Real-time Trajectory Optimization for Terrain Following Based on Non-linear Model Predictive Control

Cecilia Flood

LiTH-ISY-EX-3208
2001-11-01
Real-time Trajectory Optimization for Terrain Following Based on Non-linear Model Predictive Control

Cecilia Flood

LiTH-ISY-EX-3208
2001-11-01
Abstract

There are occasions when it is preferable that an aircraft flies as close to the ground as possible. It is difficult for a pilot to predict the topography when he cannot see beyond the next hill, and this makes it hard for him to find the optimal flight trajectory. With the help of a terrain database in the aircraft, the forthcoming topography can be found in advance and a flight trajectory can be calculated in real-time. The main goal is to find an optimal control sequence to be used by the autopilot. The optimization algorithm, which is created for finding the optimal control sequence, has to be run often and therefore, it has to be fast.

This thesis presents a terrain following algorithm based on Model Predictive Control which is a promising and robust way of solving the optimization problem. By using trajectory optimization, a trajectory which follows the terrain very good is found for the non-linear model of the aircraft.
Preface

This master thesis was carried out at the department “Primary flight data and navigation” at Saab AB in Linköping. I would like to send my appreciation to the people at the department who have been very helpful and also made the time during my research more fun. Specially, I would like to thank my supervisors Predrag Pucar (Saab) and Johan Löfberg (LiTH) for their patient and appreciated help with both my research and this report.

Without the support from my family and friends, the last couple of years of studying would not have been as fun as they have been so a special thanks goes to them. Also, I would like to thank Fredrik Wanhainen for all your help and support, you are my sunshine.

Linköping, 2001-11-01

Cecilia Flood
Titel
Title
Trajektorieoptimering för terrängföljning i realtid baserad på olinjär prediktionsregering
Real-time Trajectory Optimization for Terrain Following Based on Non-linear Model Predictive Control

Författare
Author
Cecilia Flood

Sammanfattning
Abstract
There are occasions when it is preferable that an aircraft flies as close to the ground as possible. It is difficult for a pilot to predict the topography when he cannot see beyond the next hill, and this makes it hard for him to find the optimal flight trajectory. With the help of a terrain database in the aircraft, the forthcoming topography can be found in advance and a flight trajectory can be calculated in real-time. The main goal is to find an optimal control sequence to be used by the autopilot. The optimization algorithm, which is created for finding the optimal control sequence, has to be run often and therefore, it has to be fast.

This thesis presents a terrain following algorithm based on Model Predictive Control which is a promising and robust way of solving the optimization problem. By using trajectory optimization, a trajectory which follows the terrain very good is found for the non-linear model of the aircraft.
<table>
<thead>
<tr>
<th>Nyckelord</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model Predictive Control, MPC, Non-linear Systems, Terrain Following, Trajectory Optimization</td>
<td></td>
</tr>
</tbody>
</table>
1 Introduction

At occasions when it is important that a military aircraft flies without getting discovered or when it is pursued, it might be desirable to fly as low as possible. It is a complicated task for the pilot to find the optimal flight trajectory when his only available sources of information are his own visual field and the radar image. By using these sources, only a locally optimal flight trajectory can be found since there is no information about the terrain beyond the next hill. The exposure time to threats is unnecessarily high and there are risks of getting into situations when collision with the ground is a risk. If the terrain is known by the pilot, he can make a good optimization himself, but when flying over unknown terrain, it is desirable to have a utility that can calculate the best flight path.

A secure flight trajectory can be calculated if a database of the terrain is available in the aircraft. By using the information in the terrain database, a global optimal trajectory can be calculated in real-time. This trajectory can either be presented to the pilot, so he manually can follow it to the best of his abilities, or a sequence of control signals can be sent directly to the autopilot.

1.1 Problem Specification

The objective of terrain following is to find the optimal trajectory that lies as near the ground as possible. The problem lies in minimizing the height over a reference trajectory when the aircraft’s limitations and dynamics are considered. A database containing information about the terrain should be located in the aircraft, and this database is used for finding the optimal flight trajectory. The database used for this thesis only contains information about the ground, and does not include the height of trees and buildings. Because of that, a reference height is set at e.g. 50 meters over the ground and the aircraft is not allowed below this reference trajectory. The object of this thesis is to find an optimization algorithm and not to examine the safety aspect of the flight. The reference trajectory can be varied to make the flight safer, but that is not done in this thesis.

An algorithm for three dimensions is desired, but in this thesis only two dimensions are considered. Optimization in two dimensions means that the aircraft only flies on a straight line from the starting point to the final point and the optimal height over the terrain is calculated along this line. In optimization in three dimensions, the aircraft also has the ability to turn to the sides around hills. This would make the height over the ground even smaller.

1.2 Outline

This thesis will study how the trajectory optimization problem can be solved with Model Predictive Control (MPC). Two different approaches to MPC will be analyzed and compared to find an algorithm which is fast and calculates a trajectory that is as close to the reference trajectory as possible. In a first attempt to use these approaches, some simplifications are made such as considering the aircraft to be a point mass and the earth to be flat, as opposite a globe. When the best algorithm and parameters are found, aircraft dynamics are added to it but the earth will still be considered flat.
The theories behind aircraft dynamics are discussed in Chapter 2. In Chapter 3, the MPC algorithm is introduced and it is implemented in Chapter 4 and 5 for two aircraft models (a point mass model and a model including the aircraft dynamics). The simulations of the algorithm are presented in Chapter 6 together with an analysis of what parameters to use. The final results, computational aspects and conclusions are presented in Chapters 7, 8 and 9 respectively and Chapter 10 gives some ideas about future work.

In Appendix A, there is a list of notations and also an explanation of some words used in the thesis.
2 The Aircraft

This chapter discusses the theories behind aircraft dynamics (2.2) and also the properties which are important to consider when controlling an aircraft (2.1). A lot of the theory in this chapter is reviewed more thoroughly in [1].

2.1 Inertial Navigation

In developing an algorithm for aircraft control, there are many factors that have to be considered. When flying, the aircraft can both accelerate and rotate in different directions and these depend on each other and are limited. The aircraft’s properties can be presented in three coordinate frames (2.1.1), the body frame (which is fixed to the aircraft and moves with it), the local level frame (which is also fixed to the aircraft but does not rotate with it) and the stability frame. The three angles, $\psi$, $\phi$ and $\theta$ give the relationship between the body and local level frames.

2.1.1 Coordinate Frames

Three different coordinate frames are considered, the L frame, the B frame and the stability frame. These frames are shown in Figures 1 and 2.

L - Local level frame:
This frame, also called the NED frame, has its origin at the center of gravity of the aircraft. The $x$ axis always points to the north, the $y$ axis to the east and the $z$ axis downwards relative the earth’s surface. The values in the local level frame have the indices $n$, $e$ and $d$.

B - Body frame:
The origin is also in this frame at the center of gravity of the aircraft. The $x$ axis points out from the nose, the $y$ axis out from the right wing, and the $z$ axis points downwards relative the aircraft. The values in the body frame have the indices $x$, $y$ and $z$.

FIGURE 1. The coordinate frames B (solid) and L (dashed).
S - Stability frame:

When the aircraft flies straight forward, the nose points up from the horizon with an angle \( \alpha \), and the velocity vector coincides with the horizon. The frame that are rotated \( \alpha \) around the y axis is called the stability frame and the velocity vector is in this frame’s \( x \) direction. The angle between the horizon and the velocity vector, \( \gamma \), is in level flight zero. As the aircraft rotates upwards, the velocity vector rises from the horizon and \( \gamma \) is increased according to Figure 2.

![Figure 2. The angles \( \alpha \) and \( \gamma \) and the velocity vector \( v_{tot} \).](image)

2.1.2 Position

The globe is throughout the thesis simplified to be flat instead of being spherical. The result will still be accurate for our purposes since the curvature of the globe is negligible over intervals of 3 kilometers.

The position of the aircraft is described with a vector containing latitude (\( l \)), longitude (\( L \)) and altitude (\( h \)). Latitude and longitude are measured in degrees (\( 0 - 360^\circ \)) while the altitude is measured in meters. The terrain database is based on latitude and longitude so that is why these are used instead of \( x \) and \( y \) values in meters.

2.1.3 Attitude

The aircraft accelerates and rotates relative the L frame while the B frame moves with it. The attitude is defined by three angles, the yaw, roll and pitch angles (\( \psi, \phi, \theta \)). These angles describe the relation between the body frame and the local level frame as shown in Figure 3.

The yaw angle, \( \psi \), is defined as the angle between \( n \) and \( x \).

The roll angle, \( \phi \), is defined as the angle between the horizontal plane and \( y \).

The pitch angle, \( \theta \), is defined as the angle between the horizontal plane and \( x \).
2.1.4 Velocity Equations

When deriving the aircraft dynamics, some velocity equations will be needed and this section will give a brief overview of these. The relation between the time derivative of a velocity vector in coordinate frame B and a velocity vector in coordinate frame L is

\[
\frac{d}{dt}v_L = \left. \frac{d}{dt}v_B \right|_B + \omega_{BL}^T \times v_B
\]

(2.1)

where \( \omega_{BL}^T = (p, q, r) \) is frame B’s rotation in relation to frame L. A circumscription of the equation, and the accelerations \( a_x, a_y \) and \( a_z \) in frame B yield

\[
\begin{align*}
\dot{v}_n &= a_x - qv_z + rv_y \\
\dot{v}_e &= a_y + pv_z - rv_x \\
\dot{v}_d &= a_z - pv_y + qv_x
\end{align*}
\]

(2.2)

The states \( l, L \) and \( h \) are updated using the velocities in Equation 2.2:

\[
\begin{align*}
\dot{l} &= v_n T_{m2deg} \\
\dot{L} &= v_e T_{m2deg} \\
\dot{h} &= -v_d
\end{align*}
\]

(2.3)
\( \dot{h} \) has a negative sign since the vector \( d \) in the L frame points downwards relative the aircraft and the height over the ground has the opposite direction. A transformation of the latitude and longitude from meters into degrees has to be done in the calculation of \( \dot{l} \) and \( \dot{L} \) since the velocities in the L frame are expressed in m/s and \( \dot{l} \) and \( \dot{L} \) are expressed in degrees/s. The transformation is done according to Equation 2.4 where \( R = 6378000 \) meters is the radius of the earth. If the globe was considered spherical, \( T_{m2deg} \) would depend on the current attitude too.

\[
T_{m2deg} = \frac{180}{\pi R}
\]  

(2.4)

In the actual model, the velocity will be stated in the B frame instead of in the L frame as in Equation 2.3 so a transformation between these frames has to be done.

### 2.1.5 Transformation Between the B and L Frame

A transformation between the two frames, B and L, is needed as discussed above.

\[
\begin{pmatrix}
  v_n \\
  v_e \\
  v_d
\end{pmatrix} = C_B^L
\begin{pmatrix}
  v_x \\
  v_y \\
  v_z
\end{pmatrix}
\]  

(2.5)

\( C_B^L \) is a transformation matrix which depends on \( \phi, \theta \) and \( \psi \).

\[
C_B^L =
\begin{pmatrix}
  \cos \psi & -\sin \psi & 0 \\
  \sin \psi & \cos \psi & 0 \\
  0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
  \cos \theta & 0 & \sin \theta \\
  0 & 1 & 0 \\
  -\sin \theta & 0 & \cos \theta
\end{pmatrix}
\begin{pmatrix}
  1 & 0 & 0 \\
  0 & \cos \phi & -\sin \phi \\
  0 & \sin \phi & \cos \phi
\end{pmatrix}
\]  

(2.6)

In this thesis, the yaw angle will be constant and the roll angle will always be equal to zero since the aircraft is not allowed to turn. Equation 2.6 can therefore be simplified into

\[
C_B^L =
\begin{pmatrix}
  \cos \psi & -\sin \psi & 0 \\
  \sin \psi & \cos \psi & 0 \\
  0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
  \cos \theta & 0 & \sin \theta \\
  0 & 1 & 0 \\
  -\sin \theta & 0 & \cos \theta
\end{pmatrix}
\]  

(2.7)

### 2.2 Aircraft Dynamics

To be able to understand the aircraft dynamics, this section will present a more detailed analysis of how the aircraft rotates and accelerates (2.2.3) when a load factor (2.2.1) is commanded.
The control of the aircraft is done by changing the thrust ($\delta_t$) and the deflection of the canard, aileron, rudder and elevator ($\delta_c$, $\delta_a$, $\delta_r$ and $\delta_e$). The elevators and the canards stabilize the aircraft which is inherently longitudinally unstable at subsonic speeds. They also control the acceleration in the $z$ direction. The rudders are used to coordinate the turns (yawing) while the ailerons are used to control the roll motion.

2.2.1 Load Factor

For this thesis the desired load factor is decided and the actual control signals are calculated from that. When an acceleration upwards is desired, the aircraft cannot just accelerate up in its $z$ direction. Instead, the aircraft has to rotate and in that way create an upward motion. The load factor, $n_z$, is the control signal and is measured in $\text{m/s}^2$. This signal is set as an input signal to the aircraft which, with the help of a controller, transforms the signal into a rotation and an acceleration.

2.2.2 Aerodynamic Forces and Moments

The aerodynamic forces and moments depend on the configuration of the aircraft. Factors that come into play are the wing span, the wing form and also the air properties and air speed. In the derivations in this section the aerodynamical notation below will be used.

- Wing reference area, $S$ [m$^2$]
- Wing span, $b$ [m$^2$]
- Dynamic pressure, $p$ [N/m$^2$]

![Aerodynamic forces and moments](FIGURE_4.png)
Assume that the state, $x$ consist of velocity and rotation. Further we denote the control surface deflections by $\delta$. The force and aerodynamic coefficients are:

$$
C_{F_L} = f_{F_L}(x, \delta) \\
C_{F_D} = f_{F_D}(x, \delta) \\
C_{F_S} = f_{F_S}(x, \delta) \\
C_{\Im} = f_{\Im}(x, \delta) \\
C_{\Re} = f_{\Re}(x, \delta) \\
C_{\wp} = f_{\wp}(x, \delta)
$$

(2.8)

The aircraft is affected by the drag force $F_D$, lift force $F_L$, and side force $F_S$ which are expressed in the stability frame defined in Section 2.1.1. The aerodynamic moments $\Im$, $\Re$ and $\wp$ are expressed in the body frame. The relations between the forces and the aerodynamic coefficients are defined by

$$
\sum F_L = -SqC_{F_L} \\
\sum F_D = -SqC_{F_D} \\
\sum F_S = -SqC_{F_S} \\
\sum \Im = -SqC_{\Im} \\
\sum \Re = -SqC_{\Re} \\
\sum \wp = -SqC_{\wp}
$$

(2.9)

### 2.2.3 Equations of Motion

By using Newton’s laws, the equations of motions can be derived. $\sum F$ is the sum of the forces acting on the aircraft and $\sum M$ is the sum of the moments.

$$
\sum F = ma \\
\sum M = I \dot{\omega}
$$

(2.10)

After some manipulation that can be found in Section 2.1.4 and in [1], the following equations of motions can be found in the same manner as in Equation 2.2.

$$
\sum F_x = m(\dot{v}_x + v_z q - v_y r) \\
\sum F_y = m(\dot{v}_y + v_x r - v_z p) \\
\sum F_z = m(\dot{v}_z + v_y p - v_x q)
$$

(2.11)
\[ \sum \mathcal{I} = \dot{p} I_x - \dot{r} J_{xz} + qr (I_z - I_y) - pq J_{xz} \]
\[ \sum \mathcal{R} = q I_y + pr (I_x - I_z) + (p^2 - r^2) J_{xz} \]
\[ \sum \varphi = \dot{r} I_z - \dot{p} J_{xz} + pq (I_y - I_x) + qr J_{xz} \]  
(2.12)

\[ J_{xz} \] is the product of inertia between the frames, and \( I_x, I_y \) and \( I_z \) are the moments of inertia in the different directions \( x, y \) and \( z \) respectively. \( J_{xz}, I_x, I_y \) and \( I_z \) are all constant and do not vary with the rotation. If we solve for the state variables, we obtain

\[ \dot{v}_x = v_y r - v_z q + \frac{1}{m} \sum F_x \]
\[ \dot{v}_y = v_z p - v_y r + \frac{1}{m} \sum F_y \]
\[ \dot{v}_z = v_x q - v_y p + \frac{1}{m} \sum F_z \]

(2.13)

\[ \dot{r} = \frac{\sum \varphi + \frac{J_{xz}}{I_x} \sum \mathcal{I} - \frac{J_{xz}}{I_x} qr (I_z - I_y) + \frac{J_{xz}}{I_x} pq}{I_z - \frac{J_{xz}^2}{I_x}} \]
\[ \dot{p} = \frac{1}{I_x} (\sum \mathcal{I} + \dot{r} J_{xz} - qr (I_z - I_y) + pq J_{xz}) \]
\[ \dot{q} = \frac{1}{I_y} (- pr (I_x - I_z) - (p^2 - r^2) J_{xz} + \sum \mathcal{R}) \]  
(2.14)

Equations 2.13 and 2.14 will be used for generating Equation 4.8.
3 Model Predictive Control

Model Predictive Control, MPC, is a method of predicting the process output at future time instants. An objective function (Equation 3.3) is minimized to give the optimal control sequence for each instant. The underlying theory reviewed in this chapter can be found in [3], [10] and [11].

3.1 The Basic Linear MPC Controller

In the basic formulation of the MPC problem, the model is time discrete and linear

\[ x(k + 1) = Ax(k) + Bu(k) \]
\[ y(k) = Cu(k) \]

(3.1)

with the control constraints

\[ u \in U \]

(3.2)

The characteristics of the optimization problem is that the problem is simplified from an optimal control problem with an infinite horizon \((N \to \infty)\) into a problem with a finite horizon \((N < \infty)\).

\[ I(k) = \sum_{j=k}^{k+N-1} y^T(j|k)Qy(j|k) + u^T(j|k)Ru(j|k) \]

(3.3)

This yields the MPC optimization problem

\[
\begin{align*}
\text{min} & \quad J(k) = \sum_{j=k}^{k+N-1} y^T(j|k)Qy(j|k) + u^T(j|k)Ru(j|k) \\
\text{subject to} & \quad u(k + j|k) \in U \\
& \quad x(k + j|k) = Ax(k + j - 1|k) + Bu(k + j - 1|k) \\
& \quad y(k) = Cu(k)
\end{align*}
\]

(3.4)

\(Q\) and \(R\) are positive definite matrices. The basic MPC controller is defined as

1. **Measure** \(x(k|k)\)
2. **Obtain the control sequence** \(u(\cdot|k)\) along the whole horizon by solving the optimization problem 3.4
3. **Apply** \(u(k) = u(k|k)\)
4. **Time update** \(k := k + 1\)
5. **Repeat from step 1**
3.1 Model Predictive Control, The Basic Linear MPC Controller

3.1.1 Quadratic Programming

An optimization problem with a quadratic objective function, \( J \), and linear constraints is called a quadratic program. These programs can always be solved in a finite number of iterations. The number of free variables and constraints in the objective function decide how much effort is required to find the optimal solution.

Quadratic Programming, QP, will be used later on as a way of solving the MPC problem. In the QP algorithm, the problem

\[
\min_u J(u) = \frac{1}{2} u^T Q u + f^T u
\]

subject to \( Au = b \)
\( Eu \leq e \)

is solved. The QP problem has been widely studied and there are many efficient ways of solving the problem. Some of these can be found in [12].

3.1.2 QP Formulation of MPC

The signals in each state are put together in vectors according to

\[
X = \begin{bmatrix} x(k+1|k) \\ x(k+2|k) \\ \vdots \\ x(k+N|k) \end{bmatrix}, U_k = \begin{bmatrix} u(k|k) \\ u(k+1|k) \\ \vdots \\ u(k+N-1|k) \end{bmatrix}, \text{and } Y = \begin{bmatrix} y(k+1|k) \\ y(k+2|k) \\ \vdots \\ y(k+N|k) \end{bmatrix}
\]

and

\[
Y = H x(k|k) + S U_k
\]

where

\[
H = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix}, S = \begin{bmatrix} CB & 0 & \ldots & 0 \\ CAB & CB & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{N-1} B & CA^{N-2} B & \ldots & CB \end{bmatrix}
\]

To be able to use the QP algorithm, some circumscription of the minimization problem stated in the MPC section have to be done. In the derivations below, \( U_k \) is the optimal
control sequence. \( Q_{\text{new}} \) and \( R_{\text{new}} \) are matrices that consist of the matrices \( Q \) and \( R \) respectively in their diagonals.

The problem (3.3) can now be written as

\[
\min_{U_k} \quad (Hx(k|k) + SU_k)^T Q_{\text{new}}(Hx(k|k) + SU_k) + U_k^T R_{\text{new}} U_k
\]

subject to \( EU_k \leq e \)  

(3.9)

This can easily be written as a QP problem in the form in Section 3.1.1. However, we omit this since a similar problem will be described in detail in Section 4.6.2. The QP formulation for MPC is reviewed in [3].

### 3.2 Non-linear MPC Based on Trajectory Linearization

MPC was in the beginning primarily an algorithm for solving linear optimal control problems, but it has been shown to be applicable even to non-linear systems. To use the standard MPC framework, the problem has to be stated as a discrete time linear problem so some approximations have to be done. These approximations consist of sampling of the model after it has been linearized. [10] shows two algorithms for linearizing the model, current state linearization and trajectory linearization. Since the second algorithm is shown to give a better result, this one is used for the linearization. The following sections will present the theory behind non-linear MPC based on trajectory linearization.

#### 3.2.1 Sampling and Linearization

In cases when we want to solve an optimal control problem in each sample, the model we use should be discrete. The model we wish to sample and linearize is from the beginning defined as

\[
\begin{align*}
\dot{x}(t) &= f(x(t), u(t)) \\
y(t) &= \tilde{C}x(t)
\end{align*}
\]

(3.10)

This model is linearized using the Taylor expansion ([13]) around the point \((x_0, u_0)\) so that an approximately linear model is obtained.

\[
\begin{align*}
\dot{x}(t) &= f(x_0, u_0) + \tilde{A}(x(t) - x_0) + \tilde{B}(u(t) - u_0) \\
y(t) &= \tilde{C}x(t)
\end{align*}
\]

(3.11)

\( \tilde{A} \) and \( \tilde{B} \) are the Jacobians of \( f \) with respect to \( x \) and \( u \) respectively.
3.2 Model Predictive Control, Non-linear MPC Based on Trajectory Linearization

According to [9] and [10], a discretization can then be done by calculating the new matrices $F$, $A$, $B$ and $C$ ($T$ is the sample time).

$$\tilde{A} = \left( \begin{array}{ccc} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{array} \right), \quad \tilde{B} = \left( \begin{array}{ccc} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} & \cdots & \frac{\partial f_1}{\partial u_m} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} & \cdots & \frac{\partial f_2}{\partial u_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \frac{\partial f_n}{\partial u_2} & \cdots & \frac{\partial f_n}{\partial u_m} \end{array} \right) \quad (3.12)$$

$$F = \int_0^T e^{\tilde{A}s} f(x_0, u_0) ds \quad (3.13)$$

$$A = e^{\tilde{A}T} \quad (3.14)$$

$$B = \int_0^T e^{\tilde{A}s} \tilde{B} ds \quad (3.15)$$

$$C = \tilde{C} \quad (3.16)$$

This yields a discrete state space model

$$\hat{x}(k+1) = x_0 + F(k) + A(x(k) - x_0) + B(u(k) - u_0) + y(k) = C\hat{x}(k) \quad (3.17)$$

The model can now be used as if it was linear, but it should be pointed out that this model only gives an approximate solution. To calculate a new state, $x$ (the properties at a sample instant), the previously calculated state together with the control signal are needed. Then, new linearizations and samplings are done around the new state and the proceeding state can be calculated.

### 3.2.2 Initial Trajectory Prediction

Calculation of the exact future trajectory would be very time consuming for a non-linear model. A discrete model of a linear system is an exact integration of a constant input on a linear continuous-time system. Linear models are calculated for each sample instant along the trajectory to give approximate integrations of the non-linear model. This yields discrete linear models for the whole horizon at the same time as the trajectory is calculated. The trajectory is discretized and the models are calculated according to Figure 5 and for each sample, the control signal and the linearized model are constant.
FIGURE 5. Discretization of the control sequence for an horizon of length N and calculation of the models along the linearized trajectory.

If the previously calculated control input is $(\hat{x}(k|k), u(k|k - 1))$, and the previously calculated control signals for the whole trajectory are put in a vector $U_{k-1}$ (in accordance to $U_k$ in Equation 3.6), the model 3.17 can be expressed as

$$
\begin{align*}
\dot{x}(k + 1|U_{k-1}) & = \dot{x}(k|k) + F(k) + A(x(k) - \hat{x}(k|k)) + B(u(k) - u(k|k - 1)) \\
\dot{y}(k + 1|U_{k-1}) & = C\dot{x}(k + 1|U_{k-1})
\end{align*}
\tag{3.18}
$$

The initial trajectory can be approximated by assuming that $x(k) = \hat{x}(k|k)$ and making the notation $\bar{x}(k + i) = \hat{x}(k + i|U_{k-1})$. An approximate equation for the trajectory is obtained when applying the previously calculated control signal, $u(k|k - 1)$ to the system.

$$
\bar{x}(k + 1) = x(k) + F(k)
\tag{3.19}
$$

$\bar{x}$ is the notation for the approximated trajectory and $F$ is recalculated between each sample and defined as in Equation 3.13. $F$ has to be recalculated because it depends on the lin-
earized approximated value of the previous state and the linearized vector \( U_{k-1} \). The initial trajectory for the whole horizon is calculated according to the equations below.

\[
\begin{align*}
\tilde{x}(k) &= x_0 \\
\tilde{x}(k+1) &= \tilde{x}(k) + F(k) \\
\tilde{x}(k+2) &= \tilde{x}(k+1) + F(k+1) = \tilde{x}(k) + F(k) + F(k+1) 
\end{align*}
\] (3.20)

Between each of these steps, the state is linearized and a new \( F \) is determined. This yields the approximation of the predicted trajectory.

\[
\tilde{x}(k+i) = \tilde{x}(k) + \sum_{j=0}^{i-1} F(k+j)
\] (3.21)

### 3.2.3 The Predictor

When the control sequence, \( U_k \), is found, the predictor can be defined for every future sample

\[
\hat{x}(k+1|U_k) = \hat{x}(k|k) + F(k) + A(x(k) - \hat{x}(k|k)) + B(u(k) - u(k|k-1))
\] (3.22)

After the assumption \( x(k) = \hat{x}(k|k) \), the new predicted trajectory can be calculated from \( U_k \).

\[
\begin{align*}
\hat{x}(k+1|U_k) &= x(k) + F(k) + B(k)(u(k) - u(k|k-1)) \\
&= \tilde{x}(k+1) + B(k)(u(k) - u(k|k-1)) \\
\hat{x}(k+2|U_k) &= x(k+1) + F(k+1) + A(k+1)(\tilde{x}(k+1) - \tilde{x}(k+1)) \\
&+ B(k+1)(u(k+1) - u(k+1|k-1)) \\
&= \tilde{x}(k+2) + A(k+1)(B(k)(u(k) - u(k|k-1))) \\
&+ B(k+1)(u(k+1) - u(k+1|k-1))
\end{align*}
\] (3.23)

To simplify notation, we introduce

\[
\prod(i, n) = \begin{cases} 
A(k + i - 1)A(k + i - 2) \cdots A(k + n + 1) & \text{when } i > n + 1 \\
I & \text{when } i = n + 1 \\
0 & \text{when } i < n + 1 
\end{cases}
\] (3.24)

where \( I \) is an identity matrix. This makes it possible to state the predictor.
\[
\hat{x}(k + i | U_k) = \tilde{x}(k + i) + \sum_{n=0}^{i-1} (i, n) B(k + n) (u(k + n) - u(0))
\]  

(3.25)

The vector \( Y \) containing the output signals for all the states can be derived from the predictor

\[
Y = C \hat{X} = H + S(U_k - U_{k-1})
\]  

(3.26)

which in accordance to Equation 3.8 and 3.10 are put in the optimization problem 3.3 in the QP form.

\[
H = \begin{pmatrix}
C \hat{x}(k + 1) \\
C \hat{x}(k + 2) \\
\vdots \\
C \hat{x}(k + N)
\end{pmatrix}
\]

\[
S = \begin{pmatrix}
CB(k) & 0 & \ldots & 0 \\
C \prod_{i=1}^{N} B(k) & CB(k + 1) & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
C \prod_{i=1}^{N} B(k) & C \prod_{i=N}^{N} B(k + 1) & \ldots & CB(k + N - 1)
\end{pmatrix}
\]  

(3.27)

### 3.3 The Direct Method

In the Direct method, the infinite dimensional optimal control problem is converted into a finite dimensional non-linear programming problem so this is also a MPC problem. In this optimization method, we want to find the control input \( u(t) \) that minimizes the cost function, \( J(u) \)

\[
\min_u J(u) = \int_t^{t+T} y^T(\tau)Qy(\tau) + u^T(\tau)Ru(\tau)d\tau
\]

subject to

\[
\dot{x}(t) = f(x(t), u(t)) \\
h(u, y) \leq 0
\]

(3.28)

\( x(t) \) are the state variables in the time continuos non-linear model as in Equation 3.10. The time interval \( T \) is discretized into \( N \) time nodes and an initial guess is made on the control sequence \( u(t) \) parametrized by the nodes \( u_i(t) \) where \( i = 0, 1, \ldots, N - 1 \). The parametrization can be made in linear, cubic and spline manners for example.

\[
u_i(t) = \left( u(t), u(t + \frac{T}{N-1}), \ldots, u(t + T) \right)
\]

(3.29)
To be able to find the optimal control sequence, the cost function, $J(u)$, is differentiated numerically. This is done by changing one part of the control sequence at a time. First, $u_1$ is changed into $u_1 + \delta$, where $\delta$ is an very small constant. The cost function is calculated again, giving $J(u_1 + \delta)$. The gradient with respect to $u_i$ can then be approximated by the difference of the cost functions

$$\frac{d}{du_1}J(u) \approx \frac{J(u_1 + \delta) - J(u_1)}{\delta}$$ \hspace{1cm} (3.30)

This is done for all the elements in $u_i$ so that the search direction can be found for the entire control sequence. When the differentiations of the control sequence are done, the new control sequence can be calculated according to Equation 3.31 where $H$ is the approximated hessian containing the search direction and $\alpha$ is a constant obtained by line search (see [12] for Hessian approximations and line search).

$$u := u + H^{-1} \frac{d}{du}J(u)\alpha$$ \hspace{1cm} (3.31)

Constraints are included in the algorithm by adding a penalty to the cost function. When the constraint $h(u, y) \leq 0$ is satisfied, the penalty is 0, but when it is not, a quadratic penalty is used. The new cost function is calculated according to Equation 3.32 where $\mu \to 0$ (see [12], Chapter 15).

$$J_{\text{new}} = J + \frac{1}{\mu} \sum_i max(0, h(u))^2$$ \hspace{1cm} (3.32)

A more thoroughly analysis of the Direct Method can be found in [5].
4 MPC for Trajectory Optimization

The aim of the optimization is to find a control sequence that minimizes the total height over the ground and also avoids exceeding constraints on the acceleration and jerk of the aircraft. First, a reference trajectory is obtained using the terrain database, over a finite horizon. The problem then lies in finding an optimal control sequence which gives a trajectory that follows the reference trajectory as closely as possible without getting below it. In MPC, the control problem is solved at each sample instant and the first part of the control sequence is applied to the actual aircraft.

From the in beforehand decided starting and ending points and the terrain database, the reference height is found for all the states. The reference trajectory is set at e.g. 50 meters over the ground. The optimal flight trajectory is then calculated for the prediction horizon, e.g. 3000 meters. The first control signal of the optimized control sequence is sent to the aircraft’s control system. A new optimization is made at the next sample instant to obtain a new control signal. This goes on until the aircraft has reached the final point of the flight interval.

The optimization problems will be presented in this chapter, both in the case when the aircraft is considered to be a point mass and when the aircraft dynamics are included in the model. First there will be an explanation of the states that will be used and after that, the actual optimization problems will be explained.

4.1 The MPC Algorithm

The algorithm that will be used in this chapter has the following appearance

1. Measure $x(k|k)$ (from the aircraft)
2. Make an initial guess on the control sequence $U_{k-1}$ (the previously calculated control sequence) if it is not calculated in step 8 already
3. Linearize along $U_{k-1}$
4. Obtain the reference values along the linearized trajectory
5. Find an optimal $U_k$ (the new control sequence) by solving the optimization problem
6. If the output $Y$ is good enough, go to step 7. Else, go back to step 3 with $U_{k-1} = U_k$
7. Apply $u(k|k)$
8. Time update
   $U_{k-1} = (U_k(2), U_k(3), ..., U_k(N), U_k(N))$
   $k := k + 1$
9. Go to 1
4.2 Control Variables

The control signals for finding the optimal trajectory are the accelerations in two directions and the rotation of the aircraft (the rotation is only used when the aircraft dynamics are considered). As explained in Section 2.2.1, instead of using both the acceleration and the rotation as control signals, only the load factor in the $z$ direction is used.

4.3 Linear Point Mass Model

The algorithm finds the best flight trajectory along a straight line from the starting point to the ending point. Only the acceleration of the aircraft in the $z$ direction is necessary to find when the aircraft is considered to be a point mass since the velocity forward is set constant and there are no velocities or accelerations at all to the sides. In this case, the B and L frame coincide so the only states that are interesting are the velocity, $v_d$, and the altitude of the aircraft. The model is linear and becomes

$$\begin{align*}
\dot{h}(t) &= -v_d(t) \\
\dot{v}_d(t) &= a_d(t)
\end{align*}$$

(4.1)

The output signal, $y$, is the altitude of the aircraft and the states and the control signals are

$$\begin{align*}
x &= \begin{pmatrix} h \\ v_d \end{pmatrix} \\
u &= a_d
\end{align*}$$

(4.2)

The equations for the model are

$$\begin{align*}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &= Cx(t)
\end{align*}$$

(4.3)

The height is the only output that is interesting so $C$ becomes

$$C = \begin{pmatrix} 0 & 1 \\ \end{pmatrix}$$

(4.4)

4.4 Non-linear Model with Aircraft Dynamics

As will be described in Sections 4.5, the constraints on the aircraft are set in the body frame. To be able to deliver the actual position of the aircraft, the states latitude, longitude and height will be stated in the local level frame. Since there have to be transformations between coordinate frames B and L during the optimization of the flight trajectory, the model becomes non-linear. The non-linear model of the aircraft is built up according to the following equations

$$\begin{align*}
\dot{x}(t) &= f(x(t), u(t)) \\
y(t) &= Cx(t)
\end{align*}$$

(4.5)
where the states and the control signal are

\[
x(t) = (l(t), L(t), h(t), v_z(t), \theta(t), q(t), \delta_e(t), \delta_c(t))^T
\]
\[
u(t) = n_z(t)
\]

The states and control signal for this model are already explained in Section 2.2. They are: longitude, latitude, altitude, velocity in the \(z\) direction, pitch angle, rotation in the \(y\) direction, and the elevator and canard deflections. The control signal is the load factor in the \(z\) direction. In this algorithm, the velocity in the velocity vector direction, \(\nu_{\text{tot}}\), will always be constant so \(\nu_x\) will vary when \(\nu_z\) varies. It is not necessary to include \(\nu_x\) as a state since it can be calculated from \(\nu_{\text{tot}}\) and \(\nu_z\) according to

\[
\nu_x(t) = \sqrt{\nu_{\text{tot}}^2 - \nu_z(t)^2}
\]

The actual model becomes

\[
\begin{align*}
\dot{l}(t) &= T_{\text{mdeg}} v_n(t) = T_{\text{mdeg}} \cos \psi (\nu_x(t) \cos \theta(t) + \nu_z(t) \sin \theta(t)) \\
&= T_{\text{mdeg}} \cos \psi (\sqrt{\nu_{\text{tot}}^2 - \nu_z(t)^2} \cos \theta(t) + \nu_z(t) \sin \theta(t)) \\
\dot{L}(t) &= T_{\text{mdeg}} v_e(t) = T_{\text{mdeg}} \sin \psi (\nu_x(t) \cos \theta(t) + \nu_z(t) \sin \theta(t)) \\
&= T_{\text{mdeg}} \sin \psi (\sqrt{\nu_{\text{tot}}^2 - \nu_z(t)^2} \cos \theta(t) + \nu_z(t) \sin \theta(t)) \\
\dot{h}(t) &= v_d(t) = -\nu_x(t) \sin \theta(t) + \nu_z(t) \cos \theta(t) \\
&= \sqrt{\nu_{\text{tot}}^2 - \nu_z(t)^2} \sin \theta(t) + \nu_z(t) \cos \theta(t) \\
\dot{\nu_z}(t) &= a_{44} \nu(t) + a_{45} q(t) + a_{46} \theta(t) + a_{47} \delta_e(t) + a_{48} \delta_c(t) + b_4 n_z(t) \\
\dot{q}(t) &= a_{54} \nu(t) + a_{55} q(t) + a_{56} \theta(t) + a_{57} \delta_e(t) + a_{58} \delta_c(t) + b_5 n_z(t) \\
\dot{\theta}(t) &= q(t) \\
\dot{\delta}_e(t) &= a_{74} \nu(t) + a_{75} q(t) + a_{76} \theta(t) + a_{77} \delta_e(t) + a_{78} \delta_c(t) + b_7 n_z(t) \\
\dot{\delta}_c(t) &= a_{84} \nu(t) + a_{85} q(t) + a_{86} \theta(t) + a_{87} \delta_e(t) + a_{88} \delta_c(t) + b_8 n_z(t)
\end{align*}
\]

The \(a\) and \(b\) coefficients in Equation 4.8 are constant and depend on the dynamics of the aircraft. The indices describe the rows and the columns at which they are placed in the matrix. The output will also in this case be the height so \(C\) becomes

\[
C = (0 \, 0 \, 1 \, 0 \, 0 \, 0 \, 0 \, 0)
\]

### 4.5 Constraints on the Accelerations and Rotations

The pilot can manage certain accelerations, but there have to be limits set on them and also on how much the accelerations are allowed to change. The limits are set in the \(B\) frame since that is the frame that moves with the pilot. In the optimization, only the load factor in
the \( z \) direction will be used as a control signal. This gives both an acceleration in the \( x \) and \( z \) direction due to the rotation of the aircraft. Limits on the acceleration in the \( x \) direction will be neglected. The reason for that is that the accelerations in the \( x \) direction are always within their boundaries. The rotation \( q \) does not change much during the flight so there are no constraints set on it.

In Equations 4.10 - 4.12, \( g \) is 9.8066 \( \text{m/s}^2 \) and \( T \) is the sample time for the model.

\[
-2g \leq a_x \leq 2g \text{m/s}^2 \tag{4.10}
\]

\[
-5.5g \leq a_z \leq 0.5g \text{m/s}^2 \tag{4.11}
\]

The change of the acceleration, jerk, in any direction must lie within

\[
-0.4g \leq a(t) \leq 0.4g \text{m/s}^3 \tag{4.12}
\]

If the change lies outside these constraints, the flight gets very uneven and uncomfortable for the pilot.

### 4.6 Trajectory Linearizations

The distance to the reference trajectory should be as small as possible so it should be part of the minimization problem and is reflected by the weight matrix \( Q \). The control signal’s magnitude should not be too high either so it is a part of the problem too and is reflected by the weight matrix \( R \). Quadratic programming is used for limiting both too big positive and negative values of the control signal. Furthermore, constraints will be set on the distance between the reference trajectory and the actual trajectory so that the aircraft never goes below the allowed limit. If those constraints are not used, the optimized flight trajectory will follow the reference trajectory as near as possible and it will at times go below the reference trajectory \( (w) \).

The problem we want to minimize is

\[
\min_u \sum_{j=k}^{k+N-1} (y(j|k) - w(j))^T Q (y(j|k) - w(j)) + u(j|k)^T R u(j|k);
\]

subject to

\[
u(k + j|k) \in U
\]

\[
y(k + j|k) \geq w(k + j)
\]

\[
(4.13)
\]

#### 4.6.1 Constraints Formulated for QP

The flight trajectory always has to lie above the reference trajectory. To ensure that, the constraint \( Y - W \geq 0 \) has to be met \( (Y \) is the actual flight trajectory and \( W \) is the reference trajectory). According to Section 4.5, there are also constraints on the acceleration and
jerk in the \( z \) direction. These constraints must be part of the minimization problem and are rewritten below so that they can be formulated for the QP problem. First, we recall the predicted outputs, 3.26

\[
Y = H + S(U_k - U_{k-1})
\]

The constraints on \( Y - W \) can now be written as

\[
Y - W = H + S(U_k - U_{k-1}) - W \geq 0
\]

This yields constraints on \( U_k \) which can be put in Equation 3.9

\[
-SU_k \leq H - W - SU_{k-1}
\]  \hspace{1cm} (4.14)

Each control signal in \( U_k \) has to lie within the boundaries in 4.11. The constraints on the sequence are set according to

\[
U_{LB} \leq U_k \leq U_{UB} \quad \Rightarrow
\]

\[
-U_k \leq -U_{LB}
\]

\[
U_k \leq U_{UB}
\]  \hspace{1cm} (4.15)

where

\[
U_{LB} = \begin{pmatrix} -5.5g \\ \vdots \\ -5.5g \end{pmatrix} \quad \text{and} \quad U_{UB} = \begin{pmatrix} 0.5g \\ \vdots \\ 0.5g \end{pmatrix}
\]  \hspace{1cm} (4.16)

The constraints on the jerk are in the discrete time case written as

\[
-0.4gT \leq u_{i+1} - u_i \leq 0.4gT \quad \Rightarrow
\]

\[
JU_k \leq j , \quad \text{where}
\]  \hspace{1cm} (4.17)
The constraints above can be put together to a constraint $EU_k \leq e$ where $I$ is an identity matrix of the size $N \times N$

$$E = \begin{pmatrix} -I \\ I \\ -S \\ J \end{pmatrix} \text{ and } e = \begin{pmatrix} -U_{LB} \\ U_{UB} \\ H - W - SU_{k-1} \end{pmatrix}$$ (4.20)

### 4.6.2 Quadratic Programming

According to Equations 3.9 and 3.26, the optimization problem 4.13 can now be written as

$$\begin{array}{c}
\min_{U_k} \quad V(U_k) = (Y - W)^T Q_{new} (Y - W) + U_k^T R_{new} U_k \\
= (H + SU_k - SU_{k-1} - W)^T Q_{new} (H + SU_k - SU_{k-1} - W) \\
+ U_k^T R_{new} U_k \\
= \ldots + U_k^T (S^T Q_{new} S + R_{new}) U_k + 2(H^T - U_{k-1}^T S^T - W^T) Q_{new} SU_k
\end{array}$$

subject to $EU_k \leq e$

(4.21)

The two last parts of the problem 4.21 can now be put in the QP algorithm as in Equation 3.9. The resulting optimal $U_k$ in the QP algorithm is the optimal solution for the MPC algorithm.
4.6.3 Implementation Issues

After optimizing the trajectory, the states longitude and latitude along the whole trajectory might have changed since \(v_x\) is not constant. The values of the altitude along the reference trajectory depend on the longitude and latitude so they are no longer accurate. For the evaluation of the quality of the optimized trajectory, the new reference trajectory has to be obtained by reading from the terrain database with the new values of \(l\) and \(L\). This corresponds to step 4 in the algorithm description.

The quality of the trajectory is calculated (step 6 in the algorithm description) and if it is good enough, the control signal is sent to the control system of the aircraft (step 7). If the quality is bad, a new iteration of the QP algorithm is done. The quality of the optimized trajectory is calculated by comparing the mean level above the reference trajectory with the mean level calculated for the previously optimized trajectory. If the difference between these two is less than 0.1 meters, the optimized trajectory is considered good enough. If the difference is more than 0.1 meters, the optimization algorithm is run again with the previously calculated control sequence as input (step 3). QP problems can always be solved in a finite number of iterations, so the trajectory will converge towards a solution.

When the optimization for the sample instant is done, a control signal is sent to the aircraft’s control system (step 7 and 8) and the optimization for the next sample begins. The vector \(U_k\) contains the control signals for the whole horizon but it is only the first control signal that is sent to the control system in step 7. To decide the initial value of \(U_{k-1}\) for the next sample instant, the previously calculated \(U_k\) is used. The first value of \(U_k\) is excluded and the rest of the vector is put in \(U_{k-1}\) according to

\[
U_{k-1} = (U_k(2), U_k(3), \ldots, U_k(N), U_k(N))
\]

(4.22)

4.6.4 Blocking

When the horizon gets very long, the calculation time for finding the optimal trajectory grows rapidly. To decrease the calculation time, blocking, described in this section, of the control signal can be used. The first 20 control signals are the interesting ones and the rest of them are just there to make sure that the future trajectory lies above the reference trajectory. Therefore, a simplification of the control sequence can be made so that the control signals beyond 20 samples are decided less often. Beyond the 20th sample, the control signals are blocked together so that the same control signal is set for three sample instants in a row. In that way, the sequence \(U_k\) (size \(K \times 1\)) can be reduced to a shorter sequence, \(V_m\) (size \(M \times 1\))

\[
V_m = (U_k(1), U_k(2), U_k(3), \ldots, U_k(20), U_k(21), U_k(24), U_k(27), \ldots, U_k(N))
\]

(4.23)

The relationship between the vectors is expressed in Equation 4.24 where \(\Omega\) is a \(K \times M\) matrix.
\[ U_k = \Omega V_m \]  \hspace{1cm} (4.24)

with

\[ \Omega = \begin{bmatrix}
1 & 0 & 0 & \ldots & 0 & 0 & \ldots & 0 \\
0 & 1 & 0 & \ldots & 0 & 0 & \ldots & 0 \\
0 & 0 & 1 & \ldots & 0 & 0 & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \ldots & 1 & 0 & \ldots & 0 \\
0 & 0 & 0 & \ldots & 1 & 0 & \ldots & 0 \\
0 & 0 & 0 & \ldots & 0 & 1 & \ldots & 0 \\
0 & 0 & 0 & \ldots & 0 & 1 & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \ldots & 0 & 0 & \ldots & 1 \\
0 & 0 & 0 & \ldots & 0 & 0 & \ldots & 1 \\
0 & 0 & 0 & \ldots & 0 & 0 & \ldots & 1 \\
\end{bmatrix} \]  \hspace{1cm} (4.25)
For the actual implementation of the Direct Method in the MPC algorithm, not many changes have to be done from the basic algorithm in Section 3.3. Now we want to minimize the distance between the reference trajectory, \( w(t) \), and the output signal, \( y(t) \), so the optimization problem 3.28 becomes

\[
\begin{align*}
\min_u & \quad J(u) = \int_t^{t+T} (y(\tau) - w(\tau))^T Q (y(\tau) - w(\tau)) + u(\tau)^T R u(\tau) d\tau \\
\text{subject to} & \quad \dot{x}(t) = f(x(t), u(t)) \\
& \quad y(\tau) - w(\tau) \leq 0 \text{ for } t \leq \tau \leq t + T
\end{align*}
\]

(5.1)

The model for the system is defined as in Section 4.3

\[
\begin{align*}
\dot{h}(t) &= -v_d(t) \\
\dot{v}_d(t) &= a_d(t) \\
x &= \begin{pmatrix} h \\ v_d \end{pmatrix} \\
u &= a_d
\end{align*}
\]

(5.2)

(5.3)

This yields the model

\[
\begin{align*}
\dot{x}(t) &= A x(t) + B u(t) \\
y(t) &= C x(t)
\end{align*}
\]

(5.4)

The height is also in this case the output signal that is interesting so \( C \) becomes

\[
(5.5)
\]
6 Simulations

The simulations of the problems are done in Matlab and are presented in this chapter. Different coordinate frames for representing the earth can be used, e.g. RT90 and WGS-84. In the terrain database, the coordinates are relative to the WGS-84 reference frame so this is the frame used throughout the whole thesis. The simulations are done for intervals that are approximately 30 kilometers long in three different terrain types in Sweden, Motala (flat), Västergötland (varying) and Kisa (very varying).

Section 6.1 shows some examples of when different parameters in the MPC algorithm are varied for the linear point mass model to see their impact on the system. In that section, QP is used to solve the optimization problem but in Section 6.2, the Direct Method is used instead. In Section 6.4, the aircraft dynamics are added to the model and the MPC algorithm is run for the non-linear model.

6.1 Linear Point Mass Model

In this section, the MPC algorithm will be tried out. The aircraft is throughout Section 6.1 considered to be a point mass so the dynamics are not considered. The velocity in the $x$ direction will be 270 m/s for all the simulations.

Different parameters ($N$, $T$, $Q$ and $R$) are changed in the different simulations to see the impact of them in the MPC controller. The goal is to get a flight trajectory that lies as near the reference trajectory as possible but it is also important that the optimization is not too time consuming. During this whole section, simulations are run without using the blocking presented in Section 4.6.4.

The sample time is the time period for which each control signal is commanded by the control system. The new trajectory has to be calculated before a new control signal is to be commanded. If we want to have the same control signal for 150 meters, and the velocity is 270 m/s, the sample time becomes

$$T = \frac{150}{270} = 0.56 \text{ seconds}$$

The time slot for calculating and accepting a trajectory has to be less than $T$ since the algorithm works in real-time. If the prediction horizon is chosen to be 2900 meters and the sample interval still is 150 meters, the horizon becomes

$$N = \text{ceil}\left(\frac{2900}{150}\right) = 20$$

where $\text{ceil}$ rounds the quotient upwards to the nearest integer.
6.1.1 Variation of the Prediction Horizon

The following section describes how to choose the horizon for the optimization. To make sure that the prediction horizon is chosen properly, it is chosen for two different terrain types.

If the horizon is shorter than, say 1000 meters, the algorithm does not account hills that are further away than 1000 meters. When the optimization for the next interval starts, the aircraft might have such a low altitude that it has to ascend way above the reference trajectory to be able to avoid hills that now are within range. The reason for why the aircraft ascends so much more than necessary is that the acceleration upwards gets very high and it takes a while to be able to lower it again. The problem can easily be avoided by extending the horizon to e.g. 3000 meters. Now, the compensation for hills ahead will be made by starting the acceleration upwards earlier. In this way, the aircraft can start descending when it coincides with the highest point of the reference trajectory.

As the prediction horizon gets longer, a better flight trajectory is obtained which makes sure that the aircraft does not ascend too much anywhere. Simulations show that a horizon of 3000 meters is good enough in this case. If the horizon is made longer, the optimization will be much more time consuming so it is important not to make the horizon too long.

Case 1 (Motala). For the first case, an interval of the terrain is chosen when the terrain does not rise more than 0.11 meter each meter. The aircraft begins the flight in the point (58.6, 16.1) and ends it in the point (58.34, 16.1) in the WGS-84 coordinate frame. The algorithm is run for three different horizons, 1.65, 3 and 4 kilometers (N = 11, 20 and 27) and the sample time is set to 0.56 seconds (150 meters).

![Figure 6: Trajectory obtained using MPC and the prediction horizon is 1650 meters. The mean level above the reference trajectory is 7.3 meters. (Motala)](image-url)
Figures 7 and 10 show the acceleration while Figures 8 and 11 show the jerk in the two cases. As can be seen, the acceleration and jerk always lie within their boundaries.

**FIGURE 7.** Acceleration in the $d$ direction when the prediction horizon is 1650 meters (Motala).

**FIGURE 8.** Jerk in the $d$ direction when the prediction horizon is 1650 meters (Motala).
6.1 Simulations, Linear Point Mass Model

**FIGURE 9.** Trajectory obtained using MPC and the prediction horizon is 3000 meters. The mean level above the reference trajectory is 6.4 meters. (Motala)

**FIGURE 10.** Acceleration in the d direction when the prediction horizon is 3000 meters (Motala).
When the horizon is 1.65 kilometers, the mean distance between the optimized trajectory and the reference trajectory becomes 7.8 meters. The mean distance to the reference trajectory is 6.4 meters both when the horizon is 3 and 4 kilometers. Since the mean distance stays approximately the same as the horizon is extended beyond 3 kilometers, this horizon is chosen. We do not want to have a too time consuming algorithm and it is more important to have a relatively fast optimization than to lower the mean level a few centimeters extra. The calculation time for each sample instant for the three horizons are 0.63, 0.75 and 1.2 seconds respectively when the simulations are done in Matlab.

It can clearly be seen in the 1.65 and 3 kilometer cases that when the horizon gets extended, the optimization becomes less local. When a bigger hill is within reach, compensation for it is done by making the aircraft ascend earlier. When the horizon is long enough, the compensation is so good that the aircraft is on its way down when the top of the hill is reached. Even though the aircraft ascends so it lies a little bit more over the reference trajectory in the second case, the total time of being seen is reduced.

Case 2 (Västergötland). In the Västergötland terrain used for this simulation, the terrain is very varying and horizons of 2.5, 3 and 4 kilometers are tried out. In Figures 12 and 14, the horizon is 2.5 and 3 kilometers respectively \((N = 17 \text{ and } 20)\). The aircraft begins the flight in the point \((59.55, 14.23)\) in the WGS-84 coordinate frame at an altitude of 300 meters above the sea level and it ends it in the point \((59.55, 14.58)\).

Also in this case, the best prediction horizon becomes 3 kilometers. The simulation when the horizon is 2.5 kilometers gives a flight trajectory which at times ascend way above the reference trajectory while a horizon of 4 kilometers gives approximately the same result as an horizon of 3 kilometers. The medium height over the reference trajectory is 8.0 meters when the horizon is 2.5 kilometers and 7.0 m when the horizon is 3 kilometers.
FIGURE 12. Trajectory obtained using MPC and the prediction horizon is 2500 meters. The mean level above the reference trajectory is 8.0 meters. (Västergötland)

FIGURE 13. Acceleration in the d direction when the prediction horizon is 2500 meters (Västergötland).
FIGURE 14. Trajectory obtained using MPC and the prediction horizon is 3000 meters. The mean level above the reference trajectory is 7.0 meters. (Västergötland)

FIGURE 15. Acceleration in z direction when the prediction horizon is 3000 meters (Västergötland).
The cases above show that the horizon can be chosen around 3 kilometers. It can be increased to give a little better trajectory or to give a better safety margin but 4 kilometers will be enough. In the cases in this section, the sample interval has been 150 meters, so the horizon is $N = 27$ when the horizon is 4 kilometers.

### 6.1.2 Variation of the Sample Time

In the following figures, the sample time is varied. The optimization take up very much time so by increasing the sample time, the simulation is done much faster than before. The different sample times tested are 0.18, 0.37, 0.56 and 0.74 seconds. These sample times are calculated from the velocity, and the desired sample intervals (50, 100, 150, 200 meters).

$$T = \frac{\text{desired sample interval}}{v_x}$$  \hspace{1cm} (6.3)

The figures below show the predicted flight trajectories for the different cases. As the sample time gets higher, the reference trajectory gets more evened out and the fine structures disappear. This is of course at times desirable since the flight then becomes smoother for the pilot since the acceleration is not changed that much. Instead of descending at smaller hollows, the aircraft now flies straight forward. In this thesis the objective is to lie as near the ground as possible though, and that task becomes more difficult when the reference trajectory differs too much from the terrain. Therefore, the sample interval should not be longer than 200 meters (0.74 seconds). The prediction horizon is for the following figures 3000 meters.
FIGURE 17. The sample time is $T = 0.2$ s (50 m) (Motala).

FIGURE 18. The sample time is $T = 0.37$ s (100 m) (Motala).
The time it takes for the algorithm to simulate the optimal trajectory in Matlab is 73 minutes (7 seconds for each sample) when $T = 0.2$ seconds. When $T = 0.74$ seconds, the simulation time is 2 minutes (0.2 seconds for each sample). The sample interval 150 meters ($T = 0.56$ seconds) seems to be a good choice since the algorithm is relatively fast and most of the variations in the terrain are included in the reference trajectory.
6.1.3 Constraints

In this section, the algorithm is run without the constraints $Y - W \geq 0$. This is the constraint that keeps the optimized trajectory above the reference trajectory.

When the constraints $Y - W \geq 0$ are ignored, and a very big weight, $Q$, is set at instances when the terrain is at its peaks, the total height over the flight trajectory gets even lower. The disadvantage with this method is that the flight trajectory lies below the reference trajectory at times. Figure 21 shows the flight trajectory for the same terrain interval as in Figure 19. As seen in the figure, this type of optimization cannot be trusted completely. The optimized trajectory follows the reference trajectory rather well but it often falls below it. The trajectory could be used as a rough estimate for the pilot so that he can see where to steer. He still has to analyze the terrain himself though to make the final control of the trajectory.

The advantage of not using the constraint $Y - W \geq 0$ is that the algorithm gets faster that way. As the number of constraints are reduced, the algorithm gets more rapid and this can sometimes be more desirable than finding the perfect trajectory.

**FIGURE 21. There are no constraints on the distance to the reference trajectory (Motala).**
6.1.4 Weights

In this section, we will try to find out how the weights $Q$ and $R$ affect the obtained trajectories. When the weight $Q$ is varied along the horizon, the trajectory does not improve considerably. Different methods have been used for finding a better way of deciding the weight but the result is that $Q$ should be chosen equal for all states.

It is more interesting to see what happens when the weight $R$ is varied compared to $Q$. If the weight on the control signal is increased, a more even flight trajectory is found. This gives a smoother trajectory and the flight gets more comfortable for the pilot. Depending on how important it is to fly as near the ground as possible, different weights can be used. The following figure shows the trajectory when $R$ is than same magnitude as $Q$ and when it is 10 and 100 times greater. Figures 23 and 24 show how the acceleration and jerk are changed to get a more smooth appearance. The mean level over the reference trajectory increases when $R$ gets higher. In the three cases in Figure 22, the mean level is 7.0, 8.2 and 11.3 meters.

When the trajectory gets higher above the reference trajectory, it does not take as much effort to satisfy the constraints so the calculation time gets much faster. For each sample, the simulation takes 0.54, 0.28 and 0.22 seconds respectively, so it saves calculation time when the weight $R$ is increased.

![Figure 22](image_url)

**FIGURE 22. The optimized trajectories when the weight $R$ is varied (Motala).**
FIGURE 23. The acceleration in the d direction when the weight R is varied (Motala).

FIGURE 24. The jerk in the d direction when the weight R is varied (Motala).
6.2 The Direct Method

The simulation of the problem using the Direct Method for the optimization was not very successful since the algorithm used did not manage a discretization of higher order than \( N = 5 \). This problem is not solved here. Figure 25 shows the result of the simulation.

The Direct Method is quite time consuming so it would probably not be possible to implement the algorithm in the aircraft with the present computer but it might be interesting to examine the algorithm further to see the quality of the optimized trajectory compared to the QP solution of the MPC problem.

![Figure 25. The optimized trajectory for the Direct Method when \( N = 5 \) (Motala).](image)

6.3 Comparison

It is difficult to compare the QP solution with the Direct Method solution of the MPC problem since only the QP solution was implemented fully. The QP solution is less time consuming than the Direct Method solution in this thesis. If a better algorithm for the Direct Method was found, it would be even more time consuming but it would probably give a better trajectory. In the time consumption aspect, the QP solution is preferable right now for an aircraft with the current computer capacity.

The algorithm in this thesis can be compared with algorithms in other reports that are based on a linear model of the aircraft. In [7], an algorithm is found which yields the mean distance to the reference trajectory 26.5 meters. When the same sample time and interval is used with the MPC algorithm in this thesis, the mean level is 12.7 as seen in Figure 26.
The sample time is here 0.2 and the horizon is 4 kilometers. The terrain interval used for this example is in Kisa. The aircraft starts in the point (58.0825, 15.1641) and end the flight in the point (58.0825, 15.55).

The algorithm in [7] is also based on MPC, but it tries out different ways (the best one is Dynamic Programming) of making the actual optimization. This method is more time consuming. Since there are no constraints on the jerk in the algorithms either, it gets too high at times. In the algorithm in this thesis, there are constraints set on the jerk, so it never gets too high.

![Figure 26](image.png)

**FIGURE 26.** Optimized trajectory when flying over the Kisa terrain (Kisa). The mean level above the reference trajectory is 12.5 meters.

### 6.4 The Non-linear Model

When the aircraft dynamics are included in the algorithm, the calculations take a little bit more time since this is a 8 dimensional system. When using a prediction horizon of 4 kilometers the calculation time for each sample interval is 1.9 seconds (which can be compared to 1.2 seconds in the point mass case). As a comparison, Figure 27 shows the optimized trajectory when the terrain is very varying. The simulation is in this case over the Västergötland terrain as in Figures 12 and 14. The aircraft flies with a constant velocity of 270 m/s in the $x$ direction in the stability frame.

The constraints on the acceleration and the jerk are in the non-linear case set in the B frame (instead of in the L frame as in the linear case) so the aircraft ascends a little bit slower than before. Therefore, the prediction horizon has to be extended to 5 kilometers at least. Blocking of the control signal is in this chapter used since the horizons are extended and the calculation time gets too long otherwise.
FIGURE 27. Trajectory obtained using MPC and the prediction horizon is 5000 m and the terrain is rather varying (Västergötland). The aircraft dynamics are included in the algorithm and the mean level above the reference trajectory is 6.6 meters.

Figure 28 shows the optimized trajectory in the case when the terrain is rather flat (Motala).

FIGURE 28. Trajectory obtained using MPC and the prediction horizon is 4000 m and the terrain is rather flat (Motala). The aircraft dynamics are included in the algorithm and the mean level above the reference trajectory is 7.6 meters.
The figures show that the algorithm can be implemented also when the aircraft dynamics are added to it. In this case, it is even more interesting to study the control signal. The figures below show that the acceleration and the jerk lie within their boundaries for both of the cases above. Even though there are no constraints set on $\alpha_x$, it still lies within its boundaries.

**FIGURE 29.** Acceleration and jerk in the Västergötland case.

**FIGURE 30.** Acceleration in the x direction in the Västergötland case.
6.5 Simulations, Sensitivity and Robustness of the Model

Apparently, the MPC problem can be solved with QP, giving a good result even when the model is non-linear and the algorithm never makes the control signal exceed its constraints. When the horizon gets longer, the algorithm gets very time consuming. By using blocking of the control signal, the calculation time is reduced a lot. When the blocking method is used, the number of free variables is reduced from 34 to 26 when the horizon is 5 kilometers. The calculation time for each QP optimization iteration is reduced from 15 seconds to 5 seconds, so this is a good improvement of the algorithm when the optimized trajectory still is of as good quality as before.

6.5 Sensitivity and Robustness of the Model

There is always noise acting on the aircraft, and the model might not always agree with the model. This chapter will present how good the algorithm is at compensation for errors like this.

6.5.1 Sensitivity

There exists turbulence that disturbs the model and even if the aircraft compensates for it very good, an analysis of its effect is interesting. It is important to know how the control system carries out its task when it is affected by noise. To examine the sensitivity, noise is added to the velocity $v_{tot}$ according to Equation 6.4. ($\text{randn}(1)$ gives a random number with the variance 1). The noise is the size of 10\% of the velocity.

$$v_{tot} := v_{tot} + 27 \text{randn}(1)$$  \hspace{1cm} (6.4)

When optimizing over an interval of terrain, the optimized trajectories do not differ that much. They still have a similar look even though the noise affects the trajectory. The opti-
mized trajectory still lies over the reference trajectory and even if the trajectory is not perfect, it can be still be considered a pretty good optimization.

**FIGURE 32. The optimized trajectories with and without noise added to the velocity (Motala).**

### 6.5.2 Robustness

The model differs from the true system of the aircraft but for accelerations within \(-2.5g \leq a \leq 2.5g\), it can be considered a good simplification of it. The control signal never exceeds these limits so the model can be considered rather true but when higher accelerations are desired, a more complex model has to be used.

When the appearance of the model is disturbed a little, the algorithm should still be able to give a good solution. It is interesting to examine the robustness of the model to see what would happen if the model would differ from the true system. The coefficients in the model are varied and an optimal trajectory is calculated for the true model and the modified model. When the different coefficients are varied, the optimized trajectory should not vary much if the model is to be considered robust.

The big difference when using the model which is altered with is that there have to be more iterations in each sample instant, so the algorithm gets more time consuming. Instead of a maximum of two iterations in each sample instant, there are now up to seven iterations. No matter what and how many coefficients in the model that are altered with, the trajectory still looks pretty accurate. The figure below shows the trajectory when the
model is changed compared to the trajectory obtained from the original model. There is not much difference between the trajectories even though the model is varied quite a bit so the model can be considered very robust.

FIGURE 33. Robustness test of the aircraft model (Motala).
7 Results

According to Section 6.3 - 6.5, the MPC algorithm can be used for finding the optimal flight trajectory for the aircraft. In the following section, the optimal parameters found in Chapter 6 are stated and analyzed. The parameters that are found depend on the velocity, $v_{tot}$, which has been chosen to be 270 m/s in all the simulations.

7.1 Prediction Horizon

The needed prediction horizon does not depend much on the type of terrain. Both when the terrain is varying and not, the prediction horizon can be chosen to 3 kilometers in the linear case. If the prediction horizon is too short, hills are not discovered until it is too late to make a nice acceleration over the hill. Instead, the aircraft has to accelerate a lot and in the long run, this is very uncomfortable for the pilot. When the aircraft accelerates too much, it takes quite a while to adjust the velocity and get the desired altitude again. When the aircraft dynamics are included in the model and it becomes non-linear, a prediction horizon of 5 kilometers is needed.

It is not desirable to have a too long horizon either since the calculation time gets very high then. When the mean level over the reference trajectory is not lowered more than a decimeter when the horizon is lengthened a kilometer, it is better to use the shorter horizon. As a rule of thumb, a horizon of 5 kilometers is always enough to give a nice flight trajectory and it can usually be lower.

Since the optimization gets very time consuming when the horizon gets longer, blocking can be used on the last kilometers. In this way, the control sequence is shortened and the optimization gets faster.

7.2 Sample Time

It is crucial to choose a sample time which is not to low since it decides together with the prediction horizon how time consuming the algorithm will be. The terrain database gives the height of the terrain for every 50 meters so there is no reason for using sample interval lower than that. When the sample interval gets higher than 200 meters, the reference trajectory gets less detailed. The sample interval should therefore lie somewhere between 50 and 200 meters. According to the tests in Section 6.1.2, there is not much difference between the optimized trajectories in the different cases. To make the algorithm as fast as possible and still not neglect any peaks, a sample interval of 150 meters is chosen. This yields the sample time 0.56 seconds when the aircraft has a velocity of 270 m/s.

7.3 Weights

The weight can be decided in many different ways. By comparing the methods tried out in Section 6.1.3, it seems like the best way is to use an even weight $Q$ along the whole horizon. It is more interesting to vary the weight $R$. If the flight is too rough, $R$ can be multi-
plied by a factor 30 for example so that the control sequence gets more even. The optimized trajectory lies further up from the reference trajectory in this case but the acceleration and jerk vary less so the flight gets more comfortable for the pilot.
8 Computational Aspects

Different parameters in the algorithm are varied in Chapter 6 to get a good controller. These parameters all have an influence on how fast the algorithm will be. It should in beforehand be pointed out that the QP algorithm used for the MPC optimization is rather slow and a much faster algorithm for QP can be made.

The aircraft computer is a RISC computer (PowerPC 740, see [14]) which has a CPU frequency of 266 MHz. One double precision multiplication, addition or subtraction takes up one clock interval. In a case when the horizon is chosen to 4 kilometers and the sample time is 0.56 seconds, there are $1.1 \times 10^7$ double precision multiplications, additions and subtractions each sample. The optimal control signal has to be delivered to the control system of the aircraft within 0.56 seconds, so that is the maximum time that can be used for the whole algorithm. Of course there are other operations done during the simulation but these are not paid attention to in this chapter. The calculations below are only done to give a harsh estimation of the simulation time. When using the CPU frequency 266 MHz, the time for the simulation is for every sample instant

$$T_{alg} = \frac{1}{266 \times 10^6} \cdot 1.1 \times 10^7 = 0.04 \text{ seconds}$$

The algorithm does not have to find the optimal solution that fast at all so a lower frequency can be used. The lowest CPU frequency for which the simulation will be done in time is in this case 22 MHz. When the horizon is extended to 8 kilometers, the CPU frequency for the calculations must be at least 160 MHz if the optimization is to be done within 0.56 seconds.

It is not always necessary to use double precision operations so the calculation time can get even lower by using single precision operations for some of the calculations. In cases when the horizon has to be very long, it is better to use single precision since that is less time consuming.

The model used for the MPC optimizations for the non-linear model of the aircraft is 8 dimensional. This model can be simplified into a 5 dimensional model and still be accurate and the MPC algorithm would in that way get even faster.
7.3 Computational Aspects, Weights
9 Conclusions

The object of terrain following is to fly as low as possible to avoid threats and to stay out of sight for others (aircrafts and sensors). In a case like this, there are often many other factors that the pilot has to consider. It is infeasible for the pilot to follow the optimized trajectory precisely if that is his only task. In terrain following mode, the pilots attention has to be focused on more tasks and this might make it hard for him to follow the trajectory accurately.

There are cases when the pilot can find a better trajectory himself but in these cases he usually knows the terrain he is flying over and does not need the terrain following mode. The algorithm can instead be made to present the trajectory to the pilot so that he can follow it himself. He might then during the flight find an adjustment of the trajectory and decide not to follow the optimized trajectory exactly. Since the database cannot include all peaks and masts, the trajectory cannot be trusted completely so this might be a better solution than using the autopilot even though the mean distance to the ground is not minimal.

The question is how important it is to have the aircraft fly as close to the reference trajectory as possible. It might be more desirable to have a more even flight and just make sure that the aircraft is as low as possible at high peaks. By varying the weights, or using different optimization criteria, and making more analysis of the terrain before the optimization, a trajectory with the desired quality can be found. In the algorithms in this thesis, the main objective has been to make the flight trajectory lie as near the reference trajectory as possible and in that case the weight $R$ should not be too high.

The algorithm is very time consuming if the sample time is chosen too small. A sample time around 0.6 seconds seems to be appropriate. This sample time yields a trajectory which is optimized every 150 meters. The length of the prediction horizon also decides the calculation time. A horizon of 4 kilometers yields a good flight trajectory and the optimization does not take very long. The algorithm gets much more time consuming when the prediction horizon is extended so then there have to be some simplifications of it. The best way of decreasing the simulation time when the horizon is long is to use blocking (4.6.4) of the control signals since it reduces the optimization time but still gives a very good trajectory.

When comparing the MPC method to methods used in other reports, e.g. [7], it can easily be found that MPC is a superior method. The best algorithm found in [7] gives a mean level over the reference trajectory at 26.5 meters for the Kisa terrain while the MPC algorithm in this thesis gives a mean level of 12.7 meters under the same constraints and terrain type. The Kisa terrain is one of the most difficult places in Sweden for terrain following.
7.3 Conclusions, Weights
10 Future Work

To get a safer flight for the pilot, the hills should, when calculating the reference trajectory, be broadened. The height over the ground of the reference trajectory is 50 meters but it should be 50 meters to the terrain in all directions. The aircraft should never get closer to the hills than that since there might be obstacles not included in the terrain database.

An analysis that has not been done is to compare the optimized trajectory with the actual flight trajectory when a pilot flies near the terrain. What does the pilot find important that is not considered in the algorithm? Would he accelerate over the hills the same way as the algorithm suggests? How demanding is it on the body during longer flights to vary the acceleration as much as suggested in this thesis. When an analysis like this is done, it is easier to decide what type of weights that are desirable.

The weights on the distance to the reference trajectory can be decided even better if the terrain is analyzed more thoroughly. An identification of the terrain can be made in beforehand to decide the weights better. The places where it is very important to fly near the ground can get a higher weight than the places where it does not make much difference if the aircraft lies right on the reference trajectory or 20 meters over it.

The Direct Method was not developed fully in this thesis but it would be very interesting to compare the methods more thoroughly since the Direct Method probably would give a better optimized trajectory. Even though this optimization algorithm is too time consuming at the moment so it cannot be implemented in the aircraft, it might be the more preferable one in perhaps ten years when the computers are faster. The derived algorithm can be extended to optimize over 3 dimensions, i.e. even in the horizontal plane. It would be interesting to see what method is the best for doing this type of optimization. It will for many methods be very time consuming to find the best 3 D trajectory. One method is to use MPC and in that way get a local optimal trajectory. An initial guess on the flight path can be found using some search algorithms for the whole distance from the starting point to the final point. This yields a globally optimal path. After this path is found, MPC can be used in 3 dimensions.
7.3 Future Work, Weights
References

[1] S Ahlqvist
"Development, implementation and validation of software for aircraft navigation simulation"
Master Thesis at Linköping University, LiTH-ISY-EX-3018, 1999

[2] P Bergljung
"JAS39 Utvecklingspotential. Studie av terrängföljningssystem baserad på noggrann navigering och databaser."

[3] E Camacho / C Bordons
"Model Predictive Control"
Springer-Verlag, 1998

"Digital Styrning - Kurskompendium"
Linköping University, Department of Automatic Control, 2000

"Some tools for the direct solution of optimal control problems"

"Digital Signalbehandling"
Studentlitteratur, 2000

"Olinjära och dynamiska optimeringsmetoder för terrängföljning"

[8] L Ljung / T Glad
"Reglerteknik"
Studentlitteratur, 1989

[9] L Ljung / T Glad
"Reglerteori"
Studentlitteratur, 1997

[10] J Löfberg
"MPC for nonlinear systems using trajectory linearizations"
Technical Report, Linköping University, Department of Electrical Engineering, Division of Automatic Control, LiTH-ISY-R-2173, 1999
7.3 Future Work, Weights

"Linear Model Predictive Control - Stability and Robustness"
Linköping University, Department of Electrical Engineering,
Division of Automatic Control, LiU-TEK-LIC-2001:3, 2001

[12] J Nocedal / S Wright
“Numerical Optimization”
Springer-Verlag, 2000

“Analys i en variabel”
Studentlitteratur, Lund 1990

http://e-www.motorola.com/brdata/PDFDB/docs/MPC750FACT.
A Appendix - Notations

A.1 Abbreviations

MPC Model Predictive Control
QP Quadratic Programming
RT90 Swedish coordinate frame
WGS-84 World Geodetic System

A.2 Symbols

\( l, L, h \) Position - latitude, longitude and altitude
\( \psi, \phi, \theta \) Attitude - yaw, roll and pitch
\( \alpha \) Attack angle
\( \gamma \) Flight path angle
\( v_x, v_y, v_z \) Velocity components in the B frame
\( v_n, v_e, v_d \) Velocity components in the L frame
\( n_z \) Load factor in the \( z \) direction
\( v_{tot} \) Velocity in the \( x \) direction in the S frame
\( C_B^L \) Transformation matrix between frame B and L
\( T_{m2deg} \) Transformation constant between frames B and L
\( A^T \) Transpose of the matrix \( A \)
\( I \) Identity matrix
\( T \) Sample time
\( N \) Prediction horizon
\( g \) Acceleration due to gravity (9.8066 m/s\(^2\))
\( R \) Mean radius of the earth (6378000 meters)
\( J \) Cost function
\( \delta_t \) Thrust
\( \delta_c \) Canard deflection
\( \delta_a \) Aileron deflection
\( \delta_r \) Rudder deflection
\( \delta_e \) Elevator deflection

A.3 Coordinate frames

\( B \) Body frame
\( L \) Local level frame
\( S \) Stability frame
A.4 Definitions

State

The properties of the system in a specific point are the system's state in this point.

Control signal

The control signal is sent to the system to change the properties at the next state.

Sample

The properties at a specific instant are gathered together (to a state).

Sample time

The sample time is the time period between two samples. During this time, a constant control signal is commanded.

Horizon

In Chapter 2, the horizon is the earth’s horizon. In the rest of the thesis, the horizon is the terrain interval over which each optimization is done. If the horizon is 3 kilometers, a control sequence is, after the optimization, found for this whole interval. The system is sampled into \( N \) different points along the horizon (\( N \) is the prediction horizon) and a control signal is calculated for each one of these points.

Weight

In e.g. optimization algorithm 3.4, there are two weight matrices, \( Q \) and \( R \). If we want to minimize \( y(t) \) extra much (compared to \( u(t) \)), we rise \( Q \) and if it is more important to minimize \( u(t) \) extra, we rise \( R \) instead.

Jacobian

The first order differential of a matrix.

Hessian

The second order differential of a matrix.

Cost function

The function we want to minimize (e.g. \( J \)).
På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnt som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

In English

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© [Cecilia Flood]