Naeem *Model predictive control of an autonomous underwater vehicle*
in proc. UKACC Conference on Control, 2002, pp. 19-23

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

[7] Erik Kyrkjebø and Kristin Y. Pettersen *Leader-Follower output reference state feedback synchronization control of Euler-Lagrange systems*

[8] Alexey Pavlov, Even Børhaug, Elena Panteley and Kristin Y. Pettersen *Exponential Formation Control of Underactuated Surface Vessels*

[9] Jorge Nocedal and Stephen J. Wright *Numerical Optimization*
1999,

[10] J. M. Maciejowski *Predictive Control: with Constraints*
2002

[11] Thor I. Fossen *Marine Control Systems, Guidence, Navigation, and Control of ships, Rigs and Underwatre Vehicles*
2002, 3rd printing

[12] Do, K. D., & Pan, J. *Global tracking control of underactuated ships with off-diagonal terms.*
Proceedings of the 42nd IEEE conference on decision and control (pp. 1244-1249)
2003

[13] Roger Skjetne, Øyvind Smogeli and Thor I. Fossen *Modeling, Identification, and Adaptive Maneuvering of Cybership 2: A Complete Design With Experiments*

[14] Marafioti, Giancarlo (Norwegian University of Science and Technology), Bitmead, Robert (University of California San Diego), Hovd, Morten (Norwegian University of Technology and Science) *Model Predictive Control with State Dependent Input Weight: An Application to Underwater Vehicles*
17th IFAC World Congress, July 6-11, 2008, Seoul, Korea[6]

[15] Mathworks support 1 *http://www.mathworks.com/support/solutions/data/1-1BYP9.html?product=OP&solution=1-1BYP9*

[16] Mathworks support 2 *http://www.mathworks.com/support/solutions/data/1-1A484.html?product=OP&solution=1-1A484*

[6] Scheduled for presentation during the Regular Session "Marine System II" (FrB28), Friday, July 11, 2008

A Appendix: Analytical Φ_k and Γ_k

In the case of u_d constant and u assumed equal to u_d the matrices 4.2.66 and 4.2.67 for our system are

$$\Phi_k = I + T_s \begin{bmatrix} \frac{\partial f_1(x,u)}{\partial y} & \frac{\partial f_1(x,u)}{\partial \overline{y}} & \frac{\partial f_1(x,u)}{\partial \psi} & \frac{\partial f_1(x,u)}{\partial v} & \frac{\partial f_1(x,u)}{\partial \overline{v}} & \frac{\partial f_1(x,u)}{\partial r} \\ \frac{\partial f_2(x,u)}{\partial y} & \frac{\partial f_2(x,u)}{\partial \overline{y}} & \frac{\partial f_2(x,u)}{\partial \psi} & \frac{\partial f_2(x,u)}{\partial v} & \frac{\partial f_2(x,u)}{\partial \overline{v}} & \frac{\partial f_2(x,u)}{\partial r} \\ \frac{\partial f_3(x,u)}{\partial y} & \frac{\partial f_3(x,u)}{\partial \overline{y}} & \frac{\partial f_3(x,u)}{\partial \psi} & \frac{\partial f_3(x,u)}{\partial v} & \frac{\partial f_3(x,u)}{\partial \overline{v}} & \frac{\partial f_3(x,u)}{\partial r} \\ \frac{\partial f_4(x,u)}{\partial y} & \frac{\partial f_4(x,u)}{\partial \overline{y}} & \frac{\partial f_4(x,u)}{\partial \psi} & \frac{\partial f_4(x,u)}{\partial v} & \frac{\partial f_4(x,u)}{\partial \overline{v}} & \frac{\partial f_4(x,u)}{\partial r} \\ \frac{\partial f_5(x,u)}{\partial y} & \frac{\partial f_5(x,u)}{\partial \overline{y}} & \frac{\partial f_5(x,u)}{\partial \psi} & \frac{\partial f_5(x,u)}{\partial v} & \frac{\partial f_5(x,u)}{\partial \overline{v}} & \frac{\partial f_5(x,u)}{\partial r} \\ \frac{\partial f_6(x,u)}{\partial y} & \frac{\partial f_6(x,u)}{\partial \overline{y}} & \frac{\partial f_6(x,u)}{\partial \psi} & \frac{\partial f_6(x,u)}{\partial v} & \frac{\partial f_6(x,u)}{\partial \overline{v}} & \frac{\partial f_6(x,u)}{\partial r} \end{bmatrix} \tag{A.0.1}$$

$$\Gamma_k = T_s \begin{bmatrix} \frac{\partial f_1(x,g)}{\partial g_1} & \frac{\partial f_1(x,g)}{\partial g_2} & \frac{\partial f_1(x,g)}{\partial g_3} \\ \frac{\partial f_2(x,g)}{\partial g_1} & \frac{\partial f_2(x,g)}{\partial g_2} & \frac{\partial f_2(x,g)}{\partial g_3} \\ \frac{\partial f_3(x,g)}{\partial g_1} & \frac{\partial f_3(x,g)}{\partial g_2} & \frac{\partial f_3(x,g)}{\partial g_3} \\ \frac{\partial f_4(x,g)}{\partial g_1} & \frac{\partial f_4(x,g)}{\partial g_2} & \frac{\partial f_4(x,g)}{\partial g_3} \\ \frac{\partial f_5(x,g)}{\partial g_1} & \frac{\partial f_5(x,g)}{\partial g_2} & \frac{\partial f_5(x,g)}{\partial g_3} \\ \frac{\partial f_6(x,g)}{\partial g_1} & \frac{\partial f_6(x,g)}{\partial g_2} & \frac{\partial f_6(x,g)}{\partial g_3} \end{bmatrix} \tag{A.0.2}$$

where

$$\begin{aligned}
\frac{\partial f_1(x,u)}{\partial y} &= \frac{\partial f_1(x,u)}{\partial \overline{y}} = \frac{\partial f_1(x,u)}{\partial \overline{v}} = \frac{\partial f_1(x,u)}{\partial r} = \frac{\partial f_2(x,u)}{\partial y} = \frac{\partial f_2(x,u)}{\partial \overline{y}} \\
&= \frac{\partial f_2(x,u)}{\partial v} = \frac{\partial f_2(x,u)}{\partial r} = \frac{\partial f_3(x,u)}{\partial y} = \frac{\partial f_3(x,u)}{\partial \overline{y}} = \frac{\partial f_3(x,u)}{\partial \psi} \\
&= \frac{\partial f_3(x,u)}{\partial v} = \frac{\partial f_3(x,u)}{\partial \overline{v}} = \frac{\partial f_4(x,u)}{\partial y} = \frac{\partial f_4(x,u)}{\partial v} = \frac{\partial f_5(x,u)}{\partial y} \\
&= \frac{\partial f_5(x,u)}{\partial \overline{y}} = \frac{\partial f_5(x,u)}{\partial \psi} = \frac{\partial f_5(x,u)}{\partial v} = \frac{\partial f_6(x,u)}{\partial y} = \frac{\partial f_6(x,u)}{\partial v} \\
&= \frac{\partial f_1(x,u)}{\partial u_1} = \frac{\partial f_1(x,u)}{\partial u_2} = \frac{\partial f_1(x,u)}{\partial u_3} = \frac{\partial f_2(x,u)}{\partial u_1} = \frac{\partial f_2(x,u)}{\partial u_2} = \frac{\partial f_2(x,u)}{\partial u_3} \\
&= \frac{\partial f_3(x,u)}{\partial u_1} = \frac{\partial f_3(x,u)}{\partial u_2} = \frac{\partial f_3(x,u)}{\partial u_3} = \frac{\partial f_5(x,u)}{\partial u_1} = \frac{\partial f_5(x,u)}{\partial u_2} = \frac{\partial f_5(x,u)}{\partial u_3} \\
&= 0
\end{aligned} \tag{A.0.3}$$

$$\frac{\partial f_1(x,u)}{\partial \psi} = u_c \cos(\psi_n) - v_n \sin(\psi_n) \tag{A.0.4}$$

$$\frac{\partial f_1(x,u)}{\partial v} = \cos(\psi_n) \tag{A.0.5}$$

$$\frac{\partial f_2(x,u)}{\partial \psi} = u_c \cos(\psi_n) - v_n \sin(\psi_n) \tag{A.0.6}$$

$$\frac{\partial f_2(x,u)}{\partial \overline{v}} = \cos(\psi_n) \tag{A.0.7}$$

$$\frac{\partial f_3(x,u)}{\partial r} = 1 \tag{A.0.8}$$

$$\frac{\partial f_4(x,u)}{\partial \overline{y}} = -\epsilon\Bigg\{ \frac{2\Big[u_{1,n}\Big((u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))^2-u_{2,n}^2\Big)-2\overline{y}_n u_{2,n}(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$- \frac{8\overline{y}_n\Big[u_{2,n}(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))(u_{1,n}^2-\overline{y}_n^2)+u_{1,n}\overline{y}_n\Big((u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))^2-u_{2,n}^2\Big)\Big]}{(u_{1,n}^2+\overline{y}_n^2)^3}$$

$$+ \frac{u_{3,n}(u_{1,n}^2+\overline{y}_n^2)+2\overline{y}_n\Big[}{}$$

$$\frac{u_{1,n}\Big(r_n\cos(\psi_n)(u_c+u_c\Theta+M)+\overline{v}_n[(u_c\Xi+N)\cos(\psi_n)-r_n\sin(\psi_n)]\Big)-\overline{y}_n u_{3,n}\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$+ \frac{k_1\Big[u_{2,n}(u_{1,n}^2+\overline{y}_n^2)+2\overline{y}_n\Big(u_{1,n}(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))-\overline{y}_n u_{2,n}\Big)\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$- k_0\frac{u_{1,n}}{u_{1,n}^2+\overline{y}_n^2}\Bigg\} \tag{A.0.9}$$

$$\frac{\partial f_4(x,u)}{\partial \psi} = -\epsilon\Bigg\{ \frac{2\Big[u_{2,n}(u_c\cos(\psi_n)-\overline{v}_n\sin(\psi_n))(u_{1,n}^2-\overline{y}_n^2)}{}$$

$$\frac{+2u_{1,n}\overline{y}_n(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))(u_c\cos(\psi_n)-\overline{v}_n\sin(\psi_n))\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$+ \frac{u_{1,n}\Big[r_n\sin(\psi_n)(u_c+u_c\Theta+M)+\overline{v}_n\Big((u_c\Xi+N)\sin(\psi_n)+r_n\cos(\psi_n)\Big)\Big]}{u_{1,n}^2+\overline{y}_n^2}$$

$$- k_1\frac{u_{1,n}(u_c\cos(\psi_n)-\overline{v}_n\sin(\psi_n))}{u_{1,n}^2+\overline{y}_n^2} - k_0\Bigg\} \tag{A.0.10}$$

$$\frac{\partial f_4(x,u)}{\partial \overline{v}} = (u_c\Xi+N)$$

$$- \epsilon\Bigg\{ \frac{2\Big[\cos(\psi_n)u_{2,n}(u_{1,n}^2-\overline{y}_n^2)+2u_{1,n}\overline{y}_n\cos(\psi_n)(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$- \frac{u_{1,n}[(u_c\Xi+N)\cos(\psi_n)-r_n\sin(\psi_n)]}{u_{1,n}^2+\overline{y}_n^2} - k_1\frac{u_{1,n}\cos(\psi_n)}{u_{1,n}^2+\overline{y}_n^2}\Bigg\} \tag{A.0.11}$$

$$\frac{\partial f_4(x,u)}{\partial r} = (u_c\Theta + M) - \epsilon\Bigg\{ - \frac{u_{1,n}\Big[\cos(\psi_n)(u_c + u_c\Theta + M) - \overline{v}_n \sin(\psi_n)\Big]}{u_{1,n}^2 + \overline{y}_n^2} - k_1 \Bigg\} \tag{A.0.12}$$

$$\frac{\partial f_5(x,u)}{\partial \overline{v}} = (u_c\Xi + N) \tag{A.0.13}$$

$$\frac{\partial f_5(x,u)}{\partial r} = (u_c\Theta + M) \tag{A.0.14}$$

$$\frac{\partial f_6(x,u)}{\partial \overline{y}} = \frac{2\Big[u_{1,n}\Big((u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))^2-u_{2,n}^2\Big)-2\overline{y}_n u_{2,n}(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$-\frac{8\overline{y}_n\Big[u_{2,n}(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))(u_{1,n}^2-\overline{y}_n^2)+u_{1,n}\overline{y}_n\Big((u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))^2-u_{2,n}^2\Big)\Big]}{(u_{1,n}^2+\overline{y}_n^2)^3}$$

$$+\frac{u_{3,n}(u_{1,n}^2+\overline{y}_n^2)+2\overline{y}_n\Big[}{u_{1,n}\Big(r_n\cos(\psi_n)(u_c+u_c\Theta+M)+\overline{v}_n[(u_c\Xi+N)\cos(\psi_n)-r_n\sin(\psi_n)]\Big)-\overline{y}_n u_{3,n}\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$+\frac{k_1\Big[u_{2,n}(u_{1,n}^2+\overline{y}_n^2)+2\overline{y}_n\Big(u_{1,n}(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))-\overline{y}_n u_{2,n}\Big)\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$-k_0\frac{u_{1,n}}{u_{1,n}^2+\overline{y}_n^2} \tag{A.0.15}$$

$$\frac{\partial f_6(x,u)}{\partial \psi} = \frac{2\Big[u_{2,n}(u_c\cos(\psi_n)-\overline{v}_n\sin(\psi_n))(u_{1,n}^2-\overline{y}_n^2)}{+2u_{1,n}\overline{y}_n(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))(u_c\cos(\psi_n)-\overline{v}_n\sin(\psi_n))\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$+\frac{u_{1,n}\Big[r_n\sin(\psi_n)(u_c+u_c\Theta+M)+\overline{v}_n\Big((u_c\Xi+N)\sin(\psi_n)+r_n\cos(\psi_n)\Big)\Big]}{u_{1,n}^2+\overline{y}_n^2}$$

$$-k_1\frac{u_{1,n}(u_c\cos(\psi_n)-\overline{v}_n\sin(\psi_n))}{u_{1,n}^2+\overline{y}_n^2}-k_0 \tag{A.0.16}$$

$$\frac{\partial f_6(x,u)}{\partial \overline{v}} = \frac{2\Big[\cos(\psi_n)u_{2,n}(u_{1,n}^2-\overline{y}_n^2)+2u_{1,n}\overline{y}_n\cos(\psi_n)(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$-\frac{u_{1,n}[(u_c\Xi+N)\cos(\psi_n)-r_n\sin(\psi_n)]}{u_{1,n}^2+\overline{y}_n^2}-k_1\frac{u_{1,n}\cos(\psi_n)}{u_{1,n}^2+\overline{y}_n^2} \tag{A.0.17}$$

$$\frac{\partial f_6(x,u)}{\partial r} = -\frac{u_{1,n}\Big[\cos(\psi_n)(u_c + u_c\Theta + M) - \overline{v}_n \sin(\psi_n)\Big]}{u_{1,n}^2 + \overline{y}_n^2} - k_1 \tag{A.0.18}$$

$$\begin{aligned}\frac{\partial f_4(x,u)}{\partial u_1} = -\epsilon\Bigg\{ & \frac{2\Big[2u_{1,n}u_{2,n}(u_c\sin(\psi_n) + \overline{v}_n\cos(\psi_n)) + \overline{y}_n\Big((u_c\sin(\psi_n) + \overline{v}_n\cos(\psi_n))^2 - u_{2,n}^2\Big)\Big]}{(u_{,n}1^2 + \overline{y}^2)^2} \\ & - \frac{8u_{1,n}\Big[u_{2,n}(u_c\sin(\psi_n) + \overline{v}_n\cos(\psi_n))(u_{1,n}^2 - \overline{y}_n^2) + u_{1,n}\overline{y}_n\Big((u_c\sin(\psi_n) + \overline{v}_n\cos(\psi_n))^2 - u_{2,n}^2\Big)\Big]}{(u_{1,n}^2 + \overline{y}_n^2)^3} \\ & - \frac{\Big(r_n\cos(\psi_n)(u_c + u_c\Theta + M) + \overline{v}_n[(u_c\Xi + N)\cos(\psi_n) - r_n\sin(\psi_n)]\Big)}{u_1^2 + \overline{y}_n^2} \\ & + \frac{2u_{1,n}\Big[u_{1,n}\Big(r_n\cos(\psi_n)(u_c + u_c\Theta + M) + \overline{v}_n[(u_c\Xi + N)\cos(\psi_n) - r_n\sin(\psi_n)]\Big) - u_{3,n}\overline{y}_n\Big]}{(u_{1,n}^2 + \overline{y}_n^2)^2} \\ & - k_1\Big[\frac{(u_c\sin(\psi_n) + \overline{v}_n\cos(\psi_n))}{u_{1,n}^2 + \overline{y}_n^2} - \frac{2u_{1,n}[u_{1,n}(u_c\sin(\psi_n) + \overline{v}_n\cos(\psi_n)) - \overline{y}_n u_{2,n}]}{(u_{1,n}^2 + \overline{y}_n^2)^2}\Big] \\ & k_0\frac{\overline{y}_n}{u_{1,n}^2 + \overline{y}_n^2}\Bigg\}\end{aligned} \tag{A.0.19}$$

$$\frac{\partial f_4(x,u)}{\partial u_2} = -\epsilon\Bigg\{\frac{2[(u_c\sin(\psi_n) + \overline{v}_n\cos(\psi_n))(u_{1,n}^2 - \overline{y}_n^2) - 2u_{1,n}u_{2,n}\overline{y}_n]}{(u_{1,n}^2 + \overline{y}_n^2)^2} + k_1\frac{\overline{y}_n}{u_{1,n}^2 + \overline{y}_n^2}\Bigg\} \tag{A.0.20}$$

$$\frac{\partial f_4(x,u)}{\partial u_3} = -\epsilon\Bigg\{\frac{\overline{y}_n}{u_{1,n}^2 + \overline{y}_n^2}\Bigg\} \tag{A.0.21}$$

$$\frac{\partial f_6(x,u)}{\partial u_1} = \frac{2\Big[2u_{1,n}u_{2,n}(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))+\overline{y}_n\Big((u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))^2-u_{2,n}^2\Big)\Big]}{(u_{,n}1^2+\overline{y}^2)^2}$$

$$-\frac{8u_{1,n}\Big[u_{2,n}(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))(u_{1,n}^2-\overline{y}_n^2)+u_{1,n}\overline{y}_n\Big((u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))^2-u_{2,n}^2\Big)\Big]}{(u_{1,n}^2+\overline{y}_n^2)^3}$$

$$-\frac{\Big(r_n\cos(\psi_n)(u_c+u_c\Theta+M)+\overline{v}_n[(u_c\Xi+N)\cos(\psi_n)-r_n\sin(\psi_n)]\Big)}{u_1^2+\overline{y}_n^2}$$

$$+\frac{2u_{1,n}\Big[u_{1,n}\Big(r_n\cos(\psi_n)(u_c+u_c\Theta+M)+\overline{v}_n[(u_c\Xi+N)\cos(\psi_n)-r_n\sin(\psi_n)]\Big)-u_{3,n}\overline{y}_n\Big]}{(u_{1,n}^2+\overline{y}_n^2)^2}$$

$$-k_1\Big[\frac{(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))}{u_{1,n}^2+\overline{y}_n^2}-\frac{2u_{1,n}[u_{1,n}(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))-\overline{y}_nu_{2,n}]}{(u_{1,n}^2+\overline{y}_n^2)^2}\Big]$$

$$k_0\frac{\overline{y}_n}{u_{1,n}^2+\overline{y}_n^2} \tag{A.0.22}$$

$$\frac{\partial f_6(x,u)}{\partial u_2} = \frac{2[(u_c\sin(\psi_n)+\overline{v}_n\cos(\psi_n))(u_{1,n}^2-\overline{y}_n^2)-2u_{1,n}u_{2,n}\overline{y}_n]}{(u_{1,n}^2+\overline{y}_n^2)^2}+k_1\frac{\overline{y}_n}{u_{1,n}^2+\overline{y}_n^2} \tag{A.0.23}$$

$$\frac{\partial f_6(x,u)}{\partial u_3} = \frac{\overline{y}_n}{u_{1,n}^2+\overline{y}_n^2} \tag{A.0.24}$$

B Linear approximation of the derivative

The derivate of $f(t)$ is defined as

$$\dot{f}(t) = \lim_{h\to 0} \frac{f(t+h) - f(t)}{h}$$
$$\ddot{f}(t) = \lim_{h\to 0} \frac{\dot{f}(t+h) - \dot{f}(t)}{h}$$

Form Taylors theorem we have that

$$f(t+h) = f(t) + \frac{h}{1!}\dot{f}(t) + \frac{h^2}{2!}\ddot{f}(t) + ... + \frac{h^n}{n!}f^{(n)}(t) + \frac{h^{n+1}}{(n+1)!}f^{n+1}(t+\vartheta h) \tag{B.0.1}$$

where the last term denotes the residue, and

$$0 < \vartheta < 1$$

The last term ϑ approaches zero when $\lim_{n\to\infty}$ and equation B.0.1 converges. For n=1 we have that

$$f(t+h) = f(t) + \frac{h}{1!}\dot{f}(t) + \frac{h^2}{2!}\ddot{f}(t+\vartheta h) \tag{B.0.2}$$

If the expression B.0.2 for $f(t+h)$ is solved for $\dot{f}(t)$, and h approaches zero, we obtain the definition of the derivative $\dot{f}(t)$.

$$\begin{aligned}\dot{f}(t) &= \frac{f(t+h) - f(t)}{h} - \frac{h}{2!}\ddot{f}(t+\vartheta h) \\ &= \frac{f(t+h) - f(t)}{h} + o(h)\end{aligned} \tag{B.0.3}$$

where o(h) is the residue. If h is sufficiently small, that is o(h)≈0, the derivative can be estimated by

$$\dot{f}(t) = \frac{f(t+h) - f(t)}{h} \tag{B.0.4}$$

similarly the second derivative can be estimated by

$$\ddot{f}(t) = \frac{\dot{f}(t+h) - \dot{f}(t)}{h} \tag{B.0.5}$$

the first and second derivative of Δ can now be found from

$$\dot{\Delta}_i = \frac{\Delta(t_{i+1}) - \Delta(t_i)}{t_{i+1} - t_i} \tag{B.0.6}$$
$$\ddot{\Delta}_i = \frac{\dot{\Delta}(t_{i+1}) - \dot{\Delta}_i}{t_{i+1} - t_i} \tag{B.0.7}$$

Note: *If these approximation is to be good enough, the difference between t_i and t_{i-1} needs to be small enough. For the implementation it will be needed to determine the initial value of $\dot{\Delta}$ in order to calculate the first $\ddot{\Delta}$.*

C Computer details

System:

Microsoft Windows XP Professional v2002 Service Pack 2

Simulation environment:

Hardware:

- Intel(R) Pentium(R)D CPU 3GHZ DualCore
- 2 GB RAM

Software:

- Matlab 7.4.0 (R2007a)

D Optimal path-following of several straight lines, *current2*

Each section contains three figures which are the results for simulations with *current2*. All other parameters are unchanged relative to the chapter *Optimal path-following of several straight lines*.

D.1 cost-function2

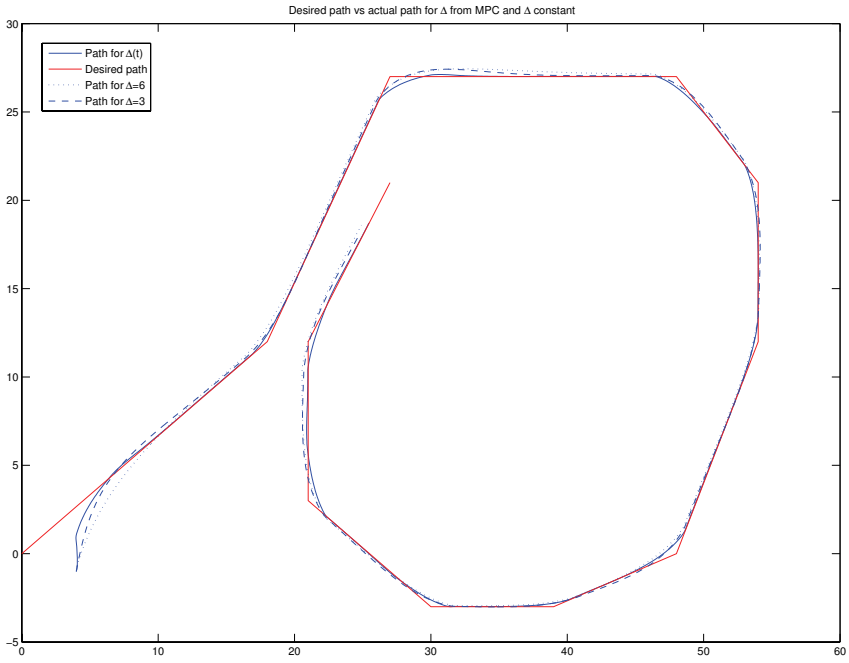

Figure 69: Position of the vessel for *cost-function2* when the vessel is subjected to *current2*.

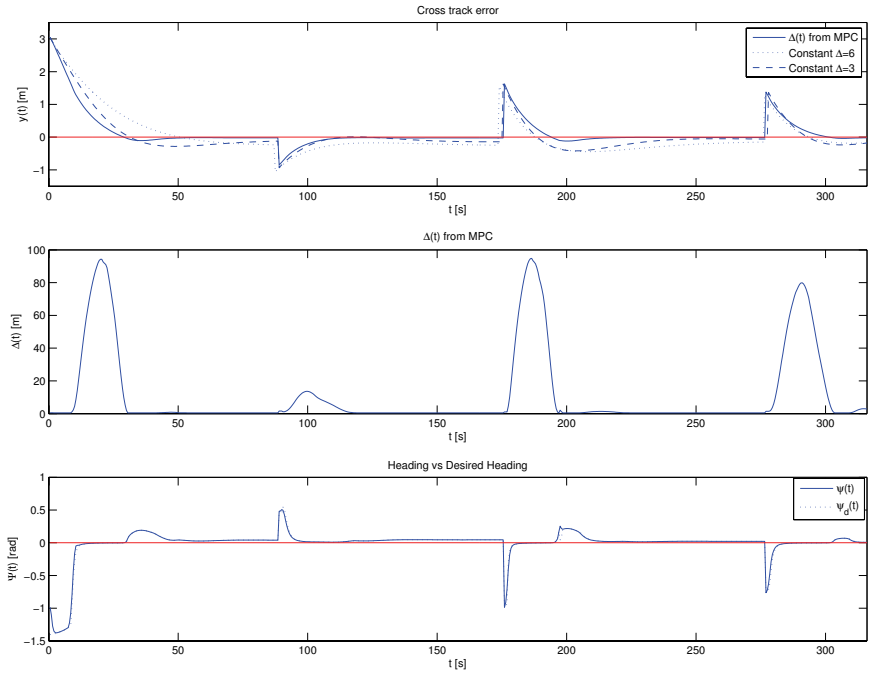

Figure 70: States from time $t = 0$ to $t = 316$ seconds, *cost-function2*, the vessel is subjected to *current2*.

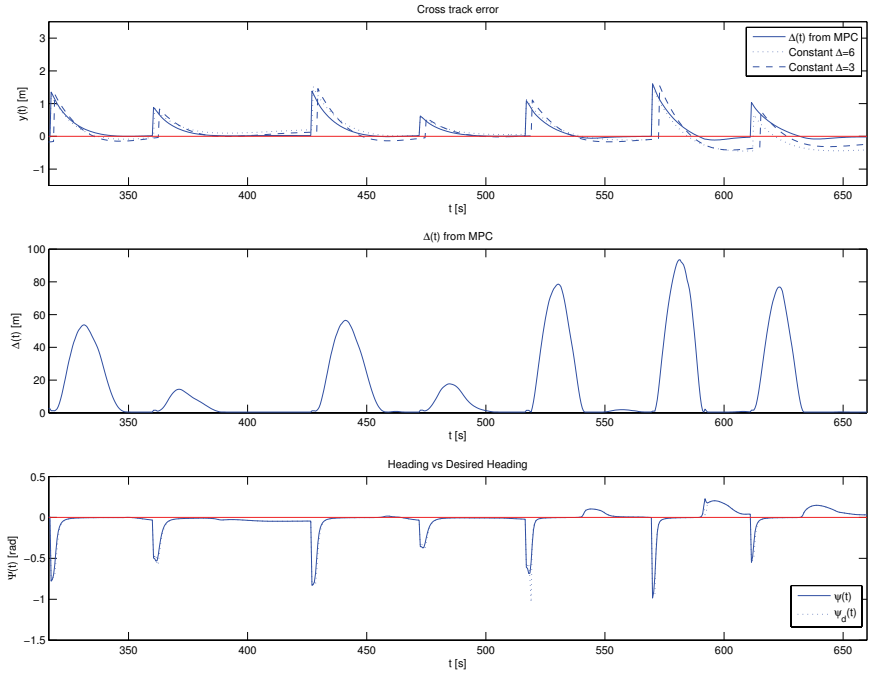

Figure 71: States from time $t = 316.5$ to $t = 660$ seconds, *cost-function2*, the vessel is subjected to *current2*.

D.2 cost-function3

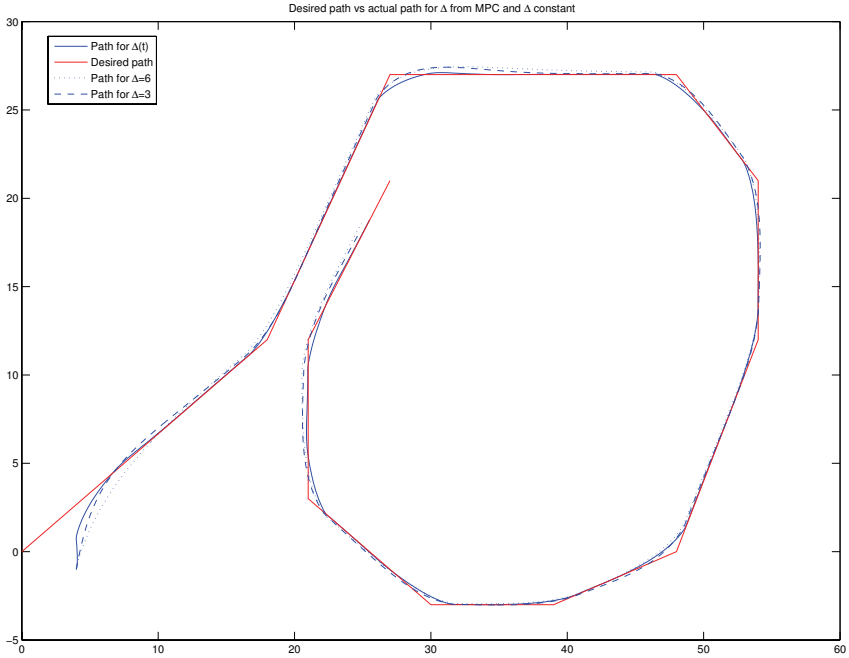

Figure 72: Position of the vessel for *cost-function3* when the vessel is subjected to *current2*.

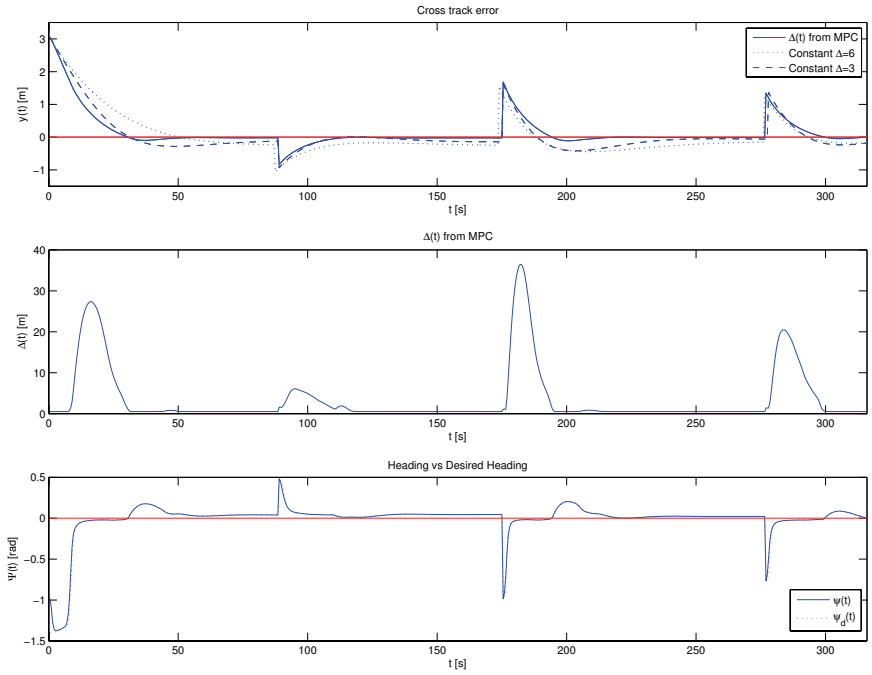

Figure 73: States from time $t = 0$ to $t = 316$ seconds, *cost-function3*, the vessel is subjected to *current2*.

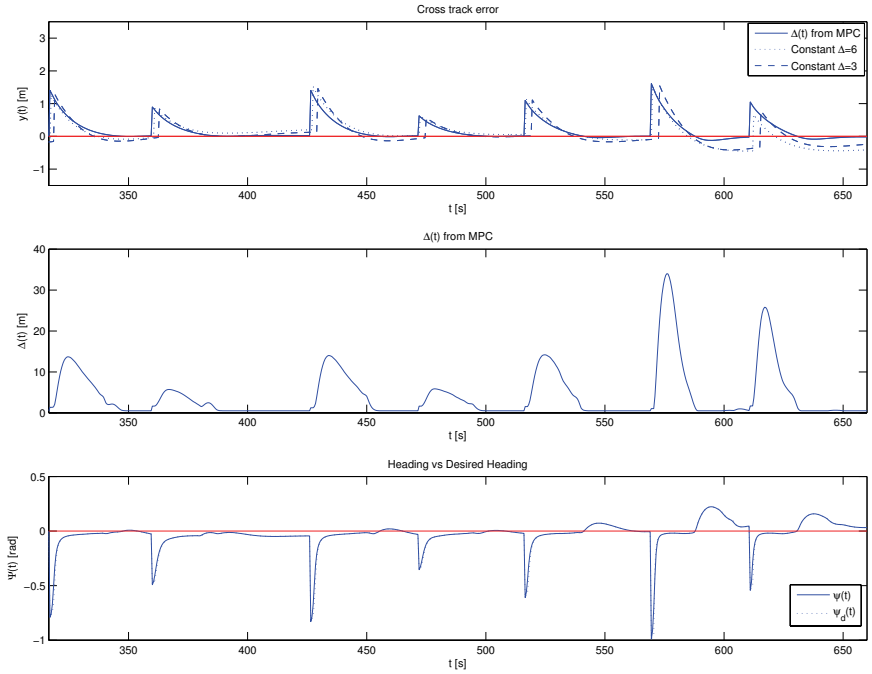

Figure 74: States from time $t = 316.5$ to $t = 660$ seconds, *cost-function3*, the vessel is subjected to *current2*.

D.3 cost-function5

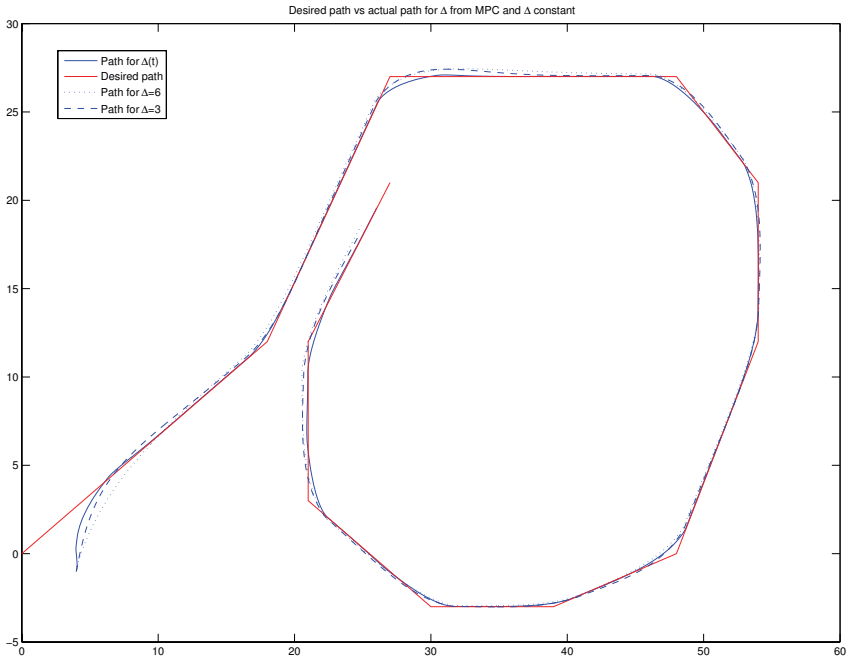

Figure 75: Position of the vessel for *cost-function5* when the vessel is subjected to *current2*.

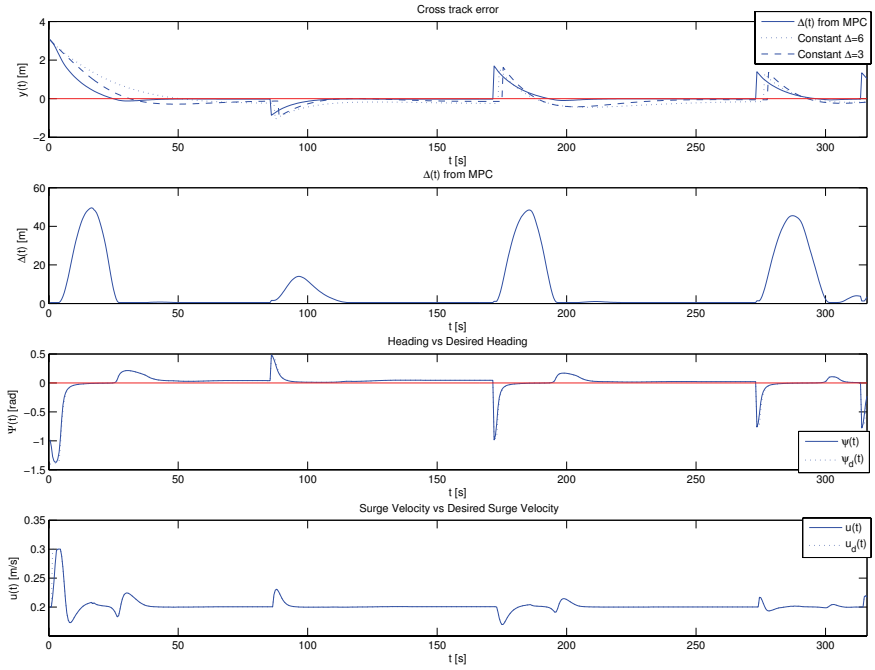

Figure 76: States from time $t = 0$ to $t = 316$ seconds, *cost-function5*, the vessel is subjected to *current2*.

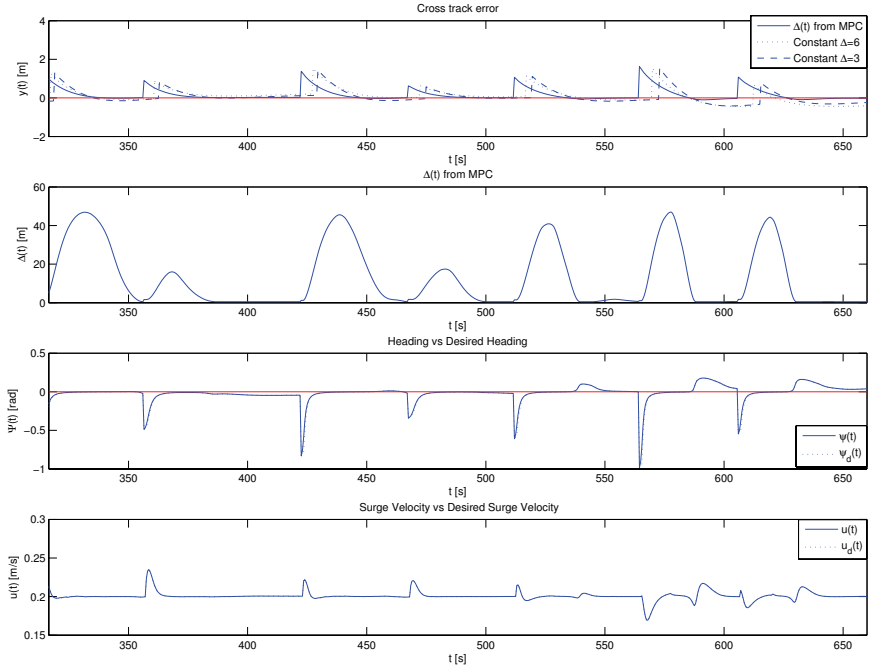

Figure 77: States from time $t = 316.5$ to $t = 660$ seconds, *cost-function5*, the vessel is subjected to *current2*.

D.4 cost-function6

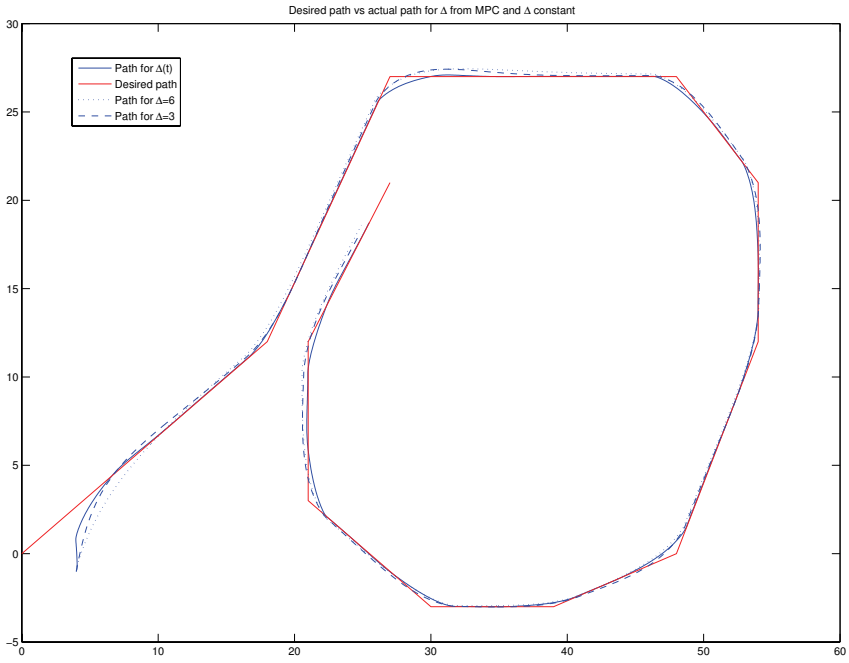

Figure 78: Position of the vessel for *cost-function6* when the vessel is subjected to *current2*.

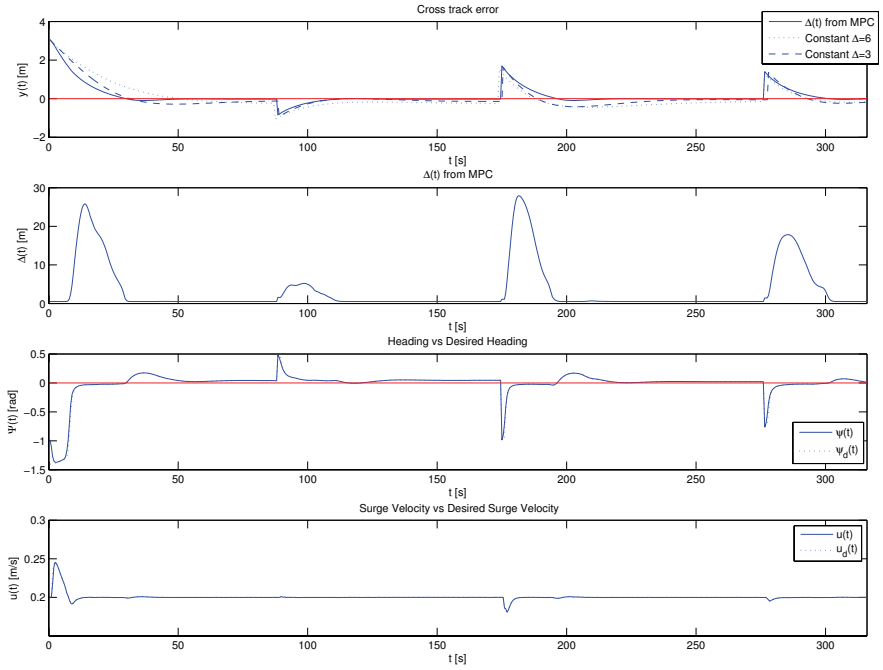

Figure 79: States from time $t = 0$ to $t = 316$ seconds, *cost-function6*, the vessel is subjected to *current2*.

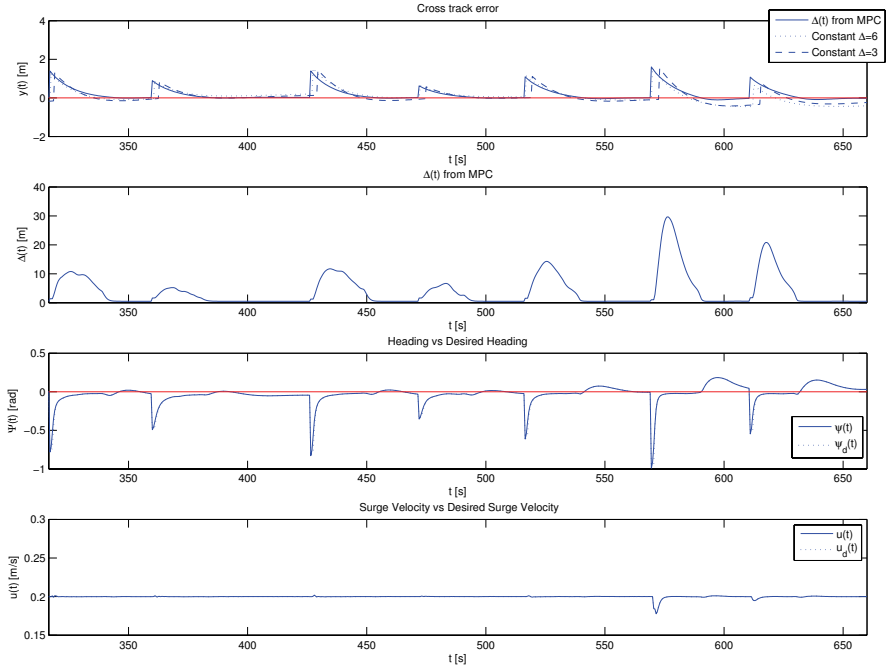

Figure 80: States from time $t = 316.5$ to $t = 660$ seconds, *cost-function6*, the vessel is subjected to *current2*.

E How to produce the results

To produce the figures in the Chapter *Line of sight and Way points*, sectioin *Way point swithing, angle og the path and initial look-ahead distance*, run file *mainCost23Pathtest.m*. Set variables *better* = 0 and *goodChoice* = 0 to produce Figures 5 and 7. Set variables *better* = 1 and *goodChoice* = 0 to produce Figures 6 and 8. Set variables *better* = 1 and *goodChoice* = 1 to produce Figure 9. The Figure 10 is produced each time the file runs.

To produce the results in Chapter *Optimal path-following for a straight line path* Section *cost-function1* run file *mainqp.m* and set: *current* = 1 if current should be included else *current* = 0, *betac* = $\frac{\pi}{3}$ for *current1 betac* = $-\frac{\pi}{3}$ for *current2*, *cost* = 0 for Q_1, *cost* = 1 for Q_2, *cost* = 2 for Q_3, simulation time in seconds is the variable *stopp*. The prediction horizon is H_p.

To produce the results in Chapter *Optimal path-following for a straight line path* Section *cost-function2* run file *mainCost23.m* and set: *current* = 1 if current should be included else *current* = 0, *betac* = $\frac{\pi}{3}$ for *current1 betac* = $-\frac{\pi}{3}$ for *current2*, *cost* = 0 for $k_{\dot{y}} = 160$, *cost* = 1 for $k_{\dot{y}} = 200$, simulation time in seconds is the variable *stopp*. For the results from using the LTV-model, run file *mainqpdy.m* set: *cost* = 0 for Q_4 and R_2, *cost* = 1 for Q_5 and R_2, *cost* = 2 for Q_5 and R_3, *cost* = 3 for Q_5 and R_4, *cost* = 4 for Q_6 and R_5. The prediction horizon is H_p. Current and simulation time is set as in *mainCost23.m*. The returned variable G contains the matrices Φ_{k+i}, an the returned variable J contains the matrices Γ_{k+i}.

To produce the results in Chapter *Optimal path-following for a straight line path* Section *cost-function3* run file *mainCost23.m* and set: *current* = 1 if current should be included else *current* = 0, *betac* = $\frac{\pi}{3}$ for *current1 betac* = $-\frac{\pi}{3}$ for *current2*, *cost* = 2 for $k_y = 0.08$, *cost* = 3 for $k_y = 0.12$, simulation time in seconds is the variable *stopp*.

To produce the results in Chapter *Optimal path-following for a straight line path* Section *cost-function5*, run file *mainUlinUs.m* and set: *path* = 0, *cost* = 0 for $k_{u_d,1}, k_{\dot{u}_d,1}$, *cost* = 1 for $k_{u_d,1}, k_{\dot{u}_d,2}$, for current *current* = 1, *betac* = $\frac{\pi}{3}$ for *current1 betac* = $-\frac{\pi}{3}$ for *current2*, simulation time in seconds is the variable *stopp*.

To produce the results in Chapter *Optimal path-following for a straight line path* Section *cost-function6*, run file *mainUlinUs.m* and set: *path* = 0, *cost* = 2 for $k_{u_d,1}, k_{\dot{u}_d,1}$, *cost* = 3 for $k_{u_d,2}, k_{\dot{u}_d,3}$, for current *current* = 1, *betac* = $\frac{\pi}{3}$ for *current1 betac* = $-\frac{\pi}{3}$ for *current2*, simulation time in seconds is the variable *stopp*.

To produce the results in Chapter *Optimal path-following of several straight lines.* Section *cost-function2*, run file *mainCost23Path.m* and set: *current* = 1, *betac* = $\frac{\pi}{3}$ for *current1 betac* = $-\frac{\pi}{3}$ for *current2*, *cost* = 0 for $k_{\dot{y}} = 160$, simulation time in seconds is the variable *stopp*.

To produce the results in Chapter *Optimal path-following of several straight lines.* Section *cost-function3*, run file *mainCost23Path.m* and set: *current* = 1, *betac* = $\frac{\pi}{3}$ for *current1 betac* = $-\frac{\pi}{3}$ for *current2*, *cost* = 3 for $k_y = 0.12$, simulation time in seconds is the variable *stopp*.

To produce the results in Chapter *Optimal path-following of several straight lines.* Section *cost-function5*, run file *mainUlinUs.m* and set: *path* = 1, *current* = 1, *betac* = $\frac{\pi}{3}$ for *current1 betac* = $-\frac{\pi}{3}$ for *current2*, *cost* = 1, simulation time in seconds is the variable *stopp*.

To produce the results in Chapter *Optimal path-following of several straight lines.* Section *cost-function6*, run file *mainUlinUs.m* and set: *path* = 1, *current* = 1, *betac* = $\frac{\pi}{3}$ for *current1 betac* = $-\frac{\pi}{3}$ for *current2*, *cost* = 3, simulation time in seconds is the variable *stopp*.

To produce the figures which divides the figure of the several-straight-line path in two figures, run file *plottResPath2.m* for *cost-function2* and *cost-function3*, and file *plottResPath.m* for *cost-function5* and *cost-function6*.

Note *The printout made by the optimizer* fmincon *contains information on which inputs are at which constraint, this is not the same as the constraint being violated.*

F Possible improvements to reduce calculation time

The calculation time can be reduced by reducing H_p, H_u, T_s. Thus, the optimal values of these variables should be found. That is the smallest values which still yields god performance.

The implementation has been done in *Matlab m-files*. This is not the most efficient language in terms of calculation time, thus the calcultion time can be reduced by coding in a more efficient language.

The calcultion time can also be reduced if a faster solver can be used. Thus the possibility of a faster solver should be investigated. It might be possible to write a solver which takes advantage of the knowledge of the system.

An improved LTV model should be derived since (in general) a LTV based minimization problem solved by *quadprog* has shorter calculation times than a nonlinear model based optimization problem solved by *fmincon*. This will also remove the problems with non-convexity.