Reference Governor for Flight Envelope Protection in an Autonomous Helicopter using Model Predictive Control

Examensarbete utfört i Reglerteknik vid Tekniska högskolan vid Linköpings universitet av

Victor Carlsson & Oskar Sunesson

LiTH-ISY-EX–14/4780–SE
Linköping 2014
Reference Governor for Flight Envelope Protection in an Autonomous Helicopter using Model Predictive Control

Examensarbete utfört i Reglertechnik vid Tekniska högskolan vid Linköpings universitet av

Victor Carlsson & Oskar Sunesson

LiTH-ISY-EX–14/4780–SE

Handledare: Daniel Simon
           ISY, Linköpings universitet
           Ola Härkegård
           SAAB AB
           Patrick Borgqvist
           SAAB AB

Examinator: Daniel Axehill
            ISY, Linköpings universitet

Linköping, 12 juni 2014
### Title
Referensövervakning för flygenvelopsskydd i en autonom helikopter via modellbaserad prediktionseglering
Reference Governor for Flight Envelope Protection in an Autonomous Helicopter using Model Predictive Control

### Author
Victor Carlsson & Oskar Sunesson

### Abstract
In this master’s thesis we study how Model Predictive Control (MPC) can be fitted into an existing control system to handle state constraints. We suggest the use of reference governing based on the predictive control methodology. The platform for the survey is SAAAB’s unmanned helicopter SKELDAR. We develop and investigate different Reference Governor (RG) formulations that can be used together with the already existing stabilizing control system. These different setups show various features regarding model predictive control. One setup is complemented with a pre-filter to prevent aggressive actuator control in response to set-point changes, while the other is developed to handle this in the MPC framework. We also show that one of these RGs can be extended to guarantee stability and convergence.

Implementation and real time requirements are also considered in this thesis. For this two different QP-solvers have been used for online solving of the optimization problem that arises from the MPC formulations. For evaluation and analysis the solutions are implemented in an advanced simulation environment developed at SAAAB and in a hardware-in-the-loop avionics test rig for the SKELDAR system.

### Keywords
MPC, UAV, RG, Model Predictive Control, Reference Governor, Skeldar, Flight Envelope Protection
Abstract

In this master’s thesis we study how Model Predictive Control (MPC) can be fitted into an existing control system to handle state constraints. We suggest the use of reference governing based on the predictive control methodology. The platform for the survey is SAAB’s unmanned helicopter SKELDAR. We develop and investigate different Reference Governor (RG) formulations that can be used together with the already existing stabilizing control system. These different setups show various features regarding model predictive control. One setup is complemented with a pre-filter to prevent aggressive actuator control in response to set-point changes, while the other is developed to handle this in the MPC framework. We also show that one of these RGs can be extended to guarantee stability and convergence.

Implementation and real time requirements are also considered in this thesis. For this two different QP-solvers have been used for online solving of the optimization problem that arises from the MPC formulations. For evaluation and analysis the solutions are implemented in an advanced simulation environment developed at SAAB and in a hardware-in-the-loop avionics test rig for the SKELDAR system.
Acknowledgments

Both of us feel that we could not have found a better suited master thesis. To be able to work with something that we really have a genuine interest in has been a privilege. For this we would like to thank SAAB and Lic. Daniel Simon and Dr. Ola Härkegård for believing in us and giving us this opportunity.

We want to thank our dual mode academic/industry supervisor Lic. Daniel Simon for his expertise in Model Predictive Control and for guiding us in the right direction during this project. Especially for giving us feedback on the thesis report and taking time to discuss stability with us, more than once...

For everyday support and for reminding us of the practical aspects, important for us in the becoming of engineers, we would like to thank our industry supervisors Dr. Ola Härkegård and Dr. Patrick Borgqvist.

We also want to thank the SKELDAR family for making us feel welcome, we always felt motivated and delighted to go to work. A special thanks goes to Erik Backlund for helping us with ARES simulations and Mattias Waldo for support in the avionics test rig.

For our five years at LiTH we want to thank our fellow students and friends that have shared this time with us.

Victor Carlsson
Finally, I would like to give a special thanks to my girlfriend for all the support and love, that persists even through out the toughest parts of my education.

Oskar Sunesson
I would like to extend a special thanks to Emma because she cheers me up when I need it the most.

Linköping, June 2014
Victor Carlsson och Oskar Sunesson
## Contents

<table>
<thead>
<tr>
<th>Notation</th>
<th>ix</th>
</tr>
</thead>
</table>

| 1 Introduction | 1 |
| 1.1 Background | 1 |
| 1.2 Problem Description | 2 |
| 1.3 Method | 2 |
| 1.4 Approach | 3 |
| 1.5 Objective | 3 |
| 1.6 Limitations | 3 |
| 1.7 Thesis Outline | 4 |

| 2 Helicopter Dynamics | 5 |
| 2.1 Rigid-Body Dynamics | 6 |
| 2.1.1 Linearization of the Equations of Motion | 7 |
| 2.2 Main Rotor Dynamics | 8 |
| 2.2.1 Blade Motion | 8 |
| 2.2.2 Swashplate Mechanism | 8 |
| 2.2.3 Tip-path-plane | 9 |
| 2.3 Coupling of the Rigid-Body and Main Rotor Dynamics | 10 |
| 2.4 Tail Rotor Dynamics | 11 |
| 2.5 Heave Dynamics | 11 |
| 2.6 Complete Linear State Space Model | 11 |

| 3 Model Predictive Control | 13 |
| 3.1 Background | 13 |
| 3.2 Introducing the Model Predictive Controller | 14 |
| 3.2.1 Problem Formulation | 14 |
| 3.2.2 Stability Properties | 15 |
| 3.2.3 Further Notations | 15 |
| 3.3 Reference Tracking | 16 |
| 3.4 Relaxed Constraints | 17 |
| 3.5 Integral Action | 18 |
## Contents

4 Reference Governors 21
  4.1 Background and Introduction 21
  4.2 Earlier Work 22
    4.2.1 RG via Low-pass Filtering, [Gilbert et al., 1995] 22
    4.2.2 RG via Predictive Control, [Bemporad et al., 1997] 23

5 Closed Loop System for Heave Dynamics 25
  5.1 Vertical Model 26
  5.2 Primary Controller 27
  5.3 Inner Closed Loop System 27
    5.3.1 Closing the Loop 28
    5.3.2 Model Reduction 28

6 Reference Governor for Heave Dynamics 33
  6.1 Defining the Problem 33
    6.1.1 Simulation Environment 34
    6.1.2 Simulating Disturbances 35
  6.2 RG with Pre-filtered Nominal Reference 36
    6.2.1 Parameters and Tuning 41
  6.3 RG with Slew Rate Constraints 44
  6.4 RG with Resampled Prediction Model 51
  6.5 Stability 52
    6.5.1 Simulations with Terminal Set 55

7 Simulations 59
  7.1 ARES 59
    7.1.1 RG with Pre-filtered Nominal Reference 60
    7.1.2 RG with Slew Rate Constraints 65
    7.1.3 RG with Resampled Prediction Model 65
  7.2 HIL 66
  7.3 External Solvers 67
    7.3.1 FORCES 67
    7.3.2 CVXGEN 68
  7.4 Comparing FORCES and CVXGEN 69
    7.4.1 Usability 69
    7.4.2 Performance 70
    7.4.3 Implementation 71

8 Conclusions and Future Work 73

A Mathematical Definitions 77

Bibliography 79
# Notation

## Symbols, Operators and Functions

<table>
<thead>
<tr>
<th>Notation</th>
<th>Denotation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Helicopter</strong></td>
<td></td>
</tr>
<tr>
<td>$u$</td>
<td>Longitudinal velocity in body-fixed reference frame</td>
</tr>
<tr>
<td>$v$</td>
<td>Lateral velocity in body-fixed reference frame</td>
</tr>
<tr>
<td>$w$</td>
<td>Vertical velocity in body-fixed reference frame</td>
</tr>
<tr>
<td>$p$</td>
<td>Roll rate in body-fixed reference frame</td>
</tr>
<tr>
<td>$q$</td>
<td>Pitch rate in body-fixed reference frame</td>
</tr>
<tr>
<td>$r$</td>
<td>Yaw rate in body-fixed reference frame</td>
</tr>
<tr>
<td>$\phi$</td>
<td>Euler angle for roll</td>
</tr>
<tr>
<td>$\theta$</td>
<td>Euler angle for pitch</td>
</tr>
<tr>
<td>$\psi$</td>
<td>Euler angle for yaw</td>
</tr>
<tr>
<td>$\delta_{\text{lat}}$</td>
<td>Lateral cyclic control input</td>
</tr>
<tr>
<td>$\delta_{\text{lon}}$</td>
<td>Longitudinal cyclic control input</td>
</tr>
<tr>
<td>$\delta_{\text{col}}$</td>
<td>Collective pitch control input</td>
</tr>
<tr>
<td>$\delta_{\text{ped}}$</td>
<td>Pedal control input</td>
</tr>
<tr>
<td><strong>Reference Governor and MPC</strong></td>
<td></td>
</tr>
<tr>
<td>$J_k$</td>
<td>Objective function value at time $k$</td>
</tr>
<tr>
<td>$Q$</td>
<td>State penalty matrix</td>
</tr>
<tr>
<td>$R$</td>
<td>Input penalty matrix</td>
</tr>
<tr>
<td>$P_N$</td>
<td>Terminal penalty matrix</td>
</tr>
<tr>
<td>$\mathcal{X}$</td>
<td>State constraint set</td>
</tr>
<tr>
<td>$T$</td>
<td>Terminal state constraint set</td>
</tr>
<tr>
<td>$N$</td>
<td>Prediction horizon</td>
</tr>
<tr>
<td>$r$</td>
<td>Manipulated reference</td>
</tr>
<tr>
<td>$r_{\text{nom}}$</td>
<td>Nominal reference</td>
</tr>
<tr>
<td>$r_{\text{sp}}$</td>
<td>Set-point reference</td>
</tr>
<tr>
<td>$h$</td>
<td>Height above ground state</td>
</tr>
<tr>
<td>$v$</td>
<td>Vertical velocity state</td>
</tr>
<tr>
<td>$\Delta r(i)$</td>
<td>Reference rate at index $i$, $r(i) - r(i - 1)$</td>
</tr>
<tr>
<td>$\Delta^2 r(i)$</td>
<td>Reference acceleration at index $i$, $\Delta r(i) - \Delta r(i - 1)$</td>
</tr>
<tr>
<td>$T_s$</td>
<td>Sample time</td>
</tr>
</tbody>
</table>
## Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Notation</th>
<th>Denotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARES</td>
<td>Aircraft Rigid-body Engineering Simulation</td>
</tr>
<tr>
<td>COG</td>
<td>Center of gravity</td>
</tr>
<tr>
<td>DOF</td>
<td>Degrees of freedom</td>
</tr>
<tr>
<td>HIL</td>
<td>Hardware-in-the-loop</td>
</tr>
<tr>
<td>KKT</td>
<td>Karush-Kuhn-Tucker</td>
</tr>
<tr>
<td>LQ</td>
<td>Linear Quadratic (Regulator)</td>
</tr>
<tr>
<td>MPC</td>
<td>Model Predictive Control</td>
</tr>
<tr>
<td>PID</td>
<td>Proportional-Integral-Derivative (Controller)</td>
</tr>
<tr>
<td>QP</td>
<td>Quadratic Program</td>
</tr>
<tr>
<td>RG</td>
<td>Reference Governor</td>
</tr>
<tr>
<td>TPP</td>
<td>Tip-path-plane</td>
</tr>
<tr>
<td>UAS</td>
<td>Unmanned aerial system</td>
</tr>
<tr>
<td>UAV</td>
<td>Unmanned aerial vehicle</td>
</tr>
</tbody>
</table>
1

Introduction

1.1 Background

The development of Unmanned Aerial Vehicles (UAV) is rapidly increasing as UAVs prove to be useful in new applications. UAVs has been used mostly in the military but has begun to shift towards commercial use. Today we can find UAVs in rescue operations, aerial surveillance and even in motion picture filmmaking. In recent years the focus has changed from remotely operated UAVs to fully autonomous UAV systems. The most common UAVs today are fixed-wing aircraft as they are easier to operate as opposed to rotorcraft such as helicopters or multicopters. [Mettler, 2003]

Despite the fact that rotorcraft are more complex their abilities are highly attractive in a UAV system as they allow for missions impossible for a fixed-wing aircraft. The desirable features of a rotorcraft, such as the ability to take-off and land vertically and to hover in place, enable more flexible maneuvering and allow the aircraft to operate in close quarters.

With many years of experience in the aircraft industry a natural next step for SAAB AB was to apply their knowledge and expertise in the development of an autonomous UAV system. In 2006 SAAB AB began to develop an Unmanned Aerial System (UAS) called SKELDAR. SKELDAR is a medium-range fully autonomous unmanned helicopter able to carry a range of different payloads for information collection such as cameras and other sensory equipment. SKELDAR is commanded by high-level commands like “Point and Fly” or “Point and Look”, and is designed for different land, maritime and civil applications.
1.2 Problem Description

From an automatic control perspective the helicopter is a dynamic system with a lot of interesting control engineering challenges. Unlike the fixed-wing aircraft the helicopter does not have an inherent ability to stay in flight and instead must rely on a pilot or a control system to maintain force and moment equilibrium by commanding corrective control signals.

A control system of an aircraft must be able to handle different flight situations and must be robust towards disturbances such as wind. One control engineering challenge common to arise is the handling of constraints in the controlled system. Typical constraints of a controlled system can be limitations of the input signals, often hard physical constraints of the actuators, e.g. the maximum or minimum outlet of a valve. Other forms of limitations can be primary or secondary output constraints. These constraints usually originate from the desire to keep the system within some operating range, often specified by safety requirements or for being the most profitable operating region [Maciejowski, 2000].

In this thesis we focus on constraints of system states. Typical states of a helicopter that are desirable to keep within certain ranges are attitude angles and translational velocities. This operating region of an aircraft is called the flight envelope. Specifically we will be focusing on the vertical motion of the helicopter by considering the vertical dynamics as a subsystem to be controlled. Here the most critical state to be kept within a safe range is the climb or descent velocity during reference tracking of height.

1.3 Method

In this thesis we study how Model Predictive Control (MPC) can be used to satisfy above mentioned constraints. In MPC, constraints are handled in a structured and well defined way. MPC is based on solving an optimization problem and the constraints are taken into account by formulating them as constraints to the optimization problem.

Model predictive control can be implemented in a variety of structurally different ways where the standard form of MPC can be found in for example Rawlings and Mayne [2009] and Maciejowski [2000]. The standard MPC uses the desired reference as input and outputs suitable control signals to the system to be controlled. MPC can also be used as an outer controller to some inner control loop containing a primary, stabilizing controller and the system to be controlled. This is called a Reference Governor (RG) or Command Governor (CG), examples can be found in Gilbert et al. [1995] and Bemporad et al. [1997]. Here the input to the RG will be the desired reference or set-point to be tracked and the output will be a new reference to the primary controller, where the output reference will be chosen in such a way that no constraint violations will occur in the inner loop.

The current control system in SKELDAR is based on a bottom-up methodology
where different subsystems are implemented to solve some delimited control problems. To fit this framework we suggest the implementation of a reference governor as an add-on to the existing stabilizing control loops.

1.4 Approach

To gain necessary theoretical background, an extensive literature study of model predictive control and reference governors was made. A study of helicopter dynamics was also natural to understand the target system.

The development and analysis of our solutions was carried out in the MATLAB/SIMULINK environment and the MPC problems were formulated using the toolbox YALMIP by Löfberg [2004]. For calculation and visualization of polytopes we have used Multi-Parametric Toolbox 3 (MPT3) by Herceg et al. [2013].

For advanced simulations and testing we used a simulation environment developed at SAAB AB called ARES. This is a simulator capable of simulating a large amount of different modules interacting together to emulate the behavior of a multi-system such as an aircraft. ARES (Aircraft Rigid-body Engineering Simulation) simulates nonlinear dynamics and include time delays in servos etc., otherwise not included in the linear models used e.g. in control design.

For a complete analysis of the use of model predictive control as a design method in the SKELDAR framework the MPC regulators were implemented on an on-board computer of the same type used in SKELDAR. This was tested in a hardware-in-the-loop (HIL) simulation system to evaluate the real time requirements of the control system.

1.5 Objective

The goal of this thesis is not only to just solve a control problem, but rather to understand MPC as a design method. How to integrate the MPC framework into an already existing control system to satisfy state constraints. The MPC can be considered as an “add-on” to the existing control system and the add-on will in our case act as a flight envelope protection. The practical implementation of MPC on the SKELDAR platform is also considered as one of the main objectives.

1.6 Limitations

The main subject of this thesis lies in the field of control theory and the models used to simulate and implement MPC have been considered as known inputs. Therefore we will not conduct any modeling or identification of the helicopter. To gain an overview we present the dynamics of an helicopter and the models used in the development process.

The system to be controlled is a subsystem of the complete helicopter dynamics,
namely the vertical motion. Control of the complete helicopter system has not been examined in this thesis. Instead the process of formulating MPC to fit the existing framework and implement on similar on-board systems has been the focus.

We focus the thesis towards a subclass of controllers called reference governors where the reference to an already controlled system is manipulated in order to satisfy constraints. Both the stabilizing primary controller and the system model of the control loop to be governed by a RG, are considered given.

1.7 Thesis Outline

Chapter 2 consists of a theoretical background covering the dynamics of a helicopter and how it is modeled. This is followed by model predictive control theory in Chapter 3. In Chapter 4 we introduce the concept of a reference governor and present two articles regarding the use of reference manipulating controllers to account for systems with state and control constraints. The control loop considered in the thesis is presented and modeled through state feedback in Chapter 5. We also propose some model reduction for minimal state space representation of the model, later used as prediction model in the MPC. The control problem of the system is covered in Chapter 6 where we define the goals of the RGs. We also present the development process of the proposed RGs. In Chapter 7 we present results from simulations in ARES and the avionics test rig. The solvers used to implement the controllers in the simulation systems are described and evaluated in Chapter 7.3. Conclusions and future work can be found in Chapter 8.
Model predictive control is, as its name suggests, a model based control algorithm. It is therefore necessary to have an accurate model in order to develop a well performing MPC-regulator. Since the helicopter is a highly complex, high-order, nonlinear dynamic system it is also necessary to make approximations. The describing dynamics must be simplified, yet still be able to capture the helicopter’s primary behavior, to be able to synthesize an applicable model based controller.

"Make things as simple as possible, but not simpler."

— Albert Einstein

In this thesis the parametrization and identification are omitted and the model for the helicopter is assumed given. In this chapter we will explain the dynamics of the helicopter as well as present the basics behind the parametrization of the dynamic model.

A fixed-wing aircraft is almost exclusively modeled as a rigid-body using Newton-Euler equations of motion [Mettler, 2003]. This is also a good starting point for developing a helicopter model. The rigid-body dynamics of the helicopter are then augmented with the dynamics of the main and tail rotors to form the complete model. This is the approach proposed in Mettler [2003] and throughout this chapter we will follow the derivation of the linear parameterized model presented in the book, for details we refer the reader to the literature mentioned.

Modeling and system identification of SKELDAR can be found in an earlier master thesis by Svenson [2014].
Figure 2.1: The body-fixed reference frame with origin at the center of gravity. The body axes are represented by $x, y$ and $z$. The vehicle velocity components $u, v, w$, Euler angles $\phi, \theta, \psi$ and angular rates $p, q, r$ are also shown in the figure. The rotor blade position and angular rate denoted $\Psi$ and $\Omega$ respectively and are measured from the tail.

2.1 Rigid-Body Dynamics

The helicopter is a highly versatile aircraft because of its ability to both rotate and translate in six degrees of freedom (DOF). Figure 2.1 shows the body-fixed reference frame with origin at the center of gravity (COG) along with the state variables of the helicopter. As mentioned earlier the rigid-body dynamics for the fuselage are derived by using the Newton-Euler equations. Expressed in the body-fixed reference frame with constant mass $m$ and moment of inertia $I$ they are:

$$m\ddot{\mathbf{v}} + m(\omega \times \mathbf{v}) = \mathbf{F}$$

$$I\ddot{\omega} + (\omega \times I\omega) = \mathbf{M}$$

where $\mathbf{F} = [X\ Y\ Z]^T$ denotes the external forces acting on the aircraft COG and $\mathbf{M} = [L\ M\ N]^T$ the external moments. $\mathbf{v} = [u\ v\ w]^T$ and $\omega = [p\ q\ r]^T$ denotes the velocities and angular rates of the fuselage. This yields the following three nonlinear differential equations for translational motion:

$$\dot{u} = (-wq + vr) + X/m$$

$$\dot{v} = (-ur + wp) + Y/m$$

$$\dot{w} = (-vp + uq) + Z/m$$

and in the same manner (2.2) gives three nonlinear differential equations for rotational motion:

$$\dot{p} = -qr(I_{yy} - I_{zz})/I_{xx} + L/I_{xx}$$
\[ \dot{q} = -pr(I_{zz} - I_{xx})/I_{yy} + M/I_{yy} \quad (2.7) \]
\[ \dot{r} = -pq(I_{xx} - I_{yy})/I_{zz} + N/I_{zz} \quad (2.8) \]

where, in Mettler [2003], the assumption of small cross products of inertia has been made, i.e. the principal axes coincide with the axes of the body-fixed reference frame. This is not necessary and typically not assumed, but will simplify the notation. See Stevens and Lewis [2003] for a derivation where the principal axes and the body-fixed reference frame is not assumed to coincide.

Now we want to express these six first-order differential equations as functions of control inputs and vehicle states in the following form:

\[ \dot{x} = f(x, u) \quad (2.9) \]
\[ x = [u, v, w, \phi, \theta, \psi, p, q, r]^T \quad (2.10) \]
\[ u = [u_{lat}, u_{lon}, u_{col}, u_{ped}]^T. \quad (2.11) \]

Here \( x \) is the state vector of the aircraft and \( u \) the control input vector. \( u_{lat} \) and \( u_{lon} \) are the lateral and longitudinal cyclic rotor controls, \( u_{col} \) the collective pitch and \( u_{ped} \) the tail rotor collective control.

### 2.1.1 Linearization of the Equations of Motion

Although it is possible to develop a nonlinear MPC we will in this thesis focus on linear MPC. In order to derive the linear model, the nonlinear equations of motion (2.3)–(2.8) can be linearized about an equilibrium state, denoted \( x_0 \). In this case the equilibrium state will be chosen as the hover flight condition, defined by \( v_0 = \omega_0 = [0 0 0]^T \). By fixing the known states \( v_0, \omega_0 \) and then solve \( f(x, u) = 0 \) we get our complete equilibrium point \( x_0 \) and \( u_0 \). The linearized equations of motion can then be calculated by:

\[ \dot{\delta x} = A\delta x + B\delta u. \quad (2.12) \]

This will result in the desired state space form, where \( A \) represents the state matrix and \( B \) the control matrix, as:

\[ \dot{x} = A x + B u \quad (2.13) \]

and since we linearize about \( x_0 \) and \( u_0 \) the state space form represents perturbations close to our equilibrium point by:

\[ x = x_0 + \delta x \quad u = u_0 + \delta u. \quad (2.14) \]

The linearized form of translational motion (at a general equilibrium point) in (2.3)–(2.5) can then be written as

\[ \delta \dot{u} = (-w_0\delta q + \delta wq_0 + v_0\delta r + \delta vr_0) + \Delta X/m \quad (2.15) \]
\[ \delta \dot{v} = (-u_0\delta r + \delta ur_0 + w_0\delta p + \delta wp_0) + \Delta Y/m \quad (2.16) \]
\[ \delta \dot{w} = (-v_0\delta p + \delta vp_0 + u_0\delta q + \delta uq_0) + \Delta Z/m \quad (2.17) \]
and the corresponding linearized form of rotational motion in (2.6)–(2.8) as

\[\dot{\delta p} = (-q_0\delta r - \delta qr_0)(I_{yy} - I_{zz})/I_{xx} + \Delta L/I_{xx} \quad (2.18)\]

\[\dot{\delta q} = (-p_0\delta r - \delta pr_0)(I_{zz} - I_{xx})/I_{yy} + \Delta M/I_{yy} \quad (2.19)\]

\[\dot{\delta r} = (-p_0\delta q - \delta pq_0)(I_{xx} - I_{yy})/I_{zz} + \Delta N/I_{zz} \quad (2.20)\]

Left to do for the rigid-body dynamics is to expand the external forces and moments and express them as linear functions of the states and control inputs. This can be done by performing a Taylor series expansion, and only keeping the first order terms. For the longitudinal force component this would yield:

\[\Delta X = X_u\delta u + X_v\delta v + \cdots + X_{\delta \text{lat}}\delta \text{lat} + X_{\delta \text{lon}}\delta \text{lon} + \cdots \quad (2.21)\]

For compact expressions we use the notation \(X_u = \frac{\partial X}{\partial u}\) for the partial derivatives. In the sequel we will also drop the deltas (\(\delta\)) for all the state variables, which gives the notation \(x := \delta x\). The partial derivatives with respect to the vehicle states and control inputs are called stability derivatives and control derivatives respectively. These derivatives will be parameters to identify during the identification process, which is omitted in this thesis.

The external forces and moments acting on the rigid-body are generated by the main and tail rotor. The expansion of the force component \(\Delta X\) suggests that the control signals act instantaneously and proportionally on the forces and moments. This is not the case since the rotors themselves are dynamic systems. Therefore we need to augment the rigid-body model with the dynamics of the rotors.

### 2.2 Main Rotor Dynamics

In this section we will describe the basics of the main rotor dynamics. First the blade motion is presented to understand and introduce the blade’s three degrees of freedom. This is followed by the swashplate mechanism which describe the basis of the pitch blade control. Finally to describe the thrust generated by the main rotor we introduce the concept of the **tip-path-plane**.

#### 2.2.1 Blade Motion

The motion around the hub is described by the speed \(\Omega\) and the position \(\Psi\) seen in Figure 2.1. The blade can also move about its anchor point in three degrees of freedom described by the flapping angle \(\beta\), pitch angle \(\Theta\) and lead-lag angle \(\xi\). The blade motion is shown in Figure 2.2.

#### 2.2.2 Swashplate Mechanism

The thrust generated by the main rotor can be controlled by changing the rotor speed \(\Omega\) or the pitch angle \(\Theta\). In helicopters the rotor speed is kept constant and therefore changes of the pitch angle are used. The swashplate mechanism can be used to vary the magnitude of the pitch angle but can also vary the angle as a function of its position \(\Psi\) around the hub. This is called cyclic pitch control and
2.2 Main Rotor Dynamics

Figure 2.2: Blade motion. Flapping angle $\beta$, pitch angle $\Theta$ and lead-lag angle $\xi$ describing the blade’s three degrees of freedom.

by varying the pitch angle as a function of the blades position $\Psi$ the direction of the thrust vector can be controlled.

The pitch angle as a function of the position around the hub can be written as

$$\Theta(\Psi) = \Theta_0 - A_1 \cos \Psi - B_1 \sin \Psi$$  \hspace{2cm} (2.22)

where the average pitch angle $\Theta_0$ is controlled by the collective pitch input $\delta_{col}$. The coefficients $A_1$ and $B_1$ are controlled by the longitudinal and lateral pitch input respectively and describe the blade pitch when above the tail and on the right-hand side. $A_1$ and $B_1$ are controlled through:

$$A_1 = B_{lat} \delta_{lat}, \quad B_1 = A_{lon} \delta_{lon}.$$  \hspace{2cm} (2.23)

With $A_1$ and $B_1$ as described above and the control inputs given in percentage of their maximum range the units of $A_{lon}$ and $B_{lat}$ are $\text{rad}/\%$. This awkward use of letters is retrieved from Mettler [2003] and we choose to follow the same notation to easily accompany the literature.

2.2.3 Tip-path-plane

When applying cyclic pitch control as described in the previous section, the rotating blade undergoes periodic aerodynamic forces which in turn will generate a periodic flapping motion of the blade. The flapping motion of the main rotor blades give rise to a rotor cone, as seen in Figure 2.3, and the tip-path-plane (TPP) is defined by the top of the cone. The thrust vector $T$ will be perpendicular to the TPP and by tilting the rotor cone with respect to the hub plane one can control the direction of the thrust vector. This is the main way of maneuvering the helicopter.
The flapping angle is a $2\pi$-periodic function and can be approximated with the first harmonics of the Fourier series expansion:

$$
\beta(\Psi) \approx \beta_0(t) - \beta_c(t) \cos \Psi - \beta_s(t) \sin \Psi. 
$$

Here the constant term $\beta_0(t)$ describes the coning angle and the coefficients $\beta_c(t)$ and $\beta_s(t)$ the longitudinal and lateral tilting of the rotor cone. The motion of the TPP can, while skipping a few steps in the derivation (again, see Mettler [2003] for details), be described by first order equations expressed in $\dot{\beta}_c(t)$ and $\dot{\beta}_s(t)$:

$$
t_f \dot{\beta}_c(t) = -\beta_c(t) - \tau_f q + \frac{p}{\Omega} + A_{\beta_c} \beta_s(t) - A_{\text{lon}} \delta_{\text{lon}}
$$

$$
t_f \dot{\beta}_s(t) = -\beta_s(t) - \tau_f p - \frac{q}{\Omega} - B_{\beta_c} \beta_c(t) + B_{\text{lat}} \delta_{\text{lat}}
$$

where $\tau_f$ is the time constant of the rotor and $A_{\beta_s}$ and $B_{\beta_c}$ describe a cross-coupling effect between the longitudinal and lateral flapping.

### 2.3 Coupling of the Rigid-Body and Main Rotor Dynamics

The forces and moments produced by the main rotor can be expressed in terms of the rotor flapping described in the previous section. To couple the rigid-body and main rotor dynamics the control derivatives in (2.15)–(2.20) can be replaced with the flapping derivatives and the control inputs now act on the dynamics of the main rotor. For the control derivative of $\delta_{\text{lat}}$ in the longitudinal force component in (2.21) this would yield

$$
X_{\delta_{\text{lat}}} \delta_{\text{lat}} \rightarrow X_{\beta_s} \beta_s(t)
$$

where $X_{\beta_s}$ is the flapping derivative.
2.4 Tail Rotor Dynamics

The thrust generated by the tail rotor is the main source for maneuvering the helicopter’s yaw motion. It is controlled through the blade pitch control $\delta_{ped}$. Compared to the fuselage yaw dynamics the tail rotor dynamics are much quicker hence no tail rotor dynamics need to be modeled. The contribution of the tail rotor to the yaw dynamics $\dot{\gamma}$ is then simply expressed as $N_{ped}\delta_{ped}$. This is then coupled to the rigid-body and main rotor through (2.20).

2.5 Heave Dynamics

The heave dynamics (vertical motion) of the helicopter are of special interest since this is the subsystem considered in this thesis. The vertical motion is derived from the linearized equation of translational motion in (2.17) and through the stability derivative in (2.21) (though for the corresponding $\Delta Z$ term). In hover flight mode the trim conditions are $v_0 = \omega_0 = [0 \ 0 \ 0]^T$. This yields

$$\dot{w} = Z_w w + Z_{\beta_c} \beta_c + Z_{\beta_s} \beta_s + Z_{col} \delta_{col}. \quad (2.28)$$

The baseline of this expression is that the vertical motion is described by the thrust generated through changes in the collective pitch $\delta_{col}$ and a damping effect from the speed derivative $Z_w$. This damping accounts for rotor damping and fuselage drag.

2.6 Complete Linear State Space Model

In the previous sections of this chapter we motivate the dynamics behind the state space model derived in Mettler [2003] and the model used in this thesis. In these sections we also present some of the mechanics and physics of the helicopter, to get a feel of how it all comes together. We choose to leave the aerodynamic derivation for the sake of simplicity.

The complete state space model is collected in (2.29). Note that the model derived in the mentioned literature also include dynamical effects from having a stabilizer bar attached to the main rotor. This is not present in SKELDAR and is omitted from the model. It also includes augmentation of the yaw dynamics to fit the result of the frequency response estimation conducted in the book, which is also omitted. We will also skip a few steps where the identification of which stability derivative terms actually affect the expressions (2.15)–(2.20).

The complete state space model describes:

- longitudinal-lateral dynamics:
  - fuselage longitudinal and lateral motion ($u, v$)
  - fuselage roll and pitch motion ($p, q$)
  - fuselage absolute roll and pitch angles ($\phi, \theta$)
  - rotor longitudinal and lateral flapping ($\beta_c, \beta_s$)
- **heave dynamics:**
  - fuselage vertical motion \((w)\)

- **yaw dynamics:**
  - fuselage yaw motion \((r)\)

\[
\begin{bmatrix}
\dot{u} \\
\dot{v} \\
\dot{p} \\
\dot{q} \\
\dot{\phi} \\
\dot{\theta} \\
\dot{\beta}_c \\
\dot{\beta}_s \\
\dot{w} \\
\dot{r}
\end{bmatrix} =
\begin{bmatrix}
X_u & 0 & 0 & 0 & 0 & -g & X_{\beta_c} & 0 & 0 & 0 \\
0 & Y_v & 0 & 0 & 0 & 0 & 0 & Y_{\beta_s} & 0 & 0 \\
L_u & L_{\nu} & 0 & 0 & 0 & 0 & 0 & L_{\beta_c} & L_{w} & 0 \\
M_u & M_{\nu} & 0 & 0 & 0 & 0 & 0 & M_{\beta_c} & 0 & M_{w} \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & N_{\nu} & N_{\beta_c} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & N_{\beta_s} & N_{w} & N_{r}
\end{bmatrix}
\begin{bmatrix}
\dot{u} \\
\dot{v} \\
\dot{p} \\
\dot{q} \\
\dot{\phi} \\
\dot{\theta} \\
\dot{\beta}_c \\
\dot{\beta}_s \\
\dot{w} \\
\dot{r}
\end{bmatrix}
\begin{bmatrix}
u \\
v \\
p \\
q \\
\phi \\
\theta \\
\beta_c \\
\beta_s \\
w \\
r
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}
\begin{bmatrix}
\delta_{lat} \\
\delta_{lon} \\
\delta_{ped} \\
\delta_{col}
\end{bmatrix}
\]

\[(2.29)\]
In this chapter we will present some background to model predictive control and give a theoretical base for the control algorithm. First we will introduce MPC in its most fundamental formulation. Then we will further develop this formulation with extensions for reference tracking, integral action and relaxed constraints. The topic of stability is also covered, although quite briefly.

### 3.1 Background

Model predictive control is derived from the field of optimal control and is based on solving an optimization problem for each sampling instant. The optimization problem to be solved is a finite horizon, convex minimization problem (in the linear case) where the variables are potentially bounded by constraints. As a result of the optimization a sequence of control signals is given. Only the first control signal in this sequence is used and is passed on to the system that should be controlled. By doing this procedure in each sampling instant with new measurement data one has achieved a form of feedback control. The model is used in the optimization to predict the behavior of the system in order to calculate the optimal control signals.

In the past MPC has mainly been used in the process industry where the dynamics are slow and there is enough time to perform the optimization. But along with an increasing access to computational power, MPC has found more applications. Some of the benefits with MPC have partially been mentioned above. The variables in the optimization problem are typically the state and control signals and therefore one has a direct way to take, for example safety regulations and actuator saturations into account. Another benefit that has been mentioned is that
feedback control is achieved by using the current state of the system as initial
c condition in the optimization. This can be compared to solving the optimization
problem only once and then use the whole sequence of input signals which gives
open loop control.

3.2 Introducing the Model Predictive Controller

In this theoretical presentation of model predictive control we will show the case
where a linear state space model is used for prediction in the optimization prob-
lem. Even though the MPC controller is nonlinear due to the presence of con-
straints in the system, one refers to linear MPC when the system dynamics are
linear. Nonlinear MPC refers to systems with nonlinear dynamics, which is out-
side the scope of this thesis. We will also restrict the presentation to discrete time
systems and controllers.

3.2.1 Problem Formulation

The basic setup for an MPC controller with a quadratic cost function, linear con-
straints and with the objective to drive the system states and control signals to
zero is formed as:

\[
\begin{align*}
\text{minimize} & \quad x_N^T P_N x_N + \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i \\
\text{subject to} & \quad x_{i+1} = Ax_i + Bu_i \quad \forall i = 0, \ldots, N - 1 \\
& \quad x_{\min} \leq x_i \leq x_{\max} \quad \forall i = 0, \ldots, N - 1 \\
& \quad u_{\min} \leq u_i \leq u_{\max} \quad \forall i = 0, \ldots, N - 1 \\
& \quad x_N \in T
\end{align*}
\]

\(N\) is referred to as the prediction horizon and determines how many steps for-
dward the controller will predict and therefore also implicitly affect the number
of optimization variables. \(Q\) and \(R\) are the penalties for the states and control sig-
nals respectively. Typically you allow a different penalty matrix \(P_N\) for the final
state \(x_N\). These four parameters \((N, Q, R, P_N)\) along with \(T\) (see next paragraph)
are the tuning variables in the controller. \(Q\) and \(R\) are used to weight the impor-
tance of speed and small control signals. Large values of \(Q\) in comparison to \(R\)
will bring the states to the origin quickly but at the cost of large control inputs.

The equality constraints are the dynamics from the model of the system that will
be controlled. The inequality constraints are bounds on system states and/or on
input signals, typically safety regulations and actuator saturations. \(T\) is called
terminal set and is used to force the final state into a certain area, often used to
ensure stability of the controller. This is also something that the control designer
can choose and the terminal set will affect the behavior of the MPC. In the next
section we present a few details regarding the terminal set and how it can be
chosen to ensure stability.
3.2.2 Stability Properties

As in all control design it is important to ensure stability of the closed loop system. In model predictive control the terminal cost $P_N$ and terminal set $T$ are used to modify the problem formulation to guarantee stability. For a detailed review of stability conditions see Mayne et al. [2000].

The term $x_T^T P_N x_N$ can be considered as an approximation or upper bound of the truncated part of the (LQ) cost function

$$
\sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i + \sum_{i=N}^{\infty} x_i^T Q x_i + u_i^T R u_i \leq x_N^T P_N x_N.
$$

A widely used approach to ensure closed loop stability is to choose $P_N$ to be the solution to the Riccati difference equation.

When using a terminal set $T$ to ensure stability the most straightforward course of action is to use $T = 0$. This way the states are forced to reach the origin at the end of the prediction horizon and thus the controller is stable. The problem with this formulation is that the MPC controller might be faced with an infeasible problem if the system cannot be controlled to the origin within the horizon.

To enlarge the feasible area, an area from which the system can be controlled to reach the terminal set $T$ within the horizon, one can instead let $T$ contain a neighborhood of the origin. This neighborhood is chosen so that once it is reached, the system will not move outside this set and the constraints will not be violated, i.e.:

- The terminal set $T$ is a positively invariant set (see A.6).
- $T \subseteq \mathcal{X}$ where $\mathcal{X}$ is the set defined by the state constraints in (3.1c).
- In the terminal set $T$ the control signal $u \in \mathcal{U}$ where $\mathcal{U}$ is the control constraint set in (3.1d).

This can be achieved by letting a local stabilizing controller steer the system to the origin once the system reaches the terminal set. Given a local stabilizing controller it is possible to calculate a positively invariant set. For this approach it is enough to reach the terminal set and let the local controller take over, instead of forcing the system to the origin within the horizon. This setup is called dual mode MPC. The dual term refers to the primary MPC and the local stabilizing controller. Note that the local controller is never actually applied to the system due to the receding horizon property of the MPC. Again, for details regarding stability, see Mayne et al. [2000].

3.2.3 Further Notations

In the sequel we will, for compact expressions, denote the quadratic penalties in the cost function as weighted 2-norms, i.e. $z^T W z = \|z\|_W^2$. We will also omit
∀i = 0, . . . , N − 1 for the constraints. This is assumed if nothing else is stated.

3.3 Reference Tracking

In the MPC formulation of Section 3.2 the system will be driven towards a steady state that lies in the origin. If one wants to control the system to some steady state other than the origin the MPC formulation must be extended to include reference tracking.

Consider the linear system with states \( x_k \), input \( u_k \) and output \( y_k \)

\[
\begin{align*}
x_{k+1} &= Ax_k + Bu_k \quad (3.3a) \\
y_k &= Cx_k + Du_k. \quad (3.3b)
\end{align*}
\]

For some desired output \( y_k = y_r \), where \( y_r \) is the reference to be tracked the system will have a steady state \( (x_r, u_r) \). Since it is a steady state it must hold that:

\[
x_r = Ax_r + Bu_r. \quad (3.4)
\]

This gives for (3.3) written in matrix form:

\[
\begin{bmatrix} A - I & B \\
C & D \end{bmatrix} \begin{bmatrix} x_r \\
u_r \end{bmatrix} = \begin{bmatrix} 0 \\
y_r \end{bmatrix} \quad (3.5)
\]

Note that the state and input constraints must also be imposed on \( (x_r, u_r) \) to ensure feasible steady states. By solving (3.5) for \( (x_r, u_r) \) one calculates the steady states for a given reference \( y_r \). However it is not always trivial to solve (3.5). For (3.5) to have a solution for all set-points \( y_r \) it is sufficient to require at least as many inputs as outputs with set-points, which would make the rows of the matrix on the left hand side linearly independent [Rawlings and Mayne, 2009]. In Muske and Rawlings [1993] the authors dictates methods of solving (3.5) when the solution is non-unique, where one approach is to select steady states \( x_r \) which is the minimal norm of the input \( u_r \).

To achieve tracking of \( y_r \) in the MPC one can instead minimize the deviation of the current state to the calculated steady states \( (x_r, u_r) \). We can then reformulate the MPC problem (3.1) to include reference tracking

\[
\begin{align*}
\text{minimize} & \quad \|x_N - x_r\|^2_P + \sum_{i=0}^{N-1} \|x_i - x_r\|^2_Q + \|u_i - u_r\|^2_R \\
\text{subject to} & \quad x_{i+1} = Ax_i + Bu_i \quad (3.6a) \\
& \quad x_{\text{min}} \leq x_i \leq x_{\text{max}} \quad (3.6c) \\
& \quad u_{\text{min}} \leq u_i \leq u_{\text{max}} \quad (3.6d) \\
& \quad (x_N - x_r) \in T(x_r) \quad (3.6e)
\end{align*}
\]

where the costs in the cost function are rewritten as weighted 2-norm and \( (x_r, u_r) \) is given by (3.5). This is the standard procedure used in Rawlings and Mayne [2009] to obtain reference tracking. Note also that for the dual mode method the
terminal set $T(x_r)$ will depend on the steady state. In the literature $T$ is simply translated in the state space, but $T$ might become invalid when translated if it moves parts of the terminal set $T$ outside $X$. Simon [2014] provides an excellent example of this problem and also suggests how to deal with reference tracking in dual mode MPC.

### 3.4 Relaxed Constraints

One common problem to arise in MPC is that the optimization problem becomes infeasible. This can happen if suddenly a disturbance occur on the system when, e.g., some system state is being controlled close to its constraint. If the disturbance is large enough it might push the state beyond the constraint and the optimization problem becomes infeasible.

A systematic approach to handle infeasibility of such state constraints proposed in Maciejowski [2000] is to use relaxed constraints. By allowing the system states to cross their constraints, only if necessary, there is no possibility of the optimization problem becoming infeasible. This can be achieved by introducing another optimization variable $\epsilon$ and by adding this variable to the state constraints allowing the constraints to enlarge if necessary. This type of variable is called a slack variable. The optimization problem is then modified as

$$\begin{align*}
\text{minimize} & \quad \|x_N\|_P^2 + \sum_{i=0}^{N-1} \|x_i\|^2_Q + \|u_i\|^2_R + \|\epsilon_i\|^2_\rho \\
\text{subject to} & \quad x_{i+1} = Ax_i + Bu_i \\
& \quad x_{\text{min}} - \epsilon_i \leq x_i \leq x_{\text{max}} + \epsilon_i \\
& \quad u_{\text{min}} \leq u_i \leq u_{\text{max}} \\
& \quad x_N \in T \\
& \quad \epsilon_i \geq 0.
\end{align*}$$

(3.7a)

Note that here we penalize $\epsilon_i$ quadratically, which is the most straightforward approach, and by choosing a large penalty $\rho$ the controller will try to always set $\epsilon_i$ to zero. However, for quadratic penalty on $\epsilon_i$, the constraints will always, to some extent, be violated when active. This comes from the fact that there exist a small step $\epsilon d$ from the optimal solution $x^*$ of the hard constrained problem into the infeasible area. This step will reduce the cost function by order $O(\epsilon)$ but will penalize the cost of order $O(\epsilon^2)$, so for small $\epsilon$ the decrease in cost will be larger than the increase of penalty. If we instead use a linear penalty the increase in the cost function when violating the constraints is also $O(\epsilon)$. If then $\rho$ is large enough the decrease in cost would be smaller than the increase of penalty. This gives exact penalty as described in Maciejowski [2000]. By replacing the quadratic penalty function of $\epsilon_i$ in cost function (3.7a) with any linear norm, and choosing the weight $\rho$ accordingly, exact penalty is achieved.
3.5 Integral Action

Models used in model predictive control are always synthesized under some approximations. This will result in modeling errors that in turn will propagate to faulty predictions of the system states in the MPC controller. Considering MPC with reference tracking as in Section 3.3 it is easy to imagine that modeling error will result in incorrect steady state \((x_r, u_r)\) which would give rise to a tracking error. Another issue is the presence of disturbances in the controlled system which can cause similar prediction and tracking errors.

The suggested method in Rawlings and Mayne [2009] to include integral action in MPC is the introduction of disturbance observers. We augment the system state with a disturbance \(d_k\) driven by white noise \(w\)

\[
d_{k+1} = d_k + w. \tag{3.8}
\]

The augmented model is then given as

\[
\begin{bmatrix}
  x_{k+1} \\
  d_{k+1}
\end{bmatrix} =
\begin{bmatrix}
  A & B_d \\
  0 & I
\end{bmatrix}
\begin{bmatrix}
  x_k \\
  d_k
\end{bmatrix} +
\begin{bmatrix}
  B \\
  0
\end{bmatrix} u_k +
\begin{bmatrix}
  0 \\
  1
\end{bmatrix} w
\]

(3.9a)

\[
y_k =
\begin{bmatrix}
  C & C_d
\end{bmatrix}
\begin{bmatrix}
  x_k \\
  d_k
\end{bmatrix} + v. \tag{3.9b}
\]

The matrices \(B_d\) and \(C_d\) are chosen to describe how the disturbances effect the system. Usually one considers having both process noise \(w\) and measurement noise \(v\). From this augmented model we can formulate an observer to estimate the system states and disturbances

\[
\begin{bmatrix}
  \hat{x}_{k+1} \\
  \hat{d}_{k+1}
\end{bmatrix} =
\begin{bmatrix}
  A & B_d \\
  0 & I
\end{bmatrix}
\begin{bmatrix}
  \hat{x}_k \\
  \hat{d}_k
\end{bmatrix} +
\begin{bmatrix}
  B \\
  0
\end{bmatrix} u_k + K
\begin{bmatrix}
  y_k -
\end{bmatrix}
\begin{bmatrix}
  C & C_d
\end{bmatrix}
\begin{bmatrix}
  \hat{x}_k \\
  \hat{d}_k
\end{bmatrix}
\]

(3.10)

where the feedback gain matrix \(K\) is chosen with any design method that yields a stable \(\tilde{A} - K\tilde{C}\) matrix. If the system is assumed to be subjected to white noise it is optimal in a minimum variance sense to choose \(K\) as the Kalman filter gain [Gustafsson et al., 2010].

The steady state \((x_r, u_r)\) for a given reference \(y_r\) is now calculated as

\[
\begin{bmatrix}
  A - I & B \\
  C & D
\end{bmatrix}
\begin{bmatrix}
  x_r \\
  u_r
\end{bmatrix} =
\begin{bmatrix}
  -B_d \hat{d}_k \\
  y_r - C_d \hat{d}_k
\end{bmatrix}
\]

(3.11)
With the estimated states $\hat{x}_k$ and the disturbance $\hat{d}_k$ from (3.10) and the steady state from (3.11), we can now reformulate the MPC problem to include integral action as

$$\begin{align*}
\text{minimize} & \quad \|x_N - x_r\|_P^2 + \sum_{i=0}^{N-1} \|x_i - x_r\|_Q^2 + \|u_i - u_r\|_R^2 \\
\text{subject to} & \quad x_0 = \hat{x}_k \\
& \quad x_{i+1} = Ax_i + Bu_i + B\hat{d}_k \\
& \quad x_{\text{min}} \leq x_i \leq x_{\text{max}} \\
& \quad u_{\text{min}} \leq u_i \leq u_{\text{max}} \\
& \quad (x_N - x_r) \in T(x_r)
\end{align*}$$

(3.12a)
In this chapter we will present the idea behind reference governors. We will also describe how model predictive control can be fitted into such a framework and finally show some examples of what has been done before.

### 4.1 Background and Introduction

Systems to be controlled are almost exclusively bounded by some physical limits. These are constraints that need to be taken into account when designing control systems. Common control methods such as PID and LQ control do not have a natural way of dealing with state or input constraints. One way of getting around this problem is to use a Reference Governor (RG). A reference governor is used to modify the nominal reference value, $r_{\text{nom}}$, and pass on the modified reference, $r$, to an inner closed loop system. The standard configuration can be seen in Figure 4.1.

The reference governor shall take the constraints into account and feed the inner system with a reference signal, which the primary controller follows without violating the constraints. A key advantage of reference governors is that they can be applied to already existing systems designed for good performance in nominal conditions. If the nominal reference is such that no constraint will be violated one wishes the RG to be passive, i.e. not modify the reference fed to the inner system. Model predictive control with its structured approach to deal with both state and input constraints is a natural candidate to use as a reference governor. The closed loop dynamics of the inner system is in such cases used as prediction model in the MPC controller. Another approach is to use a nonlinear filter as reference governor (see Section 4.2.1). In this approach the bandwidth ($K$) is the parameter to
decide. Examples of this method can be found in Gilbert et al. [1995] and Gilbert and Kolmanovsky [1999]. Another method, used in the current control system in SKELDAR, is a reference governor based on override control with inspiration from Glattfelder and Schaufelberger [2004].

Override control is a control method used for systems with input and/or output constraints. In this method the control signal is allowed to be overridden in order to keep an output signal within its constraint limits. Override control can also be referred to as selector control, which refers to the selection of special control signals when the input/output is close to its constraints [Åström and Hägglund, 2006].

4.2 Earlier Work

The field of reference governors arose in the early 90s and a variation of approaches have been suggested since then. We will here summarize two important articles. The first, Gilbert et al. [1995], uses a first-order low-pass filter as RG. The second, Bemporad et al. [1997] is based on the predictive control methodology.

Note that we simply present two methods as comparison and overview of earlier work. We will not implement any of the two methods and instead we start our development of a RG from a more intuitive formulation in the framework of MPC (see Chapter 6).

4.2.1 RG via Low-pass Filtering, [Gilbert et al., 1995]

The arrangement which form the bases in this article is the same as in Figure 4.1. However, here we use the notation; \( r(k) \) for the nominal reference, \( w(k) \) for the modified signal and \( y(k) \in Y \) for constraints. The controlled system and its constraints are represented by the state space model

\[
x(k + 1) = Ax(k) + Bw(k)
\]
where the system is stable, $C$, $A$ is observable and the set $Y$ is compact.

In this article the authors suggest, as mentioned earlier, the use of a first-order low-pass filter as a RG. The design parameter will be the bandwidth $K$ and the modified reference is formed as:

$$ w(k + 1) = w(k) + K(r(k), x_G(k))(r(k) - w(k)), \quad x_G(k) = \left[ \begin{array}{c} w(k) \\ x(k) \end{array} \right] $$

(4.2)

The modified reference $w(k)$ is thus dependent of its own state and the state of the controlled process and it is required that $K(r(k), x_G(k)) \in [0, 1]$. The consequence of this is that $w(k + 1)$ belongs to the line segment between $w(k)$ and $r(k)$. One can see that when no constraints violation occur, $K(r(k), x_G(k))$ shall be equal to 1 and $w(k + 1) = r(k)$. With the possibility of constraint violations $K(r(k), x_G(k)) < 1$ and $w(k + 1)$ is driven towards $w(k)$. Before calculating $K$, (4.1) and (4.2) are written as the augmented system (4.3).

$$ x_G(k + 1) = A_Gx_G(k) + B_GK(r(k), x_G(k))(r(k) - [I \ 0]x_G(k)) $$

(4.3a)

$$ y(k) = C_Gx_G(k) \in Y $$

(4.3b)

$$ A_G = \begin{bmatrix} I & 0 \\ B & A \end{bmatrix}, \quad B_G = \begin{bmatrix} I \\ 0 \end{bmatrix}, \quad C_G = \begin{bmatrix} D & C \end{bmatrix}. $$

(4.3c)

For given $r(k)$ and $x_G(k)$ one has to make sure that no constraints will be violated, i.e. $y(\tau) \in Y$, $\forall \tau \geq k$. Since there is no information about $r(\tau)$ for $\tau > k$ it must be required that $x(\tau) \in O_\infty(A_G, C_G, Y)$, $\tau \geq k$, where $O_\infty(A_G, C_G, Y)$ is the maximal output admissible set defined by:

$$ O_\infty(A_G, C_G, Y) = \{x : C_GA_G^kx \in Y, k \in N\} \subset \mathbb{R}^n $$

(4.4)

The goal is then to find the largest value of $K$ such that the requirements are fulfilled. This is done via the maximization:

$$ K(r, x_G) = \max\{\alpha \in [0, 1] : A_Gx_G + B_G\alpha(r - [I \ 0]x_G) \in O_\infty(A_G, B_G, Y)\} $$

(4.5)

The maximization in (4.5) is done in every time instance and together with (4.2) a reference that ensures no constraint violation can be calculated. For stability and implementation reasoning we refer to the original article, Gilbert et al. [1995].

### 4.2.2 RG via Predictive Control, [Bemporad et al., 1997]

As mentioned above the authors of this article propose a reference governor based on predictive control. Here a memoryless RG is designed, which means that it does not depend on its own state, only the state of the system. The linear system that is studied here has the form:

$$ x(k + 1) = \Phi x(k) + Gg(k) $$

(4.6a)

$$ y(k) = Hx(k) $$

(4.6b)

$$ c(k) = H_c x(k) + Dg(k) $$

(4.6c)
where \( g(k) \) is the manipulated input which ideally should coincide with the nominal reference \( r(k) \) in the absence of constraints. \( y(k) \) is the tracked output and \( c(k) \) is the constrained vector that is required to be a member of the constraint set \( C \subset \mathbb{R}^{n_c} \), i.e.

\[
c(k) \in C, \forall k \geq 0.
\] (4.7)

Furthermore, the system in (4.6) is assumed to be stable and offset-free and the set \( C \) is assumed to be compact and convex.

In the following development of the reference governor the virtual command sequence is studied:

\[
v(k, \theta) = \gamma^k \mu + w, \quad \gamma \in [0, 1) \quad (4.8a)
\]

\[
\theta := [\mu^T \, w^T]^T \quad (4.8b)
\]

where \( \gamma \) is a parameter to be chosen by the control designer. In the article the authors show that there exist admissible solutions for an input in the form of (4.8). Now when we have presented the proposed command signal we can turn our attention to how the commands are chosen, namely how to choose \( \mu \) and \( w \).

First a quadratic cost function is defined:

\[
J(x(k), r(k), \theta) := \|\mu\|^2_{\Psi_\mu} + \|w - r(k)\|^2_{\Psi_w} + \sum_{i=0}^{\infty} \|y(i, x(k), \theta) - w\|^2_{\Psi_y} \quad (4.9)
\]

where \( \Psi_\mu > 0, \Psi_w > 0 \) and \( \Psi_y \geq 0 \). The term \( y(i, x(k), \theta) \) from the sum is the output from the system at time \( i \) when the command \( v(i, \theta) = \gamma^i \mu + w \) is used as an input, i.e \( g(i) = v(i, \theta) \).

The command sequence \( v(k, \theta) \) can now be calculated via:

\[
\theta(k) := \arg \min_{\theta \in \Theta} \{J(x(k), r(k), \theta) : c(\cdot, x(k), \theta) \subset C\} \quad (4.10)
\]

With these results in hand the action of the reference governor is selected as:

\[
g(k) = v(0, \theta(k)) = \mu(k) + w(k). \quad (4.11)
\]

Which gives a receding horizon control strategy since the optimization (4.10) is repeated in every time instant.

In the article the authors also give a proof of stability by using \( J(x(k), r, \theta(k)) \) as a Lyapunov function. They also suggest an algorithm used to truncate the sum in (4.9) in order to be able to implement the reference governor. For details about these topics we refer to Bemporad et al. [1997].

Notice that in the cost function (4.9) it is allowed to choose \( \Psi_y = 0 \), which will leave the dynamics of the system unchanged when the constraints are inactive.
As mentioned earlier the dynamics considered in this thesis is the heave (vertical) motion of the helicopter. In this chapter we will first present the state space model used in the thesis to describe the heave dynamics, followed by a stabilizing linear controller to the vertical model. A block diagram of the setup is illustrated in Figure 5.1. We then calculate the closed loop system of the vertical model and the stabilizing controller. For minimal complexity of the reference governor to be synthesized we suggest a few methods to reduce the state space model.

![Block diagram of the inner closed loop system with primary controller F and vertical system G.](image-url)
5.1 Vertical Model

First we will introduce a new coordinate system for the vertical state space model. In this coordinate system we consider positive velocity when the helicopter is ascending and negative velocity when descending, opposite to the coordinate system in Chapter 2.

Recall the vertical dynamics described by the differential equation in (2.28). The effects from the cyclic controls $\delta_{lat}$ and $\delta_{lon}$ are small compared to the collective pitch $\delta_{col}$ and the velocity damping effect. The vertical motion can then be approximated as (described in the new reference frame)

$$\dot{v} = \alpha v + \beta \delta_{col}$$

where instead of $Z_w$ and $Z_{col}$ we use $\alpha$ for the damping stability derivative and $\beta$ for the control derivative from the collective pitch. This is then augmented to include absolute position $h$ above ground.

$$\dot{v} = \alpha v + \beta \delta_{col} \quad (5.2a)$$

$$\dot{h} = v. \quad (5.2b)$$

Now we collect (5.2) in the familiar continuous state space form using matrix representation

$$\begin{bmatrix} \dot{v} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} \alpha & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ h \end{bmatrix} + \begin{bmatrix} \beta \\ 0 \end{bmatrix} \delta_{col}$$

$$A_G \quad B_G$$

\[5.3a\]
Both states \((v, h)\) can be measured and the corresponding \(C\)-matrix to the system thus equals the unit matrix. If all system states cannot be measured an observer can be used to estimate the unmeasured states [Glad and Ljung, 2006].

5.2 Primary Controller

The setup for which we will implement a reference modifying MPC includes a model describing the dynamics of the system (previous section) and a pre-designed linear feedback controller with certain characteristics, namely:

- The controller stabilizes the system
- The controller includes integral action for offset free tracking

Design of the stabilizing, offset free controller is not considered in this thesis. Any standard synthesis technique such as PID or LQ can be used. The controller used in this thesis is designed via LQ-control. For the sake of completeness it is described here and we will mainly present the controller to introduce what inputs, outputs and states it includes.

\[
\begin{align*}
\dot{\delta}_{\text{col,est}} & = A_F \begin{bmatrix} \delta_{\text{col,est}} \\ h_{\text{int}} \end{bmatrix} + B_F \begin{bmatrix} r \\ v \\ h \end{bmatrix} \\
\delta_{\text{col}} & = C_F \begin{bmatrix} \delta_{\text{col,est}} \\ h_{\text{int}} \end{bmatrix} + D_F \begin{bmatrix} r \\ v \\ h \end{bmatrix}
\end{align*}
\]  

(5.4a)  

(5.4b)

The controller has two states: an estimation of collective pitch actuator position \(\delta_{\text{col,est}}\) and an integral state \(h_{\text{int}}\). The input is the height reference to be tracked \(r\), the current height \(h\) and velocity \(v\). The output is the collective pitch control signal, which will be the input to the system in (5.3).

5.3 Inner Closed Loop System

The block representation of the setup including the closed loop inner system and the reference governor is illustrated in Figure 4.1. The internal model of the MPC will be some model describing the system to be controlled (the vertical model in (5.3)) under feedback of the primary controller (pre-designed LQ in (5.4)). This model is illustrated by the dashed block in the figure, denoted inner closed loop system.
5.3.1 Closing the Loop

To derive the internal model we need to compute the state space representation of the inner closed loop system with the height reference \( r \) as input and the measurable vehicle states \( v \) and \( h \) as outputs. To maintain the physical representation of the vehicle states in the inner closed loop system we simply augment the model in (5.3) with the controller (5.4) and substitute \( \delta_{col} \) with the expression (5.4b) given by the controller state space model. With some matrix manipulation we can calculate our new state space matrices \( \tilde{A}, \tilde{B} \) and \( \tilde{C} \) for the inner closed loop system as follows:

\[
\begin{bmatrix}
\dot{v} \\
\dot{h} \\
\dot{\delta}_{col,est} \\
\dot{h}_{int}
\end{bmatrix} =
\begin{bmatrix}
A_G + B_GD_F^{(v,h)} & B_GC_F \\
B_F^{(v,h)} & A_F
\end{bmatrix}
\begin{bmatrix}
v \\
h \\
\delta_{col,est} \\
\dot{h}_{int}
\end{bmatrix} +
\begin{bmatrix}
\dot{B_GD_F^{(r)}} \\
\dot{B_F}
\end{bmatrix}
\begin{bmatrix}
r
\end{bmatrix}
\]

(5.5a)

\[
y = 
\begin{bmatrix}
C_G & 0^{2 \times 2}
\end{bmatrix}
\begin{bmatrix}
v \\
h \\
\delta_{col,est} \\
\dot{h}_{int}
\end{bmatrix}
\]

(5.5b)

Here we have matrices \( A_G, B_G \) and \( C_G \) as in system (5.3) and \( A_F, B_F, C_F \) and \( D_F \) as in (5.4). The superscript in the matrices in \( \tilde{A} \) and \( \tilde{B} \) denotes the rows and columns of the matrix belonging to the corresponding subscripted states.

5.3.2 Model Reduction

The inner closed loop system has four states, seen in (5.5). Since each state of the prediction model in the MPC will typically correspond to at least \( N \) optimization variables in the QP-problem it is desirable to have a model with as few states as possible. By closer examination of the inner closed loop system we can see that we have one state whose dynamics is much faster than the others. If we plot the poles and zeros of the closed loop (see Fig. 5.3a) we can see that the system has one pole and one zero far out in the left half-plane of the pole-zero plot. This corresponds to states with fast dynamics. The question is which of the states that corresponds to the pole-zero pair. In Figure 5.3b we plot the step response for the system states. Here it is easy to conclude that the collective pitch estimate \( \delta_{col,est} \) corresponds to the fast state.

Following the method suggested in Glad and Ljung [2003] in Section 3.6 for elimination of fast dynamics we can remove \( \delta_{col,est} \) from the state space model without significant loss of the input-output characteristics. In particular, the stationary characteristics will be unchanged.

We rewrite the state space model in (5.5) to separate the states that we will still keep in the model (\( \tilde{x} := [v \ h \ h_{int}]^T \)) from the state that we want to remove (\( \delta_{col,est} \)).
5.3 Inner Closed Loop System

(a) Pole-zero diagram showing a pole-zero pair far out in the left half plane corresponding to fast dynamics in the system.

(b) Step response of the closed loop system. Note the fast response from reference \( r \) to the third state \( \delta_{\text{col,est}} \).

Figure 5.3: Pole-zero diagram and step response of the inner closed loop system.

\[
\begin{bmatrix}
\dot{\tilde{x}} \\
\dot{\delta}_{\text{col,est}}
\end{bmatrix} = 
\begin{bmatrix}
\tilde{A}_{11} & \tilde{A}_{12} \\
\tilde{A}_{21} & \tilde{A}_{22}
\end{bmatrix}
\begin{bmatrix}
\tilde{x} \\
\delta_{\text{col,est}}
\end{bmatrix} + 
\begin{bmatrix}
\tilde{B}_1 \\
\tilde{B}_2
\end{bmatrix} r
\]  

(5.6a)

\[
y = 
\begin{bmatrix}
\tilde{C}_1 & \tilde{C}_2
\end{bmatrix}
\begin{bmatrix}
\tilde{x} \\
\delta_{\text{col,est}}
\end{bmatrix}
\]  

(5.6b)

By assuming \( \delta_{\text{col,est}} = 0 \) we can replace the dynamics of \( \delta_{\text{col,est}} \) with the static expression:

\[
0 = \tilde{A}_{21}\tilde{x} + \tilde{A}_{22}\delta_{\text{col,est}} + \tilde{B}_2 r
\]

\[\delta_{\text{col,est}} = -\tilde{A}_{22}^{-1}(\tilde{A}_{21}\tilde{x} + \tilde{B}_2 r)\]

Substituting \( \delta_{\text{col,est}} \) into (5.6) we get:

\[
\dot{\tilde{x}} = \tilde{A}_r \tilde{x} + \tilde{B}_r r
\]

(5.7a)

\[y = \tilde{C}_r \tilde{x}\]

(5.7b)

with the state matrices calculated as

\[
\tilde{A}_r = \tilde{A}_{11} - \tilde{A}_{12} \tilde{A}_{22}^{-1} \tilde{A}_{21}, \quad \tilde{B}_r = \tilde{B}_1 - \tilde{A}_{12} \tilde{A}_{22}^{-1} \tilde{B}_2,
\]

\[
\tilde{C}_r = \tilde{C}_1 - \tilde{C}_2 \tilde{A}_{22}^{-1} \tilde{A}_{21}.
\]

If we again take a look at the pole-zero diagram in Figure 5.4 now for the reduced model (5.7) we can see that the fast pole/zero pair has been removed. In Figure 5.4 we can also see that we have a pole-zero cancellation close to the imaginary axis. Since the pole-zero pair is stable it is safe to let them cancel each other
Figure 5.4: Pole-zero diagram when fast dynamics has been removed. Notice the pole-zero cancellation close to the imaginary axis.

out [Ogata, 2010]. For this we compute the transfer functions $G_v(s)$, $G_h(s)$ and $G_{h_{int}}(s)$ from reference $r$ to the remaining three states $v$, $h$ and $h_{int}$. With $s$ as the Laplace variable we get:

$G_v(s) = K_v \frac{s(s + z)}{(s + p)(s^2 + 2as + b)}$  \hspace{1cm} (5.8)

$G_h(s) = K_h \frac{(s + z)}{(s + p)(s^2 + 2as + b)}$ \hspace{1cm} (5.9)

$G_{h_{int}}(s) = K_{h_{int}} \frac{(s + z)(s + c)}{(s + p)(s^2 + 2as + b)}$  \hspace{1cm} (5.10)

Here we have $z \approx p$ as predicted by the pole-zero diagram, which means that we do not have an exact pole-zero cancellation. If we assume $z = p$ we get

$G_v(s) = K_v \frac{s}{(s^2 + 2as + b)}$  \hspace{1cm} (5.11)

$G_h(s) = K_h \frac{1}{(s^2 + 2as + b)}$ \hspace{1cm} (5.12)

$G_{h_{int}}(s) = K_{h_{int}} \frac{(s + c)}{(s^2 + 2as + b)}$  \hspace{1cm} (5.13)

Now by inverse Laplace transformation we can again calculate our state equations. This yields

$\dot{v} = -2av - \frac{K_v}{K_h} bh + K_v r$ \hspace{1cm} (5.14)
5.3 Inner Closed Loop System

\[ \dot{h} = \frac{K_h}{K_v} v \]  
(5.15)

\[ \dot{h}_{int} = \frac{K_{h_{int}}}{K_v} (-2a + c)v + \frac{K_{h_{int}}}{K_h} bh + K_{h_{int}} r. \]  
(5.16)

In the state equations above it can be seen that the third state \( h_{int} \) is no longer observable in the other two states, height \( h \) and velocity \( v \), which is the two output signals. If we compute the observability matrix \( O(A, C) \) for the corresponding state space model and calculate its rank, we find that the observability matrix does not have full rank and the system is not observable [Glad and Ljung, 2003].

We exclude the state equation for the integral state \( h_{int} \) and write our reduced and approximated state space model. Note that \( K_v = K_h \).

\[
\begin{bmatrix}
\dot{v} \\
\dot{h}
\end{bmatrix} =
\begin{bmatrix}
-2a & -b \\
1 & 0
\end{bmatrix}
\begin{bmatrix}
v \\
h
\end{bmatrix}
+ \begin{bmatrix}
K_v \\
0
\end{bmatrix} \delta_{col}
\]  
(5.17a)

\[
y = \begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
v \\
h
\end{bmatrix}
\]  
(5.17b)

We have now managed to reduce the original closed loop model of four states to an approximated model of two states, saving at least \( 2N \) optimization variables in our MPC. In Figure 5.5 we plot the step response of the original model (dashed line) compared with the reduced model (solid line). Note the small differences in response.

**Figure 5.5:** Response comparison between the original closed loop system (dashed green line) and the reduced closed loop system (solid blue line).
6
Reference Governor for Heave Dynamics

In this chapter we will present the development and investigation that is the core of this thesis. First we will motivate the ideas behind our reference manipulating controllers. We provide the development process and results of three different RGs. We also add a stabilizing property to one of the RGs and provide a proof of stability.

6.1 Defining the Problem

The fundamental idea of our development process is that we want to form an RG that affects the controlled system as little as possible. Strictly speaking we want to modify the nominal reference only when constraint violation will occur, otherwise set \( r = r_{\text{nom}} \). The signal \( r_{\text{nom}} \) is a height set-point, commanded by a higher level control system. For our solution to be useful to SAAB the suggested controller is supposed to fit the existing SKELDAR framework, i.e. be able to work as an add-on as described in Section 1.3.

With these requests in mind we define the cost function

\[
J = \sum_{i=0}^{N} \| r_i - r_{\text{nom}} \|^2_R.
\] (6.1)

Here only the deviation between \( r \) and \( r_{\text{nom}} \) is penalized. With this cost function one can see that \( r \) will be equal to \( r_{\text{nom}} \) when possible, as desired to get a passive RG in nominal cases.

To fulfill the purpose of the controller, i.e. to ensure that the vertical velocity will remain in the flight envelope, the cost function (6.1) is complemented with con-
constraints for the system dynamics and vertical velocity limits. These constraints are formed as:

\[ x_{i+1} = Ax_i + Br_i \quad \forall i = 0, \ldots, N \]  
\[ v_{\text{min}} \leq v_i \leq v_{\text{max}} \quad \forall i = 0, \ldots, N \]

where \( A \) and \( B \) are system matrices for the closed loop system in (5.17) discretized in 50 Hz. The vertical velocity limits used in simulations are, \( v_{\text{min}} = -1.1 \text{ m/s} \) and \( v_{\text{max}} = 1.1 \text{ m/s} \). Combining the cost function (6.1) with the constraints (6.2) gives the problem formulation (6.3).

\[
\text{minimize} \quad \sum_{i=0}^{N} \| r_i - r_{\text{nom}} \|_R^2 \\
\text{subject to} \quad x_{i+1} = Ax_i + Br_i \quad \forall i = 0, \ldots, N \\
\quad \quad \quad \quad v_{\text{min}} \leq v_i \leq v_{\text{max}} \quad \forall i = 0, \ldots, N
\]

As stated in Section 3.4 one has to handle the feasibility problem that can occur in the optimization problem. We will here use the described approach with relaxed constraints. By adding slack variables \( \epsilon_i \) to (6.3c) we ensure that there will always exist a solution to the optimization problem (6.3). It should be stated that in the application, the velocity constraint is in fact not a hard constraint. Thus it is not a problem, rather desired, to add relaxed constraints to get a rubber band effect of the constraint violation when utilizing the slack variable, bringing the helicopter back to the flight envelope in a controlled manner. Now our initial problem formulation, which will serve as a starting point for our surveys, has the form:

\[
\text{minimize} \quad \sum_{i=0}^{N} \| r_i - r_{\text{nom}} \|_R^2 + \| \epsilon_i \|_R^2 \\
\text{subject to} \quad x_{i+1} = Ax_i + Br_i \quad \forall i = 0, \ldots, N \\
\quad \quad \quad \quad v_{\text{min}} - \epsilon_i \leq v_i \leq v_{\text{max}} + \epsilon_i \quad \forall i = 0, \ldots, N \\
\quad \quad \quad \quad \epsilon_i \geq 0 \quad \forall i = 0, \ldots, N
\]

When running initial tests with this problem formulation the results seem satisfying at a first look. The reference is tracked and the vertical velocity lies within its boundaries, as can be seen in Figure 6.1. However, if we take a closer look at the internal control signal \( \delta_{\text{col}} \) we can see an undesired behavior. Figure 6.1 shows rapid changes in the control signal. The control signal \( \delta_{\text{col}} \) is a signal to a physical servo controlling the collective pitch. Due to physical limitations it is not possible to instantaneously change the servo position. To minimize wear and tear it is also desirable to not drive the servo too hard.

### 6.1.1 Simulation Environment

Our investigation and algorithm development will be carried out in MATLAB using the modeling language YALMIP. For simulations we will also use SIMULINK to setup the scenarios. In SIMULINK it is easy to add disturbances and to get
6.1 Defining the Problem

![Graph](image)

**Figure 6.1:** System behavior for a step in nominal reference and with the initial problem formulation (6.4). Here we have used $N = 10$ and quadratic penalty on epsilon with $\rho = 1000$

a clear overview of the setup. The model used in the simulation is the vertical model under state feedback of the primary controller as described in Chapter 5. We therefore simulate the non-reduced model (5.5), while the MPC still uses the reduced model as prediction model (5.17). By simulating the inner controller and vertical model separately we can also add input disturbances to the inner control loop. In Chapter 7 we simulate using more advanced simulation systems available at SAAB.

### 6.1.2 Simulating Disturbances

So far we have only studied an undisturbed case but further on we will also make simulations when the system is subject to different disturbances. We will study two disturbances, both physically meaningful for an autonomous helicopter such as SKELDAR. These disturbances are approximations of disturbances that can be simulated in ARES. The disturbances are:

- **External force:** A disturbance that is pushing the helicopter downwards (or upwards), simulating e.g. an air pocket or a gust of wind. This is implemented by adding a constant bias term to $\delta_{col}$, which will propagate through the system similarly to an external force acting on the fuselage. Further this will be referred to as an external force disturbance.

- **Height bias:** A sudden change in the height measurement. Implemented by adding a bias to the state $h$. This disturbance simulates a sudden change in
the height measurement that can occur when changing between SKELDAR’s two GPS units or between satellite sources. The new measurement might differ drastically from the previous measurement. In the current control system the height signal is low-pass filtered before entering the system. Therefore we use a ramp disturbance instead of a step for a simple approach to mimic the measurements of the true system. Further this will be referred to as a height bias disturbance.

Explanatory plots can be found in Figure 6.2.

**Figure 6.2:** Height change of the system in presence of the different disturbances as well as the actual disturbance and the behavior of the inner control system.

### 6.2 RG with Pre-filtered Nominal Reference

In order to avoid the undesired behavior of the internal control signal $\delta_{col}$, illustrated in Figure 6.1, we suggest pre-filtering of the nominal reference. We want to avoid large rate of change of $r$ in undisturbed cases. One approach to solve this is to limit the slew rate and acceleration rate of $r_{nom}$. This will prevent the rapid changes in $r$ that causes the rapid changes in $\delta_{col}$. Considering the desire that our solution shall fit the SKELDAR framework gives further support for this approach since the filter can be considered as an additional subsystem added to the control loop of the heave dynamics. In the current control system in SKELDAR a similar pre-filter is already implemented. Figure 6.3 shows the setup including the pre-filter and Figure 6.4 shows the result of pre-filtering a step in the set-point reference $r_{sp}$. Rate limits used in the filter are: $\Delta r_{lim} = 1$ m/s and $\Delta^2 r_{lim} = 1$ m/s$^2$. This will make the slew rate limitation less than the state constraint on $v$, making it possible for $r$ to track $r_{nom}$ as long as no disturbances are present. In the sequel we will refer to this reference governor as the pre-filter RG.

For this problem setup we use quadratic penalty of the slack variables. The moti-
Figure 6.3: Block diagram setup with the suggested pre-filter with $r_{sp}$ as input and $r_{nom}$ as output.

vation is that the rate of change of the reference signal is lower than the vertical velocity constraints and the constraints will therefore not be active during a set-point change. Thus exact penalty, as explained in Section 3.4, is not motivated since we do not expect the constraints to be active in nominal conditions, which could otherwise cause unnecessary utilization of the slack variable. Figure 6.5 shows results from a simulation similar to the simulations in Figure 6.1 but with use of the suggested pre-filter. As expected we can see that the nominal reference is followed precisely without violation of the vertical velocity constraint. Additionally $\delta_{col}$ behaves a lot smoother, which was the intention of introducing the pre-filter.

We have now achieved desirable behavior in absence of disturbances, next we will show what happens when this is not the case. First it should be stated that when the system is under some sort of disturbance we have to allow the RG to be more aggressive than in nominal cases. This will cause behavior of $\delta_{col}$ that is undesired in undisturbed cases but necessary in disturbed cases.

Figure 6.6–6.8 show how the RG reacts to different disturbances. First we see a simulation with $r_{nom} = 0$ (constant) but where the external force disturbance enters the system at $t = 1$. For this simulation we also provide a plot comparing the behavior without the RG. Here we can see that the helicopter is forced upwards and the vertical velocity is increasing. In order to avoid large violations of the velocity constraint the RG is temporarily decreasing the reference. This act tells the system to descend the helicopter and counteract the effect of the disturbance. For this kind of disturbance the collective pitch finds a new trim position where it holds the helicopter in hover.

Next figure (Fig. 6.7) shows the same setup but now with a height bias disturbance of 4 m entering the system at $t = 1$. When the height measurement is biased the inner control system tries to correct the height by descending to the new measured height. If this is done too rapidly the vertical velocity tends to become too large. By raising the reference the RG eases this correction and manages to hold the vertical velocity around its boundaries. Note that the RG finds it optimal to violate the constraint (by using some slack) in order to return to
**Figure 6.4:** Comparison between the set-point reference and the filtered nominal reference.

**Figure 6.5:** System behavior when using a filtered signal as nominal reference and with initial problem formulation (6.4). Here we have used $N = 10$ and quadratic penalty on epsilon with $\rho = 1000$. 
\[ r = r_{\text{nom}}. \]

The third and last figure (Fig. 6.8) shows a case where a disturbance enters the system during a descent. The disturbance is the same external force disturbance as in Figure 6.6a but is pushing downwards at \( t = 7 \) and the set-point reference to be tracked is \( r_{sp} = -10 \). As expected the system behaves similarly as in 6.6a, when the disturbance enters the system the reference is raised in order to suppress the constraint violation.

(a) \( N = 10 \) and quadratic penalty on \( \epsilon \) with \( \rho = 1000 \).

(b) Without RG.

Figure 6.6: System behavior when disturbed with the external force disturbance.
Figure 6.7: System behavior when disturbed with the height bias disturbance. Here we have used $N = 10$ and quadratic penalty on epsilon with $\rho = 1000$

Figure 6.8: System behavior with a filtered set-point reference of $-10$ m and disturbed with the external force disturbance. Here we have used $N = 10$ and quadratic penalty on epsilon with $\rho = 1000$
6.2 RG with Pre-filtered Nominal Reference

6.2.1 Parameters and Tuning

The parameters for the pre-filter RG are the prediction horizon \( N \), the penalty term \( R \) for the reference error \( \|r_i - r_{nom}\| \) and the penalty term \( \rho \) for the slack variables \( \epsilon_i \). Since the result of the optimization problem depends on the ratio between \( R \) and \( \rho \) and not the absolute values, we will use a fixed \( R = 1 \) and vary the value of \( \rho \). Below we give a short overview of the two parameters.

- \( N \): The prediction horizon decides how many steps to predict the future behavior of the system. The parameter \( N \) is directly connected to the size of the optimization problem, since for each additional prediction step there will be \( n + m + n_{eps} \) more optimization variables. Where \( n \) is the number of states, \( m \) the number of inputs and \( n_{eps} \) the number of slack variables. The number of constraints will also depend on the prediction horizon.

- \( \rho \): The penalty term is a trade-off between the desire to hold vertical velocity \( v \) within its boundaries (large value) and the desire to drive \( r \) to \( r_{nom} \) as fast as possible (low value).

We will in this section try different combinations of settings and explain the differences in system behavior, to give a understanding for how to choose them in order to get the RG to behave as desired.

Since the constraints are always inactive in a nominal case and the MPC will set \( r = r_{nom} \) we will, in these cases, not expect a difference in behavior for variations in the parameters. Instead we will focus on the behavior for the system in presence of a disturbance. We will consider a case when the system is under the height bias disturbance. Figure 6.9–6.11 show plots of the system for different values of \( N \) and \( \rho \). We start with a high penalty term \( \rho \) and then decrease it gradually in the following figures. For each value of \( \rho \) we present two results with different \( N \).

We start by studying Figure 6.9 where we can see that the constraint violation is almost insignificant. This is as expected due to the relatively high value of \( \rho \). By looking at the reference we can see a difference between the two values of \( N \). In Figure 6.9b with the higher value of \( N \) (\( N = 25 \)) we can see that the RG has predicted the violation earlier and can therefore try to counteract this at an earlier time then the RG with \( N = 10 \). The main advantage of this can be seen in the plots of the collective pitch. With a longer prediction horizon the maximal collective pitch is \( -13.5^\circ \), to be compared with \( -14.5^\circ \) that is the maximum for \( N = 10 \).

Next we turn our attention to Figure 6.10 where \( \rho \) is decreased to 100, otherwise it is the exact same simulation setup as before. Here we can see a smoother behavior of the reference governor. The pattern is the same as for \( \rho = 1000 \), a longer prediction horizon gives a less aggressive inner controller. In this case \( \delta_{col} = -13.5^\circ \) for \( N = 25 \) and \( -14.5^\circ \) for \( N = 10 \). Decreasing the constraint penalty term has, as expected, given a larger violation of the constraint. We can also notice that this violation is slightly larger with the shortsighted RG. Note that, since
the control error of the height tracking in the primary controller will be smaller when the reference $r$ is closer to the height measurement $h$, the more aggressive RG (with larger $\rho$) will actually result in smaller collective pitch commands as observed in the simulations when comparing Fig. 6.9 and Fig. 6.10 for $N = 25$. This might not be intuitively expected from a more aggressive setup, but since the RG with larger penalty will increase the reference $r$ more the error between height $h$ and reference $r$ in the inner controller will be smaller, which in turn results in smaller actuator value of $\delta_{col}$.

The last plot in this sequence (Fig. 6.11) shows simulations where $\rho$ is decreased further and $\rho = 10$ is used. For this $\rho$ we get an even smoother RG and it once again follows the pattern that a longer prediction horizon gives a less aggressive inner controller, $\delta_{col} = -14.4^\circ$ compared to $\delta_{col} = -15.4^\circ$. The relatively low value of $\rho$ is now starting to give an undesired impact on the vertical velocity. Due to the small penalty we get large violations of the constraint, $v = -1.83$ m/s for $N = 10$ and $v = -1.97$ m/s for $N = 25$.

We have now seen the system behavior for a variety of combinations of the parameters $N$ and $\rho$. To sum up we conclude that low values of $\rho$ give a smoother RG, at the cost of a large violation of the velocity constraint. The prediction horizon $N$ does not have the same large impact to the system. We can conclude that by increasing $N$ we can get a slightly smoother RG and the constraint violation will be less, due to the earlier response of the RG. When considering the collective pitch we see similar results, namely that small values of the parameters give larger collective pitch. In Figure 6.12 we present plots that summarize the characteristics for the parameters.
Figure 6.10: System behavior under a height bias disturbance with $\rho = 100$ and different $N$.

Figure 6.11: System behavior under a height bias disturbance with $\rho = 10$ and different $N$. 
6.3 RG with Slew Rate Constraints

Instead of using the pre-filter suggested in Section 6.2 the rate of change of the height reference can be handled directly by the MPC-formulation of the RG. In model predictive control this is commonly referred to as slew rate and the standard procedure is to add a constraint that limits the rate of change of the output from the MPC. We also want to limit the acceleration of the reference, which we do in the same manner by a constraint that limits the acceleration. We augment the MPC-formulation (6.4):

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=0}^{N} \|r(i) - r_{\text{nom}}\|_R^2 + \|\epsilon(i)\|_p^2 \\
\text{subject to} & \quad x(i + 1) = Ax(i) + Br(i) \quad \forall i = 0, \ldots, N \\
& \quad v_{\min} - \epsilon(i) \leq v(i) \leq v_{\max} + \epsilon(i) \quad \forall i = 0, \ldots, N \\
& \quad \Delta r_{\min} \leq \Delta r(i) \leq \Delta r_{\max} \quad \forall i = 0, \ldots, N \\
& \quad \Delta^2 r_{\min} \leq \Delta^2 r(i) \leq \Delta^2 r_{\max} \quad \forall i = 0, \ldots, N
\end{align*}
\]

with \( \Delta r(i) = r(i) - r(i - 1) \) and \( \Delta^2 r(i) = \Delta r(i) - \Delta r(i - 1) \). Since the constraints now are dependent of reference outputs from previous time instants the MPC typically needs to be augmented with additional states to keep track of past MPC outputs. The past reference outputs are given to the RG by including them in the initial conditions. We use a quadratic penalty function for the slack variable, since the rate limits will be stricter than the velocity constraint (we use the same limits as for the pre-filter), the velocity constraint will not be active in nominal conditions and exact penalty is not motivated (as discussed already in the pre-filter RG).
In Figure 6.13a we can see that we get almost the same results as for the pre-filtered case. For this setup we can see some overshoot of the manipulated reference \( r \). This arises from the RG being too short sighted with prediction horizon \( N = 10 \). When the reference \( r \) is about to reach the nominal reference \( r_{\text{nom}} \) the RG is picking this up too late and because we have constraints on the acceleration (and deceleration) of the reference the controller can not decelerate in time, and we get overshoot. In Figure 6.13b we have increased the prediction horizon and the overshoot is reduced.

If we repeat the simulations for the external force disturbance (Fig. 6.14a) we can see that the RG has lost some of its authority to avoid violations of the velocity constraint. Since the rate constraints added are formulated as hard constraints in the RG the controller will never violate them and will instead be less aggressive when disturbances enter the system. Comparison with a simulation without the RG (Fig. 6.14b) shows only a small difference in the descending velocity. Here the velocity dip reaches \( v = -1.74 \, \text{m/s} \) and with the MPC \( v = -1.71 \, \text{m/s} \). Considering the oscillative behavior with this RG setup, the control performance is better without the RG, and therefore this formulation is not recommended to handle disturbances that may cause violation of the constraints.
To increase the RG’s authority we again suggest some modifications of the formulation:

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=0}^{N} \|r(i) - r_{\text{nom}}\|_{R}^{2} + \|\varepsilon_{1}(i)\|_{\rho_{1}}^{2} + \|\varepsilon_{2}(i)\|_{\rho_{2}}^{2} + \|\varepsilon_{3}(i)\|_{\rho_{3}}^{2} \\
\text{subject to} & \quad x(i + 1) = Ax(i) + Br(i) \quad \forall i = 0, \ldots, N \quad (6.6a) \\
& \quad v_{\text{min}} - \varepsilon_{1}(i) \leq v(i) \leq v_{\text{max}} + \varepsilon_{1}(i) \quad \forall i = 0, \ldots, N \quad (6.6b) \\
& \quad \Delta r_{\text{min}} - \varepsilon_{2}(i) \leq \Delta r(i) \leq \Delta r_{\text{max}} + \varepsilon_{2}(i) \quad \forall i = 0, \ldots, N \quad (6.6c) \\
& \quad \Delta^{2} r_{\text{min}} - \varepsilon_{3}(i) \leq \Delta^{2} r(i) \leq \Delta^{2} r_{\text{max}} + \varepsilon_{3}(i) \quad \forall i = 0, \ldots, N \quad (6.6d)
\end{align*}
\]

Here the slack variables are indexed with \( j = 1, 2, 3 \). By introducing slack variables in the rate constraints as well the RG is allowed to violate the constraints if necessary, e.g. under disturbances, and if we choose the penalties \( \rho_{2} \) and \( \rho_{3} \) with care it is possible to achieve desired behavior in nominal conditions.

Applying quadratic penalties for \( \varepsilon_{2} \) and \( \varepsilon_{3} \) results in acceptable performance under the disturbance as can be seen in Figure 6.15. The RG reacts more aggressively and the maximum velocity dip is \( v = -1.45 \text{ m/s} \).

For this combination of penalties we no longer get the desired behavior in nominal conditions. In Figure 6.16a we plot the same step input of the nominal reference \( r_{\text{nom}} \) as in Figure 6.13a. Looking closely the manipulated reference \( r \) is slightly curved through the ascent, indicating that the rate constraint \( \Delta r \) is being violated. This can also be seen in Figure 6.16b where we plot \( \Delta r \) and \( \Delta^{2} r \). The acceleration constraint \( \Delta^{2} r \) is also violated which is easy to see at \( t = 1 \text{ s} \) when the step is applied, showing no smooth acceleration of the reference.

(a) Horizon \( N = 10 \) and penalty \( \rho = 1000 \).

(b) Without RG.

Figure 6.14: System behavior under the external force disturbance described in Section 6.1 with and without the RG.
6.3 RG with Slew Rate Constraints

**Figure 6.15:** System behavior under the external force disturbance. Horizon $N = 10$, quadratic penalties for $\epsilon_1$, $\epsilon_2$ and $\epsilon_3$ with $\rho_1 = 1000$, $\rho_2 = 10000$ and $\rho_3 = 10000$.

**Figure 6.16:** System behavior under nominal conditions including reference rates. Horizon $N = 10$, quadratic penalties for $\epsilon_1$, $\epsilon_2$ and $\epsilon_3$ with $\rho_1 = 1000$, $\rho_2 = 10000$ and $\rho_3 = 10000$. 
Recall the discussion of relaxed constraints in Section 3.4. Quadratic penalty will always, to some extent, result in violation of the constraint when it is active. The $\Delta r$ constraint will always be active in nominal conditions for a step in reference $r_{nom}$ and in Figure 6.16b we can see that it is being violated all through the ascent. To prevent this we instead use a linear penalty function for $\epsilon_2$. Using the linear penalty function $\rho |\epsilon_2|$ we get the results for the nominal case in Figure 6.17, this time without constraint violations of $\Delta r$.

In Figure 6.18 we plot the disturbed case showing similar behavior as when we use quadratic penalty. Now the maximum velocity dip is $v = -1.36$ m/s, slightly less than before.

Further, to prevent violation of the acceleration constraint $\Delta^2 r$ and to recreate the behavior in nominal conditions, we use linear penalty for $\epsilon_3$ as well. For the reference step $r_{nom} = 5$ m at $t = 1$ s we get an almost identical behavior as for the RG with hard rate constraints (Fig. 6.19).

Simulating the external force disturbance again shows that the ascending velocity is still counteracted by the RG but now the reference $r$ clearly shows different phases of the RG. The breakpoints that can be seen at around $t = 2$ s and $t = 2.4$ s are typical for an MPC with linear penalty in the cost function. Quadratic penalty results in smoother utilization of the slack variable while the linear penalties results in an on-off behavior. This can also be seen in Figure 6.20 where we plot the rates $\Delta r$ and $\Delta^2 r$. The rate constraint violations are either large or none at all.

\begin{figure}[h]
\centering
\begin{subfigure}[b]{0.45\textwidth}
\includegraphics[width=\textwidth]{nominal_references.png}
\caption{System states and references.}
\end{subfigure}
\begin{subfigure}[b]{0.45\textwidth}
\includegraphics[width=\textwidth]{rates.png}
\caption{Rates $\Delta r$ and $\Delta^2 r$.}
\end{subfigure}
\caption{System behavior under nominal condition. Horizon $N = 10$, quadratic penalties for $\epsilon_1$ and $\epsilon_3$ with $\rho_1 = 1000$ and $\rho_3 = 10000$ and linear penalty for $\epsilon_2$ with $\rho_2 = 100$.}
\end{figure}
6.3 RG with Slew Rate Constraints

**Figure 6.18:** System behavior under the external force disturbance. Horizon $N = 10$, quadratic penalties for $\epsilon_1$ and $\epsilon_3$ with $\rho_1 = 1000$ and $\rho_3 = 10000$ and linear penalty for $\epsilon_2$ with $\rho_2 = 100$.

**Figure 6.19:** System behavior under nominal conditions. Horizon $N = 10$, quadratic penalty for $\epsilon_1$ with $\rho_1 = 1000$ and linear penalties $\epsilon_2$ and $\epsilon_3$ with $\rho_2 = 100$ and $\rho_3 = 1000$. 

(a) System states and references.

(b) Rates $\Delta r$ and $\Delta^2 r$. 
Figure 6.20: System behavior under the external force disturbance. Horizon $N = 10$, quadratic penalty for $\epsilon_1$ with $\rho_1 = 1000$ and linear penalties $\epsilon_2$ and $\epsilon_3$ with $\rho_2 = 100$ and $\rho_3 = 1000$.

Figure 6.21: System behavior under nominal conditions with step input $r_{nom} = 20$. Horizon $N = 10$, quadratic penalty for $\epsilon_1$ with $\rho_1 = 1000$ and linear penalties $\epsilon_2$ and $\epsilon_3$ with $\rho_2 = 100$ and $\rho_3 = 1000$. 
The difficulty of the RG formulation with relaxed rate constraints is that the behavior of the controller will depend on the initial states of the controller, namely the height $h$ and nominal reference $r_{nom}$. The controllers are tuned for a nominal case where the nominal reference changes from 0 meters to 5 meters, where we have shown good performance in both disturbed and nominal cases. If we instead simulate for some other set-point as we do in Figure 6.21 we see a completely different behavior. Here the contribution of the first term in the cost function, $\|r - r_{nom}\|_2^2$, will be larger at first when the difference in manipulated reference and nominal reference is large. This will again result in violation of rate constraints as shown before. Since in the realistic case, the set-point could vary between a few meters to a couple of hundred meters, this kind of RG formulation becomes very hard tuned and complex to analyze. Also recall that the original control problem studied in this thesis is, with two states in the target system and one constrained state, relatively small compared to other practical control problems. This indicates that the RG formulation studied in this section soon becomes too complex for its purpose.

6.4 RG with Resampled Prediction Model

One common challenge in model predictive control is to construct a problem formulation such that the prediction horizon is long enough without getting to large optimization problems. With long enough we mean such that important dynamics is covered during the time horizon. Since there is always a limit in computational power there is an upper bound of the prediction horizon $N$. In order to predict further into the future without increasing $N$ we have studied a problem formulation where the prediction model is resampled using a lower sampling rate, but still applying the control action of the MPC in the original sampling rate. This makes it possible for the controller to see further without increasing the prediction steps $N$. Note that this is something we have studied without support from the literature and we can make no statements regarding stability or convergence. The standard stability reasoning does not hold for this formulation since the initial state given to the MPC at each time interval will not be the state predicted in the previous time interval. One can however make comparisons with move-blocking [Cagienard et al., 2007] where the input is considered fixed over several time steps.

A drawback of our approach is that we do not take into account what is happening between the sampling points within the prediction model. If the controlled system has fast dynamics this might be a problem, since hard constraints can possibly be violated during these intervals. Our controlled system has relatively slow dynamics and the constraints are soft so we can tolerate some violation.

Simulations under height bias disturbance in SIMULINK indicate that this approach gives promising results. Both in terms of system behavior and solve time. Figure 6.22 presents three simulations where different sampling rate of the prediction model has been used. In the top plot of the figure, prediction horizon
$N = 25$ is used with the original sampling time of $T_s = 0.02$ s. The middle plot shows the resampled RG using $T_s = 0.1$ s with $N = 5$, which will predict as far into the future as the top formulation but with only 5 prediction steps. As comparison the bottom plot is simulated using $T_s = 0.02$ s but with $N = 5$. We can see that the top and middle plot reacts almost at the same time, at around $t = 1.3$ s, but the bottom plot does not take action until $t = 1.38$ s. Also worth noticing is that the resampled formulation is slightly less aggressive than for original sampling rate, since the manipulated reference $r$ is smaller.

![Graph showing simulations comparing the resampled RG with $T_s = 0.1$ s to the RG with original sampling time $T_s = 0.02$ s.](image)

**Figure 6.22:** Simulations comparing the resampled RG with $T_s = 0.1$ s to the RG with original sampling time $T_s = 0.02$ s.

See Section 7.1 for further investigation of the formulation using resampled prediction model in ARES simulations.

### 6.5 Stability

In this section we will present modifications of the RG in Section 6.2 to guarantee stability of the controller. For the stability analysis the model is assumed to be perfect and the system is under no disturbances. We will also omit the slack variables. The proof is based on a dual mode methodology, described in Section 3.2.2, and relies on a terminal set constraint. It is carried out in three steps. First we show that a positive invariant set for the system exists. The second step is to ensure that the system can be driven to the terminal set in $N$ steps, while the third part treats what happens when the system has reached the set.

**Step 1:** Since the inner system is stabilized and offset-free via the primary con-
troller there exists a positive invariant set $T$ for which it holds that: $Ax_k + B\bar{r} \in \mathcal{X}$ for a constant $\bar{r}$. The set $\mathcal{X}$ is the state constraint set and $T \subseteq \mathcal{X}$.

Figure 6.23 shows the positive invariant set used as terminal set $T$. Note that the velocity target will always be zero. Shifting the terminal set to the height reference $r_{N-1}$ will only shift $T$ up or down along the height-axis in the figure, which means that the terminal set will always lie within the velocity constraints (dashed lines in the figure). This is a special case for our specific problem, and will not always be true for all MPCs. If the velocity target would not always be zero, the terminal set would be shifted outside the velocity constraints. We calculate the positive invariant set using MPT3 by Herceg et al. [2013].

**Step 2:** We expand the problem formulation (6.3) to include the terminal set $T$:

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=0}^{N-1} \| r_i - r_{nom} \|^2_R \\
\text{subject to} & \quad x_{i+1} = Ax_i + Br_i \quad \forall i = 0, \ldots, N-1 \\
& \quad v_{min} \leq v_i \leq v_{max} \quad \forall i = 0, \ldots, N-1 \\
& \quad (x_N - x_r) \in T(x_r) 
\end{align*}
\]

where $x_r = [0 \ r_{N-1}]^T$. The constraint (6.7d) makes sure that the solution to the optimization problem will drive the system states to $T$ in $N$ steps.
Step 3: In step three we want to show that the cost function is decreasing when the system has reached the terminal set. It is convenient to introduce the notation:

\[
J_k = \sum_{i=0}^{N-1} \| r_{k+i} - r_{nom} \|^2_R.
\]  \hspace{1cm} (6.8)

For each \( x_0 \in \mathcal{X} \) there exists a reference \( \bar{r} \) such that \( r_i = \bar{r}, \forall i = 0, \ldots, N-1 \) is a feasible control sequence for formulation (6.7). The sequence \( \{ \bar{r}, \ldots, \bar{r} \} \) gives the suboptimal cost \( \bar{J} \).

Assume that we choose a series of these suboptimal sequences for \( k \to \infty \).

Now for the final stage we need to show that there exists one solution with a reference sequence which will decrease the cost function when converging to \( r_{nom} \). If such a solution exists we know that, because the MPC is optimal, the MPC will find solutions which decrease the cost at least as much as the proposed reference sequence, and \( r_{k+i} \) will converge to \( r_{nom} \).

Recall the control sequence \( \{ \bar{r}, \ldots, \bar{r} \} \). From this point we can shift the terminal set and for time \( k+1 \) add the control signal:

\[
r_{\gamma} = \gamma \bar{r} + (1-\gamma) r_{nom}, \quad \gamma \in [0,1)
\]  \hspace{1cm} (6.9)

which is a point on the line segment between \( \bar{r} \) and \( r_{nom} \). This gives the control sequence \( \{ \bar{r}, \ldots, \bar{r}, r_{\gamma} \} \). If we, by adding this control signal, get a lower total cost we can say that the cost function is decreasing when the system has entered the terminal set. The following calculations show that the cost is decreasing:

\[
\bar{J}_k - \bar{J}_{k+1} = \sum_{i=0}^{N-1} \| \bar{r} - r_{nom} \|^2_R - \sum_{i=0}^{N-2} (\| \bar{r} - r_{nom} \|^2_R - \| r_{\gamma} - r_{nom} \|^2_R)
\]

\[
= \| \bar{r} - r_{nom} \|^2_R - \| r_{\gamma} - r_{nom} \|^2_R
\]

\[
= \int \| r_{\gamma} - r_{nom} \|^2_R = (6.9)) = \| \gamma \bar{r} + (1-\gamma) r_{nom} - r_{nom} \|^2_R
\]

\[
= \gamma^2 \| \bar{r} - r_{nom} \|^2_R
\]

\[
= \| \bar{r} - r_{nom} \|^2_R - \gamma^2 \| \bar{r} - r_{nom} \|^2_R
\]

\[
= (1 - \gamma^2) \| \bar{r} - r_{nom} \|^2_R > 0.
\]  \hspace{1cm} (6.10)

This shows that \( \bar{J}_k - \bar{J}_{k+1} > 0 \), i.e. the cost function is decreasing and \( \lim_{k \to \infty} \bar{J}_k = 0 \).

Remember that \( J \geq 0 \), as can be seen from (6.8). We know that the MPC is optimal which means that the controller will find a control sequence with a lower (or possibly equal) cost \( J^*_k \). This in turn means that \( J^*_k \leq \bar{J}_k \forall k \). This shows that \( \lim_{k \to \infty} J^*_k = 0 \). From (6.8) we can conclude that \( \| r_i - r_{nom} \|^2_R \to 0 \), i.e. \( r_i \) converges to \( r_{nom} \).

Figure 6.24 provides an overview of the shifting terminal set \( T \) when applying the control signal \( r_{\gamma} \).
6.5 Stability

\[ r_{nom} \]

Figure 6.24: Illustration of the terminal set \( T \) and how it is shifted towards \( r_{nom} \) through the reference \( r_{\gamma} \).

6.5.1 Simulations with Terminal Set

It is interesting to investigate whether the additional constraint changes the behavior of the reference governor. First we run a simulation with a large step in \( r_{sp} \), but in absence of disturbances. Results from this simulation are presented in Figure 6.25. As we can see the behavior is not affected in the nominal case, which is intuitive because the constraints are never active and the terminal set is just being "moved".

It might seem unnecessary to run simulations with disturbances, since the proof only holds for undisturbed case. However, we find it interesting because the implementation issue is also considered in this thesis. And if the algorithm is going to be implemented such cases need to be taken care of as well. For smaller disturbances the behavior of the system, when the terminal set is added, is effectively unchanged. If we increase the bias disturbance and decrease the penalty on \( \epsilon_i \) we can trigger some differences. A simulation with height bias disturbance of 15 m and with \( \rho = 50 \) is presented in Figure 6.26. In these plots we can see that the terminal constraint results in a controller that holds the vertical velocity closer to its boundaries.
Figure 6.25: Simulation with a terminal set constraint. $N = 10$ and $\rho = 50$.

(a) With terminal set constraint. 

(b) Without terminal set constraint.

Figure 6.26: Simulation with height bias 15 m, with and without terminal set constraint. $N = 10$ and $\rho = 50$ for both simulations.
These results indicate that, for a practical implementation, the terminal set may be unnecessary since the simulations only show small differences with and without the terminal constraint. The addition of the terminal set can result in infeasible problems during disturbed cases, since it is formulated as a hard constraint. It will also result in a more complex MPC formulation which could increase the solve time of the optimization problem.
In this chapter we describe two advanced simulation environments used to evaluate our RG solutions and the external solvers studied for implementation in SKELDAR’s flight control system. First we present results from simulations in ARES, a simulation environment developed at SAAB. For evaluating the real time requirements of the RG when used in SKELDAR’s on-board computer we also simulate in a HIL simulation system.

We present two external solvers that have been used during this thesis. We also show how our problem formulations fit into their framework and give comments on performance and usability of the solvers. The solvers are used when implementing the controllers in ARES and the HIL simulation system. This implementation is also considered in this chapter.

7.1 ARES

For more sophisticated simulations of the helicopter and our proposed controllers we use SAAB’s own simulation environment called ARES. ARES includes modules simulating, e.g. the aircraft engine, rigid-body dynamics and aerodynamic forces acting on the fuselage. The dynamics are nonlinear and the total simulation system will include time delays etc. that is not included in the prediction model or in the model used for simulations in MATLAB/SIMULINK. At SAAB ARES is considered the true model, or at least the “best guess” and providing a working control system in ARES is an important step towards a system ready for implementation.

ARES also includes todays flight control system of SKELDAR, i.e. the control system implemented in the on-board computer of the helicopter. We have thus far in the thesis studied how to use MPC as add-on in an earlier developed control
system. To investigate the integration and implementation of MPC in the flight control system we implement our controllers, using C-code and the generated external solvers, into the flight control software. We have investigated both FORCES and CVXGEN but will present simulations using CVXGEN. See Section 7.3 for description of the solvers. Since we in ARES have the same flight control software as in the on-board computer the implementation of the reference manipulating MPC in the HIL simulation system will basically be the same (except for a few details discussed in Section 7.4.3).

### 7.1.1 RG with Pre-filtered Nominal Reference

The RG with pre-filtered nominal reference is implemented in ARES together with the pre-filter that generates the new nominal reference signal $r_{nom}$ from a given set point reference $r_{sp}$. Initial simulation with a step in $r_{sp}$ shows promising results, seen in Figure 7.1. The RG is, as expected, able to follow the nominal reference without using slack to violate constraints.

![Figure 7.1: ARES simulations for a step in set point reference, $N = 25, \rho = 500$.](image)

When we simulate a disturbance in ARES we get undesired behaviors. If the velocity is pushed outside its constraint limits the system starts to oscillate, as can be seen in Figure 7.2. The vertical velocity and collective pitch oscillate in an unacceptable manner. The height is still relatively smooth because of the slow dynamic of the system.

It is reasonable to assume that a time delay between the inner controller and
the vertical dynamics exists. Since the oscillations have not been observed in SIMULINK (where we have simulated without delays) our conjecture is that the oscillations is a consequence of some unmodeled time delay in the system. In order to investigate this we recreated the same behavior in SIMULINK. This is easily achieved by adding a time delay between the primary controller and the vertical system in the simulation model. By trial and error we concluded that adding a time delay of $4T_s$ was enough to induce the oscillations seen in the ARES simulations. Figure 7.3 shows a similar simulation as in Figure 7.2 and we can see the same kind of oscillations. One approach to deal with this kind of model errors could be to model the time delay and include it in the prediction model. However, there will always, to some extent, exist unknown time delays in a system we will study how to tune the RG in order to be less sensitive to this kind of model errors. As stated in Section 6.2.1, $N$ and $\rho$ are our tuning parameters. By considering the observation that an RG with a small value of $\rho$ gives smoother signals we can assume that such a RG will handle these kinds of errors better than an RG with larger $\rho$. This can be confirmed by studying Figure 7.3-7.5 where we show system behavior in SIMULINK-simulations including the time delay. In these figures we see that the oscillations are less distinct for setups with decreased $\rho$. It can also be seen that altering the prediction horizon $N$ does not affect the characteristics of the oscillations, it only affects the amplitudes slightly.

Next step is to transfer this knowledge to ARES and see if we can get rid of the oscillations by tuning the RG a little bit. Since the observation that $N$ is not as vital as $\rho$, we will keep $N = 25$ and decrease $\rho$ until the oscillations are gone.
Figure 7.3: System behavior under a height bias disturbance with a time delay between the inner controller and the vertical dynamics. Simulink-simulations with $\rho = 1000$ and different $N$.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure7.3}
\caption{(a) $N = 10$. (b) $N = 25$.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure7.4}
\caption{(a) $N = 10$. (b) $N = 25$.}
\end{figure}

Figure 7.4: System behavior under a height bias disturbance with a time delay between the inner controller and the vertical dynamics. Simulink-simulations with $\rho = 100$ and different $N$. 

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure7.4}
\caption{(a) $N = 10$. (b) $N = 25$.}
\end{figure}
Figure 7.5: System behavior under a height bias disturbance with a time delay between the inner controller and the vertical dynamics. SIMULINK-simulations with $\rho = 10$ and different $N$.

In Figure 7.6 we have studied a sequence of simulations which gives an indication how to tune the reference governor in order to handle unknown time delays in a desired way. As stated earlier, decreasing $\rho$ gives smoother control at the price of larger constraint violations. So here a trade-off between violations and control behavior regarding aggressiveness has to be considered. We can see that $\rho$ has to be decreased to 10 in order to get rid of all oscillations. But then the vertical velocity constraint is heavily violated, which might not be acceptable.
Figure 7.6: ARES simulations with height bias disturbance. Plots that show the behavior for a variety of slack penalties $\rho$ and $N = 25$. 

(a) $\rho = 100$. 

(b) $\rho = 50$. 

(c) $\rho = 30$. 

(d) $\rho = 10$. 

\[ \rho = 100 \] 
\[ \rho = 50 \] 
\[ \rho = 30 \] 
\[ \rho = 10 \]
7.1.2 RG with Slew Rate Constraints

For completeness we present some simulations in ARES (Fig. 7.7) using the slew rate MPC covered in Section 6.3. Figure 7.7a simulates a regular step corresponding to the SIMULINK simulation in Figure 6.17a. Note that for the ARES simulation we use $N = 20$ which gives less overshoot than for the SIMULINK simulation with $N = 10$. $\rho_2$ is also increased to get acceptable performance in ARES. Otherwise the simulation in ARES corresponds well to the simulation in SIMULINK. Figure 7.7b simulates the same step but also includes a disturbance. At $t = 32.2$ s a force, equivalent to 600 N, is applied at the helicopter’s COG, pushing the helicopter upwards which increases the ascending velocity above the velocity constraint. This disturbance corresponds to the external force disturbance where instead of applying a bias in the control signal $\delta_{col}$ an actual force acts on the aircrafts fuselage. In the $\delta_{col}$ signal plot in Figure 7.7b we can see that the control system finds a new trim position after the helicopter has stabilized.

Figure 7.7: ARES simulations with the slew rate MPC. Horizon $N = 20$, quadratic penalties for $\epsilon_1$ and $\epsilon_3$ with $\rho_1 = 1000$ and $\rho_3 = 10000$ and linear penalty for $\epsilon_2$ with $\rho_2 = 200$.

7.1.3 RG with Resampled Prediction Model

Recall the MPC formulation discussed in Section 6.2.1 where the prediction model is resampled with lower sampling rate. We have shown in SIMULINK that it is possible with this formulation to increase the length of which the MPC can predict. Using this formulation in ARES shows other interesting results, namely that the oscillations induced via a time delay in the system can be removed using a resampled prediction model. In Figure 7.8 we show two ARES simulations with different sampling rate. The left plot in the figure simulates prediction horizon $N = 100$ and sampling time $T_s = 0.02$ s and the right plot simulates $N = 10$ and $T_s = 0.2$ s, making both MPC controllers predict 2 seconds into the future.
For the resampled MPC the oscillations are gone. We could see in the Simulink simulations (Fig. 6.22) without the time delay that the resampled MPC was less aggressive. This could be one reason that we get less oscillations, considering the results from the above discussion were less aggressive RG, by smaller value of the penalty $\rho$, could decrease the oscillations. Note that for both the resampled MPC and the non-resampled MPC we have used $\rho = 100$. Another reason why the resampled MPC shows less, or no oscillations, could be that the time delay in the system is smaller than the sample time of the resampled prediction model, which means that the time delay might not effect the predictions in the internal model, and is only visible in the transient dynamics of system.

![Simulation plots](image)

**Figure 7.8:** ARES simulations comparing resampled MPC with sample time $T_s = 0.02s$ with MPC with original sample time $T_s = 0.02s$. $\rho = 100$ is used for both simulations.

### 7.2 HIL

The SKELDAR crew at SAAB has developed an avionics test rig capable of simulating flight tests using ARES in a hardware-in-the-loop system, making it possible to include real hardware, such as GPS units, the on-board computer with flight control system and servos controlling the rotors.

To implement the reference governors, using the external solvers presented in Section 7.3, in the on-board computer in the HIL simulation system we can use the same code used to implement the controllers in ARES. The only difference here is that we can only use floating point numbers. This is done as described in Section 7.4.3. Unfortunately simulations in the HIL system indicate that only very small MPC problems will meet the real time requirements of the hardware. Using CVXGEN’s solver, the addition of the prefiltered RG with prediction horizon $N = 1$ increased the CPU usage of the on-board computer from 50% to 70%. As the solver scale approximately linearly in solve time [Mattingley and Boyd, 2010] the maximum prediction horizon of the pre-filter RG would be only two or three.
7.3 External Solvers

There are two methods for online implementation of the MPC, explicit MPC and implicit MPC. Simply put, in explicit MPC the solution to the corresponding optimization problem of the MPC is calculated offline for all possible initial conditions $x_0$. The solutions are stored in some data structure and then used online using a lookup table or search tree. The topic of explicit MPC is covered in Kvasnica [2009]. In implicit MPC, which is the method treated in this thesis, the solution to the optimization problem is calculated online. Meaning that in each time step $k$ the controller must solve the corresponding QP-problem. For this to be possible it is important to have a fast online QP-solver. In this thesis we have considered two online solvers, FORCES and CVXGEN. Both solvers implement the primal-dual interior point method. The primal-dual interior point method solves the convex optimization problem using the KKT conditions. We will not go into detail in the solving of QP-problems using interior point methods. Instead we refer the reader to Nocedal and Wright [2006] for a complete derivation of the method.

7.3.1 FORCES

The first online solver considered is called FORCES (Fast Optimization for Real-time Control on Embedded Systems) [Domahidi, 2012]. This is a numerical optimization code generation framework for convex multistage problems developed by Automatic Control Laboratory at the Swiss Federal Institute of Technology in Zurich (ETHZ). FORCES generates a C-code solver for your specific optimization problem by defining it as a convex multistage problem in a specified form. The solver can then be used in an existing embedded software. The generated code also comes with a MATLAB MEX interface for easy evaluation of the solver.

The problem class supported by FORCES is specified in the following form:

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \sum_{i=1}^{N} z_i^T H_i z_i + f_i^T z_i \\
\text{subject to} & \quad z_{i,\text{min}} \leq z_i \leq z_{i,\text{max}}, \quad i = 1, \ldots, N \\
& \quad A_i z_i \leq b_i, \quad i = 1, \ldots, N \\
& \quad z_i^T Q_{i,j} z_j + g_{i,j} z_i \leq r_{i,j}^2, \quad i = 1, \ldots, N, \quad j = 1, \ldots, q_i \\
& \quad C_i z_i + D_{i+1} z_{i+1} = c_i, \quad i = 1, \ldots, N - 1 
\end{align*}
\]

with $H_i \succeq 0, Q_{i,j} > 0 \forall i,j$ and $z_i$ is the so-called stage variable, typically optimization variables. The constraints specified in (7.1b)–(7.1d) correspond to bounds, affine inequalities and quadratic inequalities respectively. The constraint in (7.1e) couples two stage variables and is used to specify the system dynamics.

Using FORCES we implemented the MPC formulated in (6.5). Expressed in the
convex multistage problem class we get:

\[
H_i = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & R & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix} \quad f_i = \begin{bmatrix}
0 \\
0 \\
-Rr_{nom}(k) \\
0 \\
0 \\
\rho \\
\end{bmatrix} \quad (7.2)
\]

\[
z_i = \begin{bmatrix}
x_1(k + i) \\
x_2(k + i) \\
r(k + i) \\
r(k + i - 1) \\
\Delta r(k + i - 1) \\
e(i) \\
\end{bmatrix} \quad z_{i,min} = \begin{bmatrix}
-1 \\
-1 \\
-1 \\
-1 \\
-1 \\
0 \\
\end{bmatrix} \quad z_{i,max} = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
\end{bmatrix} \quad (7.3)
\]

\[
A_i = \begin{bmatrix}
-1 & 0 & 0 & 0 & 0 & -1 \\
1 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & -1 & 1 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 \\
0 & 0 & -1 & 1 & 1 & 0 \\
0 & 0 & 1 & -1 & -1 & 0 \\
\end{bmatrix} \quad b_i = \begin{bmatrix}
x_{i,min} \\
x_{i,max} \\
-\Delta r_{max} \\
\Delta r_{max} \\
-\Delta r_{i,max}^{2} \\
\Delta r_{i,max}^{2} \\
\end{bmatrix} \quad (7.4)
\]

\[
C_i = \begin{bmatrix}
A & B & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
\end{bmatrix} \quad D_i = \begin{bmatrix}
-1 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 \\
\end{bmatrix} \quad c_i = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
\end{bmatrix} \quad (7.5)
\]

where index \(i\) has the same range for all matrices as described in formulation (7.1). Index \(k\) is the current time. Note that we get a special case for the first stage \(i = 1\) to include the initial conditions for the system states \((x_1(i), x_2(i))\) and augmented states \((r(i - 1), \Delta r(i - 1))\). To do this in FORCES we use the equality constraint (7.1e) and specify which stage variables we have conditions on in \(C_1\) and then the initial conditions goes in \(c_1\). \(D_2\) will get an extra row of zeros for each initial condition.

FORCES is a server based code generator that requires you to send your problem definition from a FORCES client to a server that generates your solver. To create your custom solver the optimization problem is first specified using the FORCES toolbox for MATLAB (this is also the FORCES client). Your problem definition is then sent from MATLAB to the FORCES server for code generation and is then returned right into MATLAB.

### 7.3.2 CVXGEN

The second online solver used in this thesis is called CVXGEN (Code Generation for Convex Optimization) and it is developed at Stanford University [Mattingley and Boyd, 2010]. CVXGEN is also a server based code generator for convex optimization with the difference that the problem is specified in a web client using CVXGENs own syntax for convex problem definition. CVXGEN generates C-code,
much like FORCES, and also provides an interface for use in MATLAB. There are four types of text blocks available in CVXGEN web client to define your optimization problem:

- **dimensions** – Used to specify the dimension of the problem. The dimensions can be used anywhere else in the problem definition.

- **parameter** – Placeholders for the problem data, may be used in the objective and constraints block. The parameter values are not given until the solver is called.

- **variables** – Defines the optimization variables whose values are found during solving. Can be used in the objective and constraints block.

- **objective** – Specifies the objective function of the optimization problem. The objective is either minimize or maximize and is followed by the subject to block for constraints definitions (optional).

The definitions in each block also have additional properties that may or may not be optional. Parameters can e.g. be defined non-negative using the specifier nonnegative after the parameter definition. Only certain functions are allowed in the objective and constraints block. For a full description of the CVXGEN syntax we refer to the CVXGEN website [Mattingley and Boyd, 2010].

To generate the solver for the complete MPC formulation described in Section 6.2 we use the CVXGEN web client and write as in Figure 7.9.

The solver is then generated and downloaded directly using the web client. CVXGEN creates the C-code for the generated solver and a MATLAB MEX interface for easy evaluation. An example of how to use the C-interface is also provided.

### 7.4 Comparing FORCES and CVXGEN

Here we will compare the two solvers in terms of usability, flexibility and performance. First we discuss the user-friendliness of the two solvers and compare their problem formulations. Then we present some results using the solvers and compare their performance followed by a discussion of the generated code and implementation of the solvers.

#### 7.4.1 Usability

Considering the problem definition of the two solvers CVXGEN has a much more user-friendly syntax as well as a less complicated, easy to understand problem specification. The optimization problem in Section 7.3.1 expressed using FORCES requires roughly 100 lines of MATLAB code, the same problem comes down to about 40 lines of code in CVXGEN syntax. Another positive aspect of the CVXGEN syntax is that it is much alike the syntax used to specify an optimization problem in YALMIP. Since we develop our controllers using YALMIP it is easy to translate the problem formulation to CVXGEN syntax.
Simulations

parameters
\begin{align*}
    & A \ (n, n) \\
    & B \ (n, m) \\
    & R \ (m, m) \ \text{psd nonnegative} \\
    & \rho \ \text{psd nonnegative} \\
    & x[0] \ (n) \\
    & r_{\text{nom}} \\
    & v_{\text{lim}} \ \text{nonnegative}
\end{align*}

variables
\begin{align*}
    & r[t] \ (m), \ t=0..N \\
    & x[t] \ (n), \ t=1..N+1 \\
    & \epsilon[t] \ \text{nonnegative}, \ t=0..N+1
\end{align*}

minimize
\begin{align*}
    \sum_{t=0..N} (\text{quad}(r[t]-r_{\text{nom}}, R) + \text{quad}(\epsilon[t], \rho))
\end{align*}

subject to
\begin{align*}
    & x[t+1] = A \times x[t] + B \times r[t], \ t=0..N \\
    & -v_{\text{lim}} - \epsilon[t] \leq x[t][1] \leq v_{\text{lim}} + \epsilon[t], \ t=0..N
\end{align*}

Figure 7.9: MPC-formulation in Section 6.2 expressed using CVXGEN syntax.

The CVXGEN web client also provides a mathematical representation of the optimization problem making it easy to check your implementation for formulation errors while translating to CVXGEN syntax. This is a very useful feature since it is not always obvious if you get the formulation that you expect. This weighs heavily to CVXGEN’s advantage since FORCES’s problem formulation is rather difficult to understand and provides no feedback of what optimization problem you actually get.

7.4.2 Performance

To compare the solvers performance we run the controllers described in Section 6.2 and 6.3 (referred to as MPC 1 and MPC 2 respectively) for three different prediction horizons. To get enough excitation of the solvers we use a simulation setup with an initial state corresponding to a disturbance making the velocity
constraint active, followed by a step in the height reference for simulating nominal conditions. We use different initial conditions in MPC 1 and MPC 2. Note that further details in how the solvers are implemented and an examination of why the solvers differ in performance are outside the scope of this thesis. We present the results gained from simulations in MATLAB and constrain the discussion to point out where the solvers differ. For details in the implementations of the solvers we refer to the papers [Domahidi et al., 2012] for FORCES and [Mattingley and Boyd, 2012] for CVXGEN.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$(n_v, n_c)$</th>
<th>FORCES</th>
<th></th>
<th>CODE SIZE</th>
<th>SOLVE TIME</th>
<th></th>
<th>CODE SIZE</th>
<th>SOLVE TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPC 1</td>
<td>10</td>
<td>(28, 40)</td>
<td>220 kB</td>
<td>0.25 ms</td>
<td>233 kB</td>
<td>0.50 ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPC 1</td>
<td>50</td>
<td>(148, 200)</td>
<td>773 kB</td>
<td>11 ms</td>
<td>1155 kB</td>
<td>32 ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPC 1</td>
<td>100</td>
<td>(298, 400)</td>
<td>1523 kB</td>
<td>24 ms</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPC 2</td>
<td>10</td>
<td>(28, 80)</td>
<td>149 kB</td>
<td>0.11 ms</td>
<td>132 kB</td>
<td>0.35 ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPC 2</td>
<td>50</td>
<td>(148, 400)</td>
<td>479 kB</td>
<td>0.42 ms</td>
<td>609 kB</td>
<td>16 ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPC 2</td>
<td>100</td>
<td>(298, 800)</td>
<td>938 kB</td>
<td>0.67 ms</td>
<td>1224 kB</td>
<td>33 ms</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.1: Simulations in MATLAB showing differences in code size and solve time. Simulations running on a 2.26 GHz Intel Core 2 Duo processor. $n_v =$ number of optimization variables, $n_c =$ number of constraints. Note that we define the number of variables and constraints from the mathematical problem definitions (6.4) and (6.5). These numbers are different for FORCES and CVXGEN because the problem is formulated differently.

In Table 7.1 we clearly see that the solve time using FORCES is less than CVXGEN, especially when comparing the results of MPC 2 where the difference is significantly large. The code size of FORCES is slightly smaller than CVXGEN, except for MPC 2 with $N = 10$ where CVXGEN is a few kilobytes smaller. CVXGEN was unable to generate the solver for MPC 1 with $N = 100$ because the problem became too large. According to the developers of CVXGEN the solver does not work well for larger problems and they limit their solver to smaller problems. CVXGEN provides the KKT-matrix [Mattingley and Boyd, 2012] when formulating the problem. Recommended maximum size of the KKT-matrix is 4000 non-zero entries and for the formulation when the generation failed the KKT-matrix had about 5800 non-zero entries. Note that the solve time for MPC 1 is significantly longer than MPC 2. In MPC 1 the number of constraints are twice as many as for MPC 2 and these constraints will almost always be active, even in nominal conditions with a regular step in the set-point reference.

### 7.4.3 Implementation

Both solvers implement a primal-dual interior point method. Both applications generate library free C-code which is specially attractive for embedded systems where the adding of new libraries might be limited. In our case this was attractive for easy implementation in the SKELDAR software. The code size proved to
be an important aspect when implementing the solver in SKELDAR’s on-board computer in the HIL simulation system. For quick and easy upload of new code for HIL simulations the code is loaded in the RAM memory of the on-board computer. Here FORCES was to prefer since it is slightly smaller in code size than CVXGEN.

With FORCES it is possible to have several solvers running in the same application because each solver can be generated with unique file-, function- and data member names, creating no naming violations. CVXGEN uses the same names for each solver generated, making it difficult to use several solvers.

FORCES provides the option of choosing what data type to use in the solver, either float or double. The SKELDAR software uses single-precision which restricts the data types to floats when we generate solvers to run in the HIL simulation system. CVXGEN does not have data type as an option but according to the paper by Mattingley [2011] the solver can be used with single-precision as well. Although FORCES has the data type as an option the solver is not yet guaranteed to work with single-precision. The expected accuracy when using single-precision will naturally be less than for double-precision and for most cases FORCES will work together with floats by tuning the accuracy conditions. For the MPC formulation in (6.5) we never managed to tune FORCES to work with single-precision. By manually replacing the data types from double to float in the generated solver from CVXGEN and by decreasing the accuracy of the solutions CVXGEN was able to solve the MPC in (6.5) with single-precision.
Conclusions and Future Work

From the different RG formulations suggested in this thesis we believe that the pre-filter RG shows the most potential in terms of complexity, applicability and ease of use. Through the separation of the reference rate constraint handling via the pre-filter, the corresponding MPC formulation becomes relatively simple and easy to tune. A small MPC formulation will result in a smaller optimization problem, which is desired for real time applications. The formulation can also, as shown, be extended to guarantee stability. On the other hand, simulations show that the stability extension might be omitted in practical implementations, as the addition only show small differences in the closed loop behavior. As further prospects the pre-filter RG requires additional investigation regarding robustness against modeling errors in the prediction model. As we have observed in ARES simulations, a possible time delay can cause undesired oscillations. Although, in the thesis, we consider the downsampled RG as a separate formulation, the only difference is the sampling time of prediction model. This RG seems to be, from results in simulations, more robust against time delays. The method of down sampling of the prediction model has not been observed in the literature and we can simply present interesting results when applied to our particular problem.

The conclusion regarding the RG with slew rate constraints included in the MPC formulation is that, although this formulation show promising results with respect to both nominal and disturbed cases, considering the relatively small control problem we have studied, this RG formulation becomes too complex for practical use. In the form that we have presented the RG, the tuning will depend on the initial states and the behavior of the total closed loop system will be different under different conditions, e.g. when the target reference is small or large. Still, as further development of the slew rate RG, we investigated some scaling of the
penalties in the cost function to reach a formulation where the RG’s behavior becomes independent of the state. As an initial test we added the functionality to scale the rate penalties by the set-point value. For the formulation with linear rate penalties, this resulted in the same behavior for the nominal case for all set-points but results in very different behaviors for the disturbed cases. This, again, indicates that the formulation requires further development, which most likely will increase the complexity of the optimization problem.

We find that a major aspect for practical use of MPC is the solver applied for the corresponding optimization problem. It seems difficult to simply pick a generator to create a solver for a general optimization problem and expect acceptable performance. The two solvers studied in this thesis have shown numerical problems, restrictions regarding problem size and real time limitations. This motivates the need for more personalized solvers, tailored for your specific problem and requirements. There is one, widely investigated, alternative method for online implementation which we have not considered in the thesis. Namely explicit solving of the optimization problem used in explicit MPC. This could be a solution to the real time requirement issue.

As we have discussed, a few obstacles remain, such as robustness and online solving, before this setup is mature enough for implementation in an embedded system. However, we still see potential in the use of Model Predictive Control as a method for flight envelope protection in the form of a Reference Governor.
Appendix
Mathematical Definitions

Definition A.1-A.5 regarding sets and functions can also be found in R.Walter [1976]. The definitions of a positive invariant set and a maximal output admissible set A.6-A.7 can be found in Gilbert and Tan [1991]. All sets in this appendix are subsets of \( \mathbb{R}^n \).

A.1 Definition (Convex Function). A function \( f(x) \) is convex if \( \forall x_1, x_2 \in \mathbb{R}^n \) it holds that:

\[
f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2)
\]

where \( \lambda \in [0, 1] \).

A.2 Definition (Convex Set). A set \( X \) is said to be convex if \( \forall x_1, x_2 \in X \) it holds that:

\[
x = \lambda x_1 + (1 - \lambda)x_2 \in X
\]

where \( \lambda \in [0, 1] \).

A.3 Definition (Closed Set). A set \( X \) is closed if all limit points of \( X \) belongs to \( X \).

A.4 Definition (Bounded Set). A set \( X \) is bounded if there is a point \( q \in \mathbb{X}^n \) such that:

\[
d(p, q) < \infty, \forall p \in X.
\]
A.5 Definition (Compact Set). A set $X$ is said to be compact if it is bounded and closed.

A.6 Definition (Positively Invariant Set). A set $X$ is said to be positively invariant for a system

$$x_{k+1} = f(x_k)$$

if $\forall x_k \in X$ it yields that $x_{k+i} \in X$ for $i > 0$.

A.7 Definition (Maximal Output Admissible Set). The maximal output admissible set for a system is defined by:

$$O_\infty(A, C, Y) = \{x \in X^n : CA^k x \in Y\}$$

where $A$ and $C$ is system matrices and $Y$ from $y_k = Cx_k \in Y$ is the constraints that has to be fulfilled in all time instances.


A.H. Glattfelder and W. Schaufelberger. A path from antiwindup to override


Upphovsrätt

Detta dokument hålls tillgängligt på Internet — eller dess framtida ersättare — under 25 år 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 icke-kommersiell 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 nämnd 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.

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

Copyright

The publishers will keep this document online on the Internet — or its possible replacement — for a period of 25 years 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 his/her 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/

© Victor Carlsson & Oskar Sunesson