Linköping studies in science and technology. Dissertations.
No. 1881

# Fighter Aircraft Maneuver Limiting Using MPC

## Theory and Application

**Daniel Simon**

LINKÖPING UNIVERSITY

**Cover illustration:** An artistic interpretation of a Gripen fighter aircraft performing a high angle of attack maneuver. The different steps in the time lapse represents past, present and future predictions of the maneuver.

Linköping studies in science and technology. Dissertations.
No. 1881

**Fighter Aircraft Maneuver Limiting Using MPCTheory and Application**

Daniel Simon

*dansi@isy.liu.se*
*www.control.isy.liu.se*
*Division of Automatic Control*
*Department of Electrical Engineering*
*Linköping University*
*SE–581 83 Linköping*
*Sweden*

ISBN 978-91-7685-450-1 ISSN 0345-7524



Printed by LiU-Tryck, Linköping, Sweden 2017

*To Juni!*

# Abstract

Flight control design for modern fighter aircraft is a challenging task. Aircraft are dynamical systems, which naturally contain a variety of constraints and nonlinearities such as, e.g., maximum permissible load factor, angle of attack and control surface deflections. Taking these limitations into account in the design of control systems is becoming increasingly important as the performance and complexity of the aircraft is constantly increasing.

The aeronautical industry has traditionally applied feedforward, anti-windup or similar techniques and different ad hoc engineering solutions to handle constraints on the aircraft. However these approaches often rely on engineering experience and insight rather than a theoretical foundation, and can often require a tremendous amount of time to tune.

In this thesis we investigate model predictive control as an alternative design tool to handle the constraints that arises in the flight control design.

We derive a simple reference tracking MPC algorithm for linear systems that build on the dual mode formulation with guaranteed stability and low complexity suitable for implementation in real time safety critical systems.

To reduce the computational burden of nonlinear model predictive control we propose a method to handle the nonlinear constraints, using a set of dynamically generated local inner polytopic approximations. The main benefit of the proposed method is that while computationally cheap it still can guarantee recursive feasibility and convergence.

An alternative to deriving MPC algorithms with guaranteed stability properties is to analyze the closed loop stability, post design. Here we focus on deriving a tool based on Mixed Integer Linear Programming for analysis of the closed loop stability and robust stability of linear systems controlled with MPC controllers.

To test the performance of model predictive control for a real world example we design and implement a standard MPC controller in the development simulator for the JAS 39 Gripen aircraft at Saab Aeronautics. This part of the thesis focuses on practical and tuning aspects of designing MPC controllers for fighter aircraft. Finally we have compared the MPC design with an alternative approach to maneuver limiting using a command governor.

# Populärvetenskaplig sammanfattning

Styrsystem i moderna flygplan ska sätta stopp om piloten gör en hastig manöver som äventyrar säkerheten. Speciellt viktigt är det i jaktflygplan där piloten kan tvingas manövrera flygplanet precis på gränsen för vad konstruktionen klarar. Reglermetoder från processindustrin anpassas nu för att passa flyget.

En av de viktigaste faktorerna när man konstruerar ett flygplan är att flygplanet ska vara enkelt och säkert att flyga. Därför är det av yttersta vikt att man konstruerar flygplanens styrsystem så att piloten inte kan sätta flygplanet i en sådan situation att säkerheten äventyras. En sådan situation kan vara att piloten styr flygplanet så att det tappar sin lyftkraft och därmed sin förmåga att flyga eller också att kraftig turbulens utsätter konstruktionen för allt för stora påfrestningar.

Speciellt viktigt är detta även i designen av styrsystem för jaktflygplan. Dessa kräver maximal manöverförmåga för att kunna ha övertaget i en luftstrid eller då de måste utmanövrera en fientlig missil. Piloterna manövrerar flygplanen väldigt nära gränsen för vad farkosten klarar av och ett automatiskt skydd, så kallat manöverskydd, mot att hamna i en riskfylld situation är en nödvändighet.

I min forskning studerar jag hur man med hjälp av intelligenta datoralgoritmer i flygplanens styrsystem ska kunna förebygga och hindra att flygplanet överskrider de begränsningar som finns i konstruktionen och hamnar i ett riskfyllt tillstånd.

Ett sätt som man kan göra detta är med så kallad prediktionsreglering, vilket har använts framgångsrikt inom till exempel processindustrin. Prediktionsreglering innebär i praktiken att datorn försöker förutspå (prediktera) flygplanets framtida rörelser och utifrån det finna de bästa styrkommandona så att inga begränsningar överskrids samtidigt som pilotens önskemål i största möjliga utsträckning följs. Detta görs genom att man formulerar ett matematiskt optimeringsproblem där man vill minimera skillnaden mellan pilotens önskemål och prediktionen av flygplanets framtida beteende. Bivillkor till detta optimeringsproblem är då flygplanets dynamik och alla de begränsningar som kan finnas i systemet. Detta optimeringsproblem löses sedan av flygplanets styrdator så fort nya mätdata är tillgängliga, det vill säga många gånger varje sekund. Dessa optimeringsproblem är komplicerade och kräver mycket beräkningskraft. En stor utmaning är därför att göra dessa enklare och mer anpassade för flygindustrin.

I min forskning fokuserar jag på de teoretiska egenskaperna hos de optimeringsproblem som prediktionsregleringen ger upphov till. Jag försöker anpassa dessa optimeringsproblem så att de ska vara relativt lätta att lösa samtidigt som de ska hantera de specifika problem som finns i flygindustrin.

# Acknowledgments

This is it, my thesis. This is the result of six years and four months of hard work or equivalently fourteen hundred days or, if you prefer, 11200 hours. This means that it has taken, on average, 233 days to write every paper or 68 hours for every page written in the thesis. Although this might seem as a very long time it would have taken much longer time if I wouldn't have had the privilege of working with some of the most talented and inspiring persons in the world.

Three of these people are my supervisors, associate professor Johan Löfberg, professor emeritus Torkel Glad and Dr. Ola Härkegård. Your knowledge, wisdom and patience show no boundaries and without your careful and insightful guidance this thesis wouldn't exist at all. Ola who is my industry supervisor also deserves my deepest gratitude for having faith in me and encouraging me to pursue a PhD.

I would also like to gratefully acknowledge the financial support from VINNOVA and the NFFP program. Without this support the thesis would never have become a reality.

My colleague and roommate Lic. Roger Larsson deserves a special acknowledgement for all the discussions, especially the ones concerning the dynamics of flight and aerodynamics. Also my friend and former colleague Dr. Daniel Peterson deserves a special acknowledgement for always having time for my questions. Dr. Johan Dahlin, associate professor Gustaf Hendeby, Dr. Daniel Peterson and Dr. Christian Lyzell should all be acknowledged for helping me with various LaTeX and TikZ issues. This really improved the quality of the thesis.

I have also received a great deal of help from my colleagues at Saab, especially Erik Backlund who have helped me setting up all the simulation models and helping me getting my research to run in Saab's simulator. Other colleagues such as Lic. Peter Rosander, Robert Hillgren, Tommy Persson, Jonas Lövgren and Bengt-Göran "Be-Ge" Sundqvist, to mention a few, is acknowledged for all the great discussions on flight control design which has put a more industrial flavor to the thesis.

Even though some people may say that I have worked a lot it hasn't been only work but a great deal of fun as well. The automatic control group at Linköping University is a great group of people and we have had much fun over the years. I have really enjoyed the coffee room discussions and all the great conference trips. Dr. Patrik Axelsson, Dr. Daniel Peterson, Dr. Zoran Sjanic, Lic. Ylva Ljung, Dr. Isak Nielsen, Dr. Sina Kohshfetrat Pakazad, Oskar Ljungqvist, Gustaf Lindmark, Dr. Jonas Linder, Dr. Niklas Wahlström among others have really made this a remarkable journey. The great atmosphere in the group is thanks to the head of the department professor Svante Gunnarsson and before him professor emeritus Lennart Ljung.

Finally I would like to thank my family and friends for the support and understanding I got over the years. Finally I would like to take the opportunity to thank my parents for always believing in me and encourage me to do the best I can.

And, to Anna...

$$x^2 + \left(y - \sqrt[3]{x^2}\right)^2 = 1$$

*Linköping, August 2017*
*Daniel Simon*

# Contents

# Notation

## Mathematics

| Notation | Meaning |
|---|---|
| $x^T$, $A^T$ | Vector or matrix transpose |
| $A^{-1}$ | Matrix inverse |
| $A \succeq 0$ | Positive semidefinite matrix |
| $x \geq 0$ | Elementwise inequality |
| $I$ | Identity matrix |
| $\mathbb{I}$ | Vertically concatenated identity matrices<br>$\mathbb{I} = [I,\ I,\ I,\ \ldots, I]^T$ |
| $\mathbb{1}$ | Vector of ones |
| $\mathbb{R}^n$ | Space of real vectors of dimension $n$ |
| $\|\|x\|\|$ | General (arbitrary) norm of a vector |
| $\|\|x\|\|_2$ | Euclidean norm of a vector |
| $\|\|x\|\|_\infty$ | Infinity norm of a vector |
| $\|\|x\|\|_D$ | Dual norm |
| $\|\|x\|\|_Q^2$ | Weighted quadratic function, $x^T Q x$ |
| $\mathcal{P}_1 \oplus \mathcal{P}_2$ | Minkovsky sum of two sets |
| $\mathcal{P}_1 \ominus \mathcal{P}_2$ | Pontryagin difference of two sets |
| $\mathcal{P}_1 \times \mathcal{P}_2$ | Carthesian product of two sets |
| $conv(\cdot)$ | Convex hull |
| $\text{int}(\mathcal{P})$ | Interior of the set $\mathcal{P}$ |
| $\text{int}_\epsilon(\mathcal{P})$ | The epsilon-interior of the set $\mathcal{P}$ |
| $\text{dom}(f)$ | Domain of a function $f(x)$ |

**Mathematics cont.**

| Notation | Meaning |
|---|---|
| $f_0(x)$ | Optimization problem objective function |
| $f_0^*$ | Optimal value of objective function |
| $x^*$ | Optimal value on optimization variable |
| $f_i(x)$ | Inequality constraint functions |
| $g_i(x)$ | Equality constraint functions |
| $\inf f(x)$ | Infimum of $f(x)$ |
| $\nabla f(x)$ | Gradient of a function |
| $\nabla^2 f(x)$ | Hessian of a function (second derivative) |

**MPC**

| Notation | Meaning |
|---|---|
| $\mathcal{X}$ | State constraint set |
| $\mathcal{U}$ | Input constraint set |
| $\mathcal{T}$ | Terminal state constraint set |
| $\Pi$ | Nonconvex input constraint set |
| $\mathcal{G}$ | Global polytopic inner approximation of $\Pi$ |
| $\mathcal{I}_i^k$ | Local polytopic inner approximation of $\Pi$, $i$ time steps into the future at time $k$ |
| $\mathcal{C}_k$ | Outer polytopic approximation of $\Pi$ |
| $V_k$ | Objective function value at time $k$ |
| $\ell(\cdot)$ | Stage cost |
| $\Psi(\cdot)$ | Terminal state cost |
| $\phi(\cdot)$ | Pseudo reference variable penalty function |
| $\kappa(x)$ | Optimal control law |
| $r$ | Reference input |
| $\bar{r}_k$ | Pseudo reference variable |
| $\bar{x}_k$ | Pseudo steady state |
| $\bar{u}_k$ | Pseudo steady state control |
| $\varepsilon$ | Slack variable |
| $\{x_i\}_{i=0}^N$ | A sequence of variables $x_i$ from $i = 0, \ldots, N$. I.e., $\{x_i\}_{i=0}^N = \{x_0, x_1, \ldots, x_N\}$ |
| $U_k$ | The vector of control signals $U_k = [u_k, u_{k+1}, \ldots, u_{k+N-1}]$ |
| $\lambda_k$ | Terminal state constraint set scaling variable |
| $N$ | Prediction horizon |
| $N_l$ | Prediction horizon for local polytope approximations, $\mathcal{I}_i^k$ |
| $N_{\mathcal{X}}$ | Number of state space partitions for explicit MPC |

### Aeronautical variables

| Notation | Meaning |
|---|---|
| $\alpha$ | Angle of attack |
| $\beta$ | Sideslip angle |
| $q$ | Angular pitch rate |
| $p$ | Roll rate |
| $r$ | Yaw rate |
| $\theta$ | Pitch angle, i.e., angle between x-axis and horizontal plane |
| $\delta_e$ | Aircraft trailing edge (elevons) control surface angular deflection |
| $\delta_c$ | Aircraft canard wing control surface angular deflection |
| $\delta_s$ | Helicopter swash plate pitch angle |
| $u$ | Velocity vector x-component |
| $v$ | Velocity vector y-component |
| $w$ | Velocity vector z-component |
| $V$ | Velocity vector $V = [u\ v\ w]$ |
| $\bar{V}$ | Speed, i.e., $\bar{V} = \sqrt{u^2 + v^2 + w^2}$ |
| $Q$ | Dynamic pressure, $Q = \frac{1}{2}\rho\bar{V}^2$ |
| $S$ | Wing surface area |
| $C_{(\cdot)}$ | Aerodynamic coefficients |
| $\bar{c}$ | Mean cord |
| $a$ | Helicopter rotor disc pitch angle |
| $c$ | Helicopter stabilizer disc pitch angle |

### Abbreviations

| Abbreviation | Meaning |
|---|---|
| ARES | Aircraft Rigid-Body Engineering System |
| BOT | Bleed off turn |
| KKT | Karush-Kuhn-Tucker |
| LQ | Linear Quadratic |
| LMI | Linear Matrix Inequality |
| MPC | Model Predictive Control |
| MILP | Mixed Integer Linear Programming |
| NMPC | Nonlinear Model Predictive Control |
| QP | Quadratic Program |
| SDP | SemiDefinite Programming |
| SQP | Sequential Quadratic Programming |

# 1 Introduction

## 1.1 Background and research motivation

Modern military aircraft offer a challenging control task since they operate over a wide range of conditions and are required to perform at their best in all conditions. Agile fighter aircraft require maximum control performance in order to have the upper hand in a dogfight or when they have to outmaneuver an enemy missile. Therefore pilots must be able to maneuver the aircraft very close to the limit of what it is capable of while at the same time focus on the tactical tasks of the mission. To enable this in open loop unstable aircraft, modern flight control systems need to have automatic systems for angle of attack and load factor limiting, so called maneuver load limits (MLL).

When surveying the available literature, such as, e.g., Nato [2000], one can draw the conclusion that for the design of these systems the aeronautical industry have traditionally used feedforward or anti-windup-similar techniques and different ad-hoc engineering solutions. The main drawback with these ad-hoc methods is that they usually lack any theoretical foundation and instead they rely on engineering experience and insight, and also that they require a tremendous amount of time to tune. However the increase in computational power over the last decade has opened up for more advanced control techniques to be used to systematically take constraints into account in the design.

One of the modern control system design techniques that has gained significant popularity in the aircraft industry is *Linear Quadratic control* (LQ), e.g., the Swedish fighter aircraft *JAS 39 Gripen* uses a gain scheduled LQ controller for its primary stabilizing controller. The LQ design technique is based on minimizing an objective which is quadratic in the states, $x(t)$, and the

controls, $u(t)$.

$$\underset{u(t)}{\text{minimize}} \quad \int_0^\infty x(t)^T Q x(t) + u(t)^T R u(t)\, dt \tag{1.1}$$

with (as the name indicate) linear system dynamics, i.e., the state evolution is described by a linear differential equation.

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{1.2}$$

Kalman [1960] showed that this problem can be solved explicitly and the optimal control law, $\kappa(x)$, is a linear state feedback.

$$u(t) = \kappa(x) = -Kx(t)$$

However, adding constraints on states and controls to the LQ problem formulation (1.1) and (1.2) results in a nonlinear optimal control problem

$$\underset{u(t)}{\text{minimize}} \quad \int_0^\infty x(t)^T Q x(t) + u(t)^T R u(t)\, dt \tag{1.3a}$$

subj. to

$$\dot{x}(t) = Ax + Bu \tag{1.3b}$$

$$x(t) \in \mathcal{X} \tag{1.3c}$$

$$u(t) \in \mathcal{U} \tag{1.3d}$$

which, in contrast to the LQ problem, can be extremely difficult to solve explicitly. An open loop solution can be obtained using the *Pontryagin Maximum Principle* (PMP) but to obtain an explicit closed loop feedback solution, i.e., $u(t) = \kappa(x)$, one has to solve the partial differential equation known as the *Hamilton-Jacobi-Bellman equation*. Unfortunately this is for almost all practical cases impossible to solve analytically [Maciejowski, 2002].

To overcome this, other application areas, such as the process industry, most frequently use real-time optimization-based methods to approximate the nonlinear optimal control problem. One of the more popular optimization-based methods is *Model Predictive Control* (MPC), in which the nonlinear optimal control problem is approximated with a finite dimensional optimization problem that is solved repeatedly on line in each iteration of the control system.

The objective of this thesis is to investigate model predictive control for the aeronautical industry with focus on the maneuver limiting tasks. The objective is to study both theoretical and application aspects to determine the suitability of MPC for flight control.

## 1.2 Previous research

Model predictive control is one of the most popular modern control techniques and there has been extensive research in the last 20 years. The theoretical development has been tremendous and as the computational power has grown the application of model predictive control has spread from the traditional process industry applications to faster more time critical applications typically found in the automotive and the aeronautical industries.

Within the aeronautical industry there has been an increasing interest in model predictive control during the last decade but early publications that apply model predictive control to the flight control problem can be traced back to the 1990's in papers such as e.g., Ebdon [1996], Shearer and Heise [1998].

One attempt to make an overview of the vast amount of literature on applications of MPC to the aeronautical industry has been done in the recent paper by Eren et al. [2017]. They review previous research from the aspects of system modelling, control problem formulation and controller structure, safety related issues and implementation aspects. For an in depth review of the area the reader is referred to this paper and we will not recapitulate the content here. Instead we will focus on some different types of applications of model predictive control in the aeronautical industry.

There are many different types of applications of model predictive control to the flight control problem that has been studied over the last decades. The general inner loop design has been studied by, e.g., Bhattacharya et al. [2002], Shearer and Heise [1998], Steinberg [1999] and Keviczky and Balas [2006a], while outer loop and guidance applications has been studied in papers by Fukushima et al. [2006], Keviczky and Balas [2005, 2006b], Petersen et al. [2013], Yang et al. [2009] and Gryte and Fossen [2016].

One of the greatest interest has been the application of MPC to design reconfigurable flight control laws that can adapt to actuator failures or battle damages. This has been studied in, e.g., de Almeida and Leissling [2009], Ebdon [1996], Hartley [2015], Kale and Chipperfield [2005], Kufoalor and Johansen [2013], Siddiqui [2010] and Maciejowski and Jones [2003]. For example in Maciejowski and Jones [2003] the authors claim that the fatal crash of the El Al Flight 1862 [NLR, 1992] could have been avoided if a fault tolerant MPC controller had been used.

Other applications are, e.g., structural load protection [Giesseler et al., 2012], and aeroelastic aircraft [Wang et al., 2016].

Studies that have focused more on the task of envelope protection and maneuver limiting control design are Falkena et al. [2011], Gros et al. [2012] and Hartley and Maciejowski [2015].

Model predictive control applied to fighter aircraft has been investigated in, e.g., Ebdon [1996], Kale and Chipperfield [2005], Keviczky and Balas [2006a], Shearer and Heise [1998] and Kufoalor and Johansen [2013]. Ebdon [1996]

investigate model predictive control for fault tolerant control of an F-18 aircraft. Kale and Chipperfield [2005] and Kufoalor and Johansen [2013] also investigate MPC for fault tolerant and reconfigurable control laws. In Kufoalor and Johansen [2013] it is applied to an F-16 aircraft and Kale and Chipperfield [2005] implement the MPC controller on the ADMIRE model [Forssell and Nilsson, 2005]. Both Shearer and Heise [1998] and Keviczky and Balas [2006a] apply model predictive control to the nonlinear dynamics of an F-16 aircraft. In the paper by Keviczky and Balas [2006a] the authors compare linear MPC control to a gain scheduled MPC and a full nonlinear MPC controller. Their conclusion is that gain scheduling is necessary to have sufficient performance over the whole nonlinear dynamics, compared to a simpler linear MPC controller. The gain scheduled controller is also more computational tractable and perform sufficiently well compared to the full nonlinear MPC controller.

## 1.3 Publications and main contributions

The contributions in this thesis can be divided into two main parts, theoretical development of algorithms and the practical aspects of implementation of MPC for realistic aircraft control problems.

The main theoretical contributions of this thesis are on different aspects of stability of model predictive controllers.

Due to the iterative nature of MPC one must take special measures to ensure that the optimization problem remains feasible and that the controller stabilises the system. However, these measures can in severe cases limit the reference tracking ability or result in a complex algorithm. In the conference paper

> Daniel Simon, Johan Löfberg, and Torkel Glad. Reference tracking MPC using terminal set scaling. In *51st IEEE Conference on Decision and Control (CDC)*, pages 4543–4548, dec 2012.

we extended the standard *dual mode* MPC formulation to a simple reference tracking algorithm and studied the stability properties of the proposed algorithm. The main theoretical contribution of this paper was to develop simplified stabilising constraints compared to existing methods for reference tracking in linear MPC by making small adjustments to the existing stabilizing constraints of the dual mode formulation.

However the proposed MPC algorithm that was derived in the above conference paper suffered some major drawbacks. It required the complete enumeration of all vertices in a, possibly complicated, polytopic set. Since the computational burden of this task, in the worst case, grows exponentially with the state dimension it was desired to reformulate the algorithm such that the vertex enumeration was avoided. In the journal paper

> Daniel Simon, Johan Löfberg, and Torkel Glad. Reference Tracking

MPC using Dynamic Terminal Set Transformation. *IEEE Transactions on Automatic Control*, 59(10):2790–2795, 2014.

we derived a dual formulation of the constraints involving vertex enumerations. This reformulation greatly reduced the worst case complexity of the controller making it suitable for implementation. An example from the aircraft industry showed that the proposed controller has the potential to be far less complex than existing state of the art algorithms without losing any performance.

In the conference paper

Daniel Simon, Johan Löfberg, and Torkel Glad. Nonlinear Model Predictive Control using Feedback Linearization and Local Inner Convex Constraint Approximations. In *Proceedings of the 2013 European Control Conference*, pages 2056–2061, 2013.

we considered the task of controlling a constrained nonlinear system with a combination of feedback linearization and linear MPC. This approach in general leads to an optimization problem with nonlinear and state dependent constraints on the control signal. The main contribution in this paper is that we replace the nonlinear control signal constraints with a set of convex approximations. The proposed algorithm results in an easy solvable convex optimization problem for which we can guarantee recursive feasibility and convergence. An example from the aircraft industry shows that the performance loss compared to using a global nonlinear branch and bound algorithm can be very small.

In principal all proofs of stability for model predictive controllers follow the same basic idea of showing recursive feasibility and convergence using the objective function as a Lyapunov function candidate. This approach works only in the most basic cases and more advanced formulations are often adapted so they can be proven stable with the basic approach. To overcome this limitation we derive an algorithm for post design stability analysis of linear model predictive controllers in the following conference paper

Daniel Simon and Johan Löfberg. Stability analysis of Model Predictive Controllers using Mixed Integer Linear Programming. In *IEEE 55th Conference on Decision and Control*, pages 7270–7275, Las Vegas, 2016.

The chapter in which this is discussed also contains a section of previously unpublished material. In this section we extend the above stability test to a test for robust stability for systems subject to additive disturbances.

The first main application-oriented contribution of this thesis can be found in the journal paper

Daniel Simon, Ola Härkegård, and Johan Löfberg. Command Governor Approach to Maneuver Limiting in Fighter Aircraft. *Journal of Guidance, Control, and Dynamics*, 40(6):1514–1527, 2017b.

which also has been presented as the conference paper

Daniel Simon, Ola Härkegård, and Johan Löfberg. Angle of Attack and Load Factor Limiting in Fighter Aircraft using Command Governors. In *AIAA Guidance, Navigation, and Control Conference, AIAA SciTech Forum*, 2017a.

Here we investigate a different approach to maneuver limiting of a fighter aircraft. We implemented and evaluated a command governor approach in Saab's simulation environment for the JAS 39 Gripen aircraft on a similar nonclassified fighter aircraft model. The command governor was augmented to the basic LQ controller already in place in the simulation environment.

The second application oriented contribution of this thesis is previously unpublished material. Here we have implemented an MPC controller for the same nonclassified aircraft model in Saab's simulation environment. The objective of that work was to review the practical implementation aspects and tuning strategies of real world flight control design problems in order to evaluate the applicability of MPC within the aeronautical industry as well as to give a proof of concept of MPC as a design technique for advanced flight control systems. The work focus on the practical aspects of model predictive controllers for flight control design such as structure of the controller, tuning and implementation.

## 1.4 Thesis outline

This thesis is divided into two main parts, Theory and Application. The theory part covers chapters 2 – 7 and the application part covers Chapter 8 and Chapter 9.

The necessary background material is covered in the chapters 2 – 4 while the main contributions are found in chapters 5 – 9.

The content of the chapters are as follows.

In Chapter 2 we outline the necessary mathematical background for the remaining parts of the thesis. We discuss nonlinear optimization and distinguish between *convex* problems and *nonconvex* problems and their properties. We also briefly discuss problems that have integer variables and how to solve them. In this chapter we also give an introduction to *Convex Polytope Geometry* which will play a central roll in two of the upcoming chapters of the thesis.

Chapter 3 covers the necessary parts of flight dynamics to give the reader who is unfamiliar with aircraft flight dynamics the basics to understand the problem formulation and the applications. This chapter also contains a slightly more detailed description of the aircraft models and simulation environments used throughout this thesis.

Chapter 4 gives the reader an introduction to Model Predictive Control. For linear systems we present the main stability results, reference tracking concepts, practical aspects of robustifications such as integral control and soft constraints

and derive the explicit MPC formulation. For nonlinear systems we only briefly discuss the complicating factors such as guaranteed stability, recursive feasibility and nonconvexity of the optimization problem.

In the Chapters 5, 6 and 7 we present the main theoretical results of the thesis. In Chapter 5 we consider reference tracking MPC, while in Chapter 6 we combine a feedback linearization controller with a linear MPC structure to stabilize the nonlinear dynamics of a fighter aircraft. In Chapter 7 we discuss a stability test for model predictive controllers based on mixed integer programming.

Chapter 8 contains the main application results where we implement an MPC controller for the pitch dynamics of an agile fighter aircraft in Saab's development simulator for the JAS 39 Gripen aircraft. We investigate a slightly different approach with a different discretisation time step in the MPC controller than in the closed loop implementation. We also investigate how to utilise existing LQ design methodology to tune the MPC controller.

Finally in Chapter 9 we investigate a different approach to maneuver limiting with the application of a command governor to the nominal flight control system of Saabs main simulation model.

Some concluding remarks and thoughts on future work are discussed in Chapter 10.

How the different chapters correspond to the different publications is presented in the beginning of each chapter.

# Part I

# Theory

# 2

# Background on optimization and polytopic geometry

In this chapter we will provide the necessary mathematical tools which we will use frequently in the remaining chapters of the thesis. Section 2.1 will give a very brief introduction to mathematical optimization, distinguish between *convex* and *nonconvex* optimization problems and discuss the concept of duality. For clarity of the presentation we have skipped many important details and concepts and we refer the reader to Boyd and Vandenberghe [2004] for a comprehensive presentation of the material.

We will also briefly discuss optimization problems with integer variables and how to solve them in Section 2.1.4 since they are a fundamental part of the theory developed in Chapter 7.

Section 2.2 outlines basic properties and fundamental calculations with convex polytopic sets.

## 2.1 Optimization

An optimization problem is the problem of finding a value for the variable, $x$, which minimizes (or maximizes) a certain objective, possibly, while satisfying a set of constraints. A general formulation of an optimization problem can be written as

$$\begin{aligned} &\underset{x}{\text{minimize}} && f_0(x) && \text{(2.1a)}\\ &\text{subj. to} \\ &&& f_i(x) \leq 0 \quad i = 1, \ldots, m && \text{(2.1b)}\\ &&& g_i(x) = 0 \quad i = 1, \ldots, p && \text{(2.1c)} \end{aligned}$$

where $f_0 : \mathbb{R}^n \to \mathbb{R}$ is the objective function (or cost function) which we want to minimize and the functions $f_i : \mathbb{R}^n \to \mathbb{R}$ and $g_i : \mathbb{R}^n \to \mathbb{R}$ are the inequality and equality constraint functions. We will adopt the convention of using a straight inequality sign, $\leq$, for scalars and for element-wise vector valued inequalities, while we will use the curly inequality sign, $\preceq$, to denote the positive semidefiniteness property of matrices.

The optimal objective value is denoted as $f_0^*$ and the optimal (minimizing) variable value is denoted as $x^*$; i.e.,

$$f_0^* = \inf_x \{f_0(x) \mid f_i(x) \leq 0 \; i = 1, \ldots, m, \quad g_i(x) = 0 \; i = 1, \ldots, p\}$$
$$x^* = \left\{x \mid f_0(x) = f_0^*\right\}$$

A value $\hat{x}$ is said to be *feasible* if and only if $f_i(\hat{x}) \leq 0$, $\forall i = 1, \ldots, m$ and $g_i(\hat{x}) = 0$, $\forall i = 1, \ldots, p$ and it is *strictly feasible* if the inequalities hold strictly.

The problem (2.1) is often referred to as the *Primal problem* and a value $\hat{x}$, satisfying (2.1b) and (2.1c) as *Primal feasible*.

### 2.1.1 Convex optimization

The optimization problem (2.1) is said to be *convex* (or having a *convex representation*) if the objective function, $f_0$, is a convex function (if it is a minimization problem and concave if it is a maximization problem) of the variable $x$, the inequality constraint functions, $f_i$, are convex functions of $x$ and the equality constraint functions are *affine*, i.e., $g_i(x) = a_i^T x + b_i$. The constraints in the optimization problem form a set, $\mathcal{X}$, of feasible values of $x$, a set that must be convex in order for the whole optimization problem to be convex.

**Definition 2.1.** A set, $\mathcal{X}$, is said to be convex if and only if, for any two points $x_1$ and $x_2$ in $\mathcal{X}$, all points on the line between $x_1$ and $x_2$ also belong to the set $\mathcal{X}$, i.e., if

$$x_1, x_2 \in \mathcal{X} \Rightarrow \gamma x_1 + (1 - \gamma) x_2 \in \mathcal{X} \; \forall \; 0 \leq \gamma \leq 1$$

then the set $\mathcal{X}$ is convex.

From this definition of convex sets we can draw the important conclusion that the intersection between any two (or more) convex sets is also a convex set.

**Example 2.2: Convex sets**

A closed halfspace in $\mathbb{R}^n$ is defined as

$$\left\{x \in \mathbb{R}^n \mid a_i^T x \leq b_i\right\}$$

The intersection of a number of such halfspaces constitutes a convex set, see

Figure 2.1. We write this as

$$\mathcal{P} = \left\{ x \in \mathbb{R}^n \mid \bigcap_i a_i^T x \leq b_i \right\}$$

or equivalently as

$$\mathcal{P} = \{ x \in \mathbb{R}^n \mid Ax \leq b \}$$

where $A$ has $a_i^T$ as its i:th row and $b$ has $b_i$ as its i:th element. Such an intersection is called a *Polyhedron* if it is unbounded and a *Polytope* if it is bounded. We will, with a slight abuse of language, only use the term polytope.

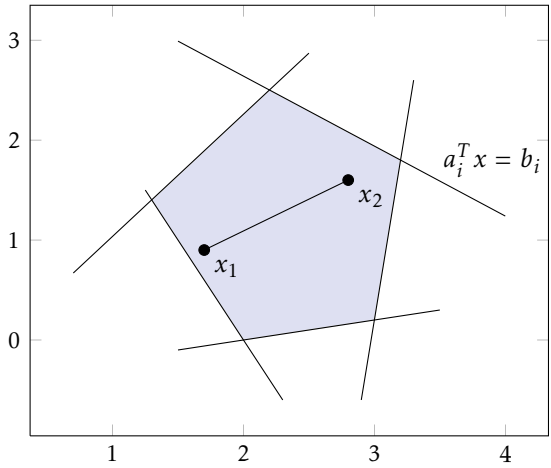


**Figure 2.1.** The figure shows a polytope, the shaded area, which is the intersection of five halfspaces (indicated with the lines $a_i^T x = b_i$) where each halfspace constitutes one edge of the polytope and each vertex is the intersection of two halfspaces. Additionally the figure shows two arbitrary points in the polytope and the line connecting them. It is evident that the entire line belongs to the set for any two points, hence the polytope is a convex set.

Let us now continue by defining convex functions

**Definition 2.3 (Boyd and Vandenberghe [2004]).** A function, $f(x)$, is said to be *convex* if the domain of $f$, $\mathrm{dom}(f)$, is convex and

$$f(\gamma x_1 + (1-\gamma)x_2) \leq \gamma f(x_1) + (1-\gamma) f(x_2)$$

for any two points $x_1, x_2 \in \mathrm{dom}(f)$ and any scalar $0 \leq \gamma \leq 1$.

In other words, a function is convex if the function curve between any two points lies below the line connecting those two points, see Figure 2.2. The

function is *concave* if the opposite holds. From the definition we can derive the first and second order conditions for convexity.

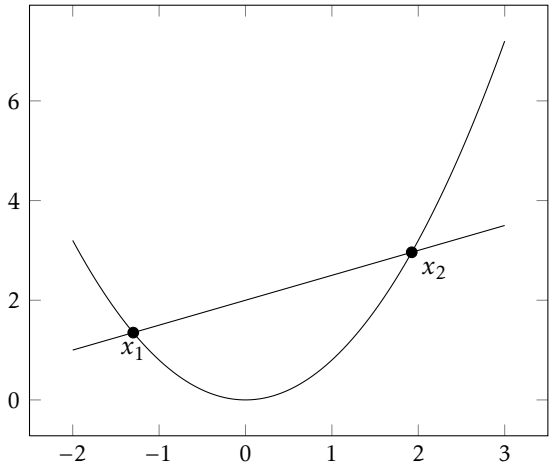


**Figure 2.2.** An example of a convex function and its relation to the straight line that passes through two arbitrary points. The curve segment of the convex function between the two points always lies below the line.

**Definition 2.4 (Boyd and Vandenberghe [2004]).** A differentiable function is convex if and only if it for every two points $x_1, x_2 \in \mathrm{dom}(f)$ satisfy

$$f(x_2) \geq f(x_1) + \nabla f(x_1)(x_2 - x_1)$$

or for a twice differential function

$$\nabla^2 f \geq 0$$

and the function is strictly convex if the inequalities hold strictly.

We illustrate these definitions with an example.

**Example 2.5: Convex functions**

Using the definition of convexity we can see that the norm function, $f(x) = ||x||$, is convex. This follows from the triangle inequality

$$\begin{aligned} f(\gamma x_1 + (1-\gamma)x_2) &= ||\gamma x_1 + (1-\gamma)x_2|| \\ &\leq ||\gamma x_1|| + ||(1-\gamma)x_2|| \\ &= \gamma\,||x_1|| + (1-\gamma)\,||x_2|| \\ &= \gamma f(x_1) + (1-\gamma) f(x_2) \end{aligned}$$

As another example, consider a quadratic function $f(x) = x^T Q x + q^T x + c$. Differentiating this function twice we have

$$\nabla^2 f = Q$$

This shows that a quadratic function is convex if and only if $Q$ is positive semidefinite.

Affine functions, $f(x) = a^T x + b$, the negative logarithm, $f(x) = -\log x$, and the max-function, $f(x) = \max\{x_1, x_2, \ldots, x_n\}$ are some other important examples of convex functions.

Convex optimization problems, also called *convex programs*, are generally divided into several different *standard forms* such as, e.g., *Linear Programs* (LP's), *Semidefinite Programs* (SDP's), *Geometric programs* (GP's) and *Quadratic programs* (QP's). In this thesis we will mostly consider QP's since, as we will see in Chapter 4, the control problems that we consider very often can be formulated as QP's.

A QP has a convex quadratic objective function, $f_0$, and affine constraint functions, $f_i$ and $g_i$

$$\underset{x}{\text{minimize}} \; x^T Q x + q^T x + c \tag{2.2a}$$

subj. to

$$Fx \leq b \tag{2.2b}$$

$$Gx = h \tag{2.2c}$$

The constraint functions form the feasible set defined by the intersection of the polytope, $Fx \leq b$, and the hyperplane $Gx = h$.

**Example 2.6: Discrete time LQ controller**

The discrete time LQ problem is given by

$$\underset{u_i, x_i}{\text{minimize}} \; \sum_{i=0}^{\infty} x_i^T Q x_i + u_i^T R u_i$$

where the states $x_i$ have to satisfy the state equation

$$x_{i+1} = Ax_i + Bu_i, \quad x_0 = x(0)$$

We can see this as an (infinite dimensional) equality constrained QP in the variables $x_i$ and $u_i$.

One great advantage and fundamental property of convex optimization problems that makes them very useful is given in the following theorem.

**Theorem 2.7.** *If $x^*$ is a local minimum of a convex optimization problem (2.1), then it is also a global minimum of the problem. Furthermore if $f_0$ is strictly convex, then the minimum is unique.*

**Proof:** Let $x^*$ be a local minimum of $f_0$, i.e.,

$$f_0(x^*) = \inf_x \{f_0(x) \mid x \in \mathcal{X}, \|x - x^*\|_2 \leq R\}$$

Now assume that there exist a feasible $\hat{x}$, with $||\hat{x} - x^*||_2 > R$, such that $f_0(\hat{x}) < f_0(x^*)$. Since both $\hat{x}$ and $x^*$ are feasible there exist a feasible point as the convex combination of $\hat{x}$ and $x^*$

$$\tilde{x} = \theta\hat{x} + (1-\theta)x^*$$

With $\theta = \frac{R}{2||\hat{x}-x^*||_2} < \frac{1}{2}$ we have $||\tilde{x} - x^*||_2 = \theta\,||\hat{x} - x^*||_2 = R/2 < R$ and by convexity of $f_0$ we have

$$f_0(\tilde{x}) \leq \theta f_0(x^*) + (1-\theta) f_0(\hat{x}) < f_0(x^*)$$

but this contradicts the assumption that $f_0(x^*)$ was the minimum within the neighborhood of radius $R$, hence $\hat{x}$ cannot exist and $x^*$ is the global minimum.

To show uniqueness of the solution assume instead that $f_0(x^*) = f_0(\hat{x})$ and that $f_0$ is strictly convex. Then it directly follows that

$$f_0(\tilde{x}) < \theta f_0(x^*) + (1-\theta) f_0(\hat{x}) = f_0(x^*)$$

which also contradicts the assumption that $x^*$ (and $\hat{x}$) are minimum of $f_0$. □

An optimization problem is said to be *unbounded below* if the optimal objective value is $f_0^* = -\infty$ and *infeasible* if it is $f_0^* = \infty$.In order to derive necessary and sufficient conditions for a point $x^*$ to be the global minimum of a convex representation of the problem (2.1) we need to consider something called *Duality*.

### 2.1.2 Duality

Let us start by defining the *Lagrangian function* for the optimization problem (2.1) as

$$\mathcal{L}(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i g_i(x)$$

The variables $\lambda_i$ and $\nu_i$ are the so called *dual variables* or *Lagrange multipliers*. We also define the *Dual function* as

$$\mathcal{D}(\lambda, \nu) = \inf_{x \in \mathcal{S}} \mathcal{L}$$

where $\mathcal{S} = \left(\bigcap_{i=0}^{m} \mathrm{dom}(f_i)\right) \bigcap \left(\bigcap_{i=1}^{p} \mathrm{dom}(g_i)\right)$ is the domain of the problem. It can easily be shown that for $\lambda \geq 0$ the dual function fulfills $\mathcal{D}(\lambda, \nu) \leq f_0^*$ for all $x \in \mathcal{S}$. So the dual function defines a global lower bound for the optimal value of the primal problem (2.1).

From this we define the *Dual problem* as

$$\begin{aligned} &\underset{\lambda,\nu}{\text{maximize}} && \mathcal{D}(\lambda, \nu) \\ &\text{subj. to} \\ &&& \lambda \geq 0 \end{aligned}$$

with the additional implicit constraint that the dual function must be bounded from below, i.e., $\mathcal{D}(\lambda, \nu) > -\infty$. The optimal value, $\mathcal{D}^*$, to the dual problem is the best lower bound for the primal problem. In the case when this best bound is tight, i.e., $\mathcal{D}^* = f_0^*$, we say that *Strong duality* holds. In the practical cases we will consider in this thesis (e.g., for most convex problems) we can assume that strong duality holds. This dual problem is one of the key details in the derivation of the controller in chapter 5.

**Example 2.8: Dual problem to an LP problem**

Consider the following LP maximization problem

$$\underset{x}{\text{maximize}} \;\; a^T x + b \quad \text{subj. to} \quad F^T x \leq g$$

This is equivalent to the minimization problem

$$\underset{x}{\text{minimize}} \;\; -(a^T x + b) \quad \text{subj. to} \quad F^T x \leq g$$

The Lagrangian is then given by

$$\mathcal{L}(x, \lambda) = -(a^T x + b) + \lambda^T (F^T x - g)$$

and the dual function

$$\begin{aligned} \mathcal{D}(\lambda) &= \inf_x \mathcal{L} \\ &= \inf_x \; -\lambda^T g - b + (F\lambda - a)^T x \\ &= \begin{cases} -\lambda^T g - b & \text{if} \quad F\lambda - a = 0 \\ -\infty & \text{otherwise} \end{cases} \end{aligned}$$

The dual problem is then given by

$$\begin{aligned} &\underset{\lambda}{\text{maximize}} \;\; -(g^T \lambda + b) \\ &\text{subj. to} \\ &\qquad \lambda \geq 0 \\ &\qquad F\lambda - a = 0 \end{aligned}$$

We now have the tools to characterize the optimal solution, $x^*$, to a convex optimization problem. As stated above, for most convex problems, strong duality holds, i.e., $\mathcal{D}(\lambda^*, \nu^*) = f_0(x^*)$. The definition of $\mathcal{D}$ then gives

$$f_0(x^*) = \mathcal{D}(\lambda^*, \nu^*) = f_0(x^*) + \sum_{i=1}^{m} \lambda_i^* f_i(x^*) + \sum_{i=1}^{p} \nu_i^* g_i(x^*)$$

and since $g_i(x^*) = 0$ it must hold that

$$\sum_{i=1}^{m} \lambda_i^* f_i(x^*) = 0 \tag{2.3}$$

This means that if the constraint is not active then the corresponding dual variable must be equal to zero, i.e., if $f_i(x^*) < 0$ then $\lambda_i = 0$ and if the constraint is active, i.e., $f_i(x^*) = 0$ the dual variable can be nonzero. Note also that both $\lambda_i$ and $f_i$ can be zero at the same time (then $f_i$ is called a *weakly active* constraint). The condition (2.3) is called *complementary slackness*.

Furthermore since $x^*$ minimizes the Lagrangian it must also hold that the gradient is zero,

$$\nabla \mathcal{L} = \nabla f_0(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla f_i(x^*) + \sum_{i=1}^{p} \nu_i^* \nabla g_i(x^*) = 0 \tag{2.4}$$

To summarize, for a point $x^*$ to be optimal for a convex instance of the optimization problem (2.1) it is necessary and sufficient that the following conditions hold

$$\nabla f_0(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla f_i(x^*) + \sum_{i=1}^{p} \nu_i^* \nabla g_i(x^*) = 0 \tag{2.5}$$

$$f_i(x^*) \leq 0 \tag{2.6}$$

$$g_i(x^*) = 0 \tag{2.7}$$

$$\lambda^* \geq 0 \tag{2.8}$$

$$\sum_{i=1}^{m} \lambda_i^* f_i(x^*) = 0 \tag{2.9}$$

These conditions are called the *Karush-Kuhn-Tucker conditions*, (KKT). The conditions (2.6) and (2.7) require $x^*$ to be primal feasible and the condition (2.8) requires $\lambda$ to be feasible for the dual problem.

Note that for the KKT conditions to be sufficient conditions for optimality strong duality must hold. We have not discussed under what conditions strong duality holds for a convex program; we merely state that for our applications (except for parts of Chapter 7) strong duality does hold and refer the details to Boyd and Vandenberghe [2004].

We will use the KKT conditions in Section 4.2.5 to derive an explicit formulation of the model predictive controller.

### 2.1.3 Nonconvex optimization

In the previous section we did not discuss how to actually solve a convex optimization problem. There are different methods to solve such problems, e.g., by using so called gradient methods which generally searches for the optimal point in the direction of the negative gradient. Due to the special properties

of convex problems described in Theorem 2.7, this search will eventually lead us to the global optimal solution. However if the problem is nonconvex, then one can not use local information to find the global optimal solution, and this makes non-convex problems much harder to solve and requires us to use more complex solution algorithms.

**Example 2.9: Controller for a nonlinear system**

Let us consider the same objective function as in Example 2.6, but now with nonlinear system dynamics

$$\begin{aligned} \underset{u_i, x_i}{\text{minimize}} \quad & \sum_{i=0}^{\infty} x_i^T Q x_i + u_i^T R u_i \\ & x_{i+1} = f(x_i, u_i), \quad x_0 = x(0) \end{aligned}$$

Since the equality constraint now consists of a nonlinear function, the optimization problem is no longer convex (that would require the equality constraint to be affine).

In Chapter 6 we will look more into some methods to overcome this difficulty and present a new method that approximates this nonlinear program with an easily solvable QP.

The techniques to solve nonconvex problems can be divided into two distinct categories, local techniques and global techniques.

The local techniques, such as, e.g., the *Broyden-Fletcher-Goldfarb-Shanno* algorithm (BFGS) or *Sequential Quadratic Programming* (SQP), use a convex approximation of the problem around an initial guess of the optimum to calculate the search direction. This will lead the algorithm to converge to a local optimum, which unfortunately can be far off from the true global solution. The local solutions are obviously very dependent on the initial guess of the optimal point. The benefit of local methods in comparison to global techniques is that they are relatively faster.

In contrast to local methods, techniques that find the true global optimal solution are extremely computationally expensive and even for modest scale problems they can take several hours to converge to the global solution. The global solution methods are either *heuristic* or *nonheuristic* in their nature. One nonheuristic method that has received much attention is the *Branch and bound* method. We will detail the branch and bound method further in the next section where we discuss the special type of nonconvex programs that contain integer variables, the so called (mixed) integer programs.

Another nonheuristic method that has gained significant popularity in the last decade is the so called *Semidefinite relaxation*. The relaxation of a general optimization problem

$$\underset{x}{\text{minimize}} \quad \{f_0(x) \mid x \in \mathcal{X}\}$$

is another optimization problem

$$\underset{x}{\text{minimize}} \ \left\{\tilde{f}_0(x) \mid x \in \tilde{\mathcal{X}}\right\}$$

such that $\tilde{f}_0(x) \leq f_0(x), \forall x \in \mathcal{X}$ and $\mathcal{X} \subseteq \tilde{\mathcal{X}}$. In semidefinite relaxation the resulting relaxed optimization problem is an SDP. Lasserre [2001] propose a method to find the global solution to a nonconvex polynomial problem by solving a sequence of semidefinite relaxations of the original problem.

There also exist several heuristic methods for global optimization such as, e.g., *Simulated Annealing*, *Direct Monte-Carlo Sampling* and *Particle Swarm Optimization*. These methods essentially perform a guided random search converging to the globally optimal solution if given sufficient time.

### 2.1.4 Mixed integer optimization

In the preceding sections all variables in the optimization problems have been real variables, $x \in \mathbb{R}^n$. But there exist another important class of optimization problems, the so called *integer programs* that only contain integer variables, $z \in \mathbb{Z}^m$, or *mixed integer programs* that has both real and integer variables. Optimization problems containing integer variables are nonconvex problems. This is easy to realize if we consider Definition 2.1 of a convex set. When we have integer valued variables, no points on the line between two integer valued points will belong to the set, i.e., it is not a convex set and hence the problem is nonconvex.

If the optimization problem is a QP problem that has (some) integer variables it is called a (mixed) integer quadratic program (MIQP) and in the same way it is called a (mixed) integer linear program (MILP) if it is an LP problem. It should be pointed out that in this thesis, i.e., Chapter 7, we will only consider the type of integer LP problems where the integer variables in fact are binary, i.e., *mixed binary LP problems*. However we will with a slight abuse of notation refer to them as mixed integer linear programs.

A mixed integer linear program can be written in a general form as

$$\underset{x,z}{\text{minimize}} \ f^T x + g^T z \tag{2.10a}$$

subj. to

$$Ax + Ez \leq b \tag{2.10b}$$

$$z \in \mathbb{Z}^m \tag{2.10c}$$

We will not go into the details of integer programming in this thesis. For that, the reader is referred to, e.g., the book by Wolsey [1998]. Instead we will focus on briefly explaining how mixed integer linear programs can be solved.

Integer and mixed integer problems are in general very hard to solve and one can not use gradient descent methods that is used for convex problems to

solve these problems. Instead methods for solving nonconvex problems need to be applied. Two of the more wide spread methods are the *Branch and bound method* and the *Cutting plane method*.

### The branch and bound method

The Branch and bound method is one of the most widespread global solvers for non convex problems. The basic idea of the branch and bound method is to partition (branching) the feasible set into subsets and successively building a search tree over partitions of the feasible set.

For each of these subsets the algorithm calculates an upper and lower bound on the optimal value by solving simpler problems. The upper bound could be found by using either one of the local optimization techniques as described earlier or by simply selecting any feasible point. The lower bound can be found by solving the (always convex) dual problem or by some convex relaxations. The upper and lower bounds are compared for each of the partitions and if the lower bound of one partition has a higher value than the upper bound in some other partition, this partition can surely not contain the global optimum, and hence it can be discarded (pruned). The algorithm then continues by splitting the best partitions into smaller and smaller partitions repeating the computations and comparisons of the bounds until all partitions are discarded but one. One big advantage with the branch and bound method is that it can quantify the level of suboptimality. This in contrast to the local methods that are unable to provide us with such quantities.

Since we are mostly interested in mixed integer linear programs in this thesis let us briefly exemplify how these are solved using branch and bound.

The algorithm starts by computing an upper and lower bound to the mixed integer linear program (2.10). An upper bound on the optimal can be found, e.g., by selecting any feasible point $z \in \mathbb{Z}^m$, $x \in \mathbb{R}^n$ such that $Ax + Ez \leq b$. A lower bound on the optimal solution to this problem can be found by simply solving the LP relaxation

$$\underset{x,z}{\text{minimize}} \quad f^T x + g^T z \tag{2.11a}$$

$$\text{subj. to}$$

$$Ax + Ez \leq b \tag{2.11b}$$

that is given by ignoring the integrality constraint. Why this is a lower bound can be realized by noting that the feasible set of the LP relaxation is larger than that of the mixed integer problem, i.e., it contains all rational numbers, $z$, such that $Ax + Ez \leq b$, not only integers.

Then the algorithm continues by splitting (branching) the set into a number of subsets (nodes). The branching can be done in several ways, e.g., one can split a subset into new ones along dimensions for which the optimal solution

to (2.11) is rational [Wolsey, 1998] or one can simply split the subset in half along the dimension for which the subset is the largest.

A node is pruned, or discarded from further splitting, if the LP problem in this node is infeasible, if the optimal solution is integer or if the lower bound is larger than the best global upper bound, because then the optimum can not be found in this branch.

There is also a decision on how to search through the tree in order to quickly find the global minimum. One can, e.g., always start by searching down the tree in the path that has the lowest lower bound.

The branch and bound method is often combined with the cutting plane method in what is called a branch and cut method.

### The cutting plane method

The cutting plane method is based on the observation that the convex hull, $\text{conv}(\mathcal{S})$, of the set $\mathcal{S} = \{z \in \mathbb{Z} \mid Az \leq b\}$ can be used to efficiently solve integer LP problems. To see this we must observe that all vertices of the convex hull are integer points and since the optimum to an LP always is at a vertex of the feasible set the optimal solution to the LP relaxation

$$\begin{aligned} &\underset{z}{\text{minimize}} && c^T z \\ &\text{subj. to} \\ & && z \in \text{conv}(\mathcal{S}) \end{aligned}$$

is, in fact, equal to the optimal solution to the integer LP. However the convex hull is often quit expensive to compute and we are only interested in having a good approximation of it around the optimal point.

The cutting plane method works by successively adding linear inequalities (cutting planes) to the LP relaxation in order to cut of any non-integer optimal solutions until the solution to the LP relaxation is an integer solution.

The inequalities that are added to the LP relaxation must be so called *valid inequalities* [Wolsey, 1998] which basically means that the inequality can not cut of integer solutions from the feasible set, i.e., the inequality $g_i^T z \leq h_i$ is a valid inequality for the set $\mathcal{S}$ if

$$g_i^T z \leq h_i \; \forall \; \{z \in \mathbb{Z}^m \mid Az \leq b\}$$

There exist several types of valid inequalities such as, e.g., Chvátal-Gomory cuts, lift and project cuts, split cuts and rounding cuts. For more information on constructing valid inequalities see, e.g., Marchand et al. [2002] or Cornuéjols [2008].

## 2.2 Convex polytopic geometry

In this section, we will consider a branch of mathematics which is concerned with geometric problem solving and computation with simple convex sets of different dimension such as, e.g., points, lines, hyperplanes and polytopes. We will detail fundamental properties and some basic algebraic calculations with polytopic sets which are particularly interesting from the perspective of our application. For a more in depth description of these topics we refer to Grünbaum [2003] and for more general computational geometry problems we refer the reader to the work of de Berg et al. [2008].

Let us first recall the definition of a polytope from the previous section. We define a polytope as the bounded intersection of a finite number of halfspaces

$$\mathcal{P} = \{x \mid Ax \leq b\}$$

We will refer to this as the *Halfspace representation*.

Using the halfspace representation we can show that the intersection of two (or more) polytopes is a new polytope (unless it is the empty set). Consider two polytopes

$$\mathcal{P}_1 = \{x \mid A_1 x \leq b_1\}, \quad \mathcal{P}_2 = \{x \mid A_2 x \leq b_2\}$$

then the intersection can be written as the polytope

$$\mathcal{P}_3 = \mathcal{P}_1 \bigcap \mathcal{P}_2 = \left\{ x \mid \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} x \leq \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \right\}$$

In this description of the polytope there can be a number of redundant hyperplanes which can be removed through solving a set of LPs in order to have a minimal representation of the polytope. Algorithms for doing this can be found in Baotic [2005].

For a set of $N$ points, $X = \{x_1, x_2, \ldots, x_N\}$, the *Convex hull*, $conv(X)$, is defined as the smallest polytope that contains all $N$ points. This set can be represented as the convex combination of all points

$$conv(X) = \sum_{i=1}^{N} \beta_i x_i, \quad \forall \ 0 \leq \beta_i \leq 1, \ \sum_{i=1}^{N} \beta_i = 1$$

Similar to the halfspace representation there exist a *vertex representation* of a polytope which is the minimal representation of its convex hull, i.e.,

$$\mathcal{P} = \sum_{i=1}^{v_p} \beta_i v_i$$

where $v_i$ are those $x_i$ that constitute the vertices of the polytope and $v_p$ is the number of vertices in the polytope, see Figure 2.3. Algorithms for extracting the convex hull from a set of points can be found in de Berg et al. [2008].

The task of going from the halfspace representation to the vertex representation

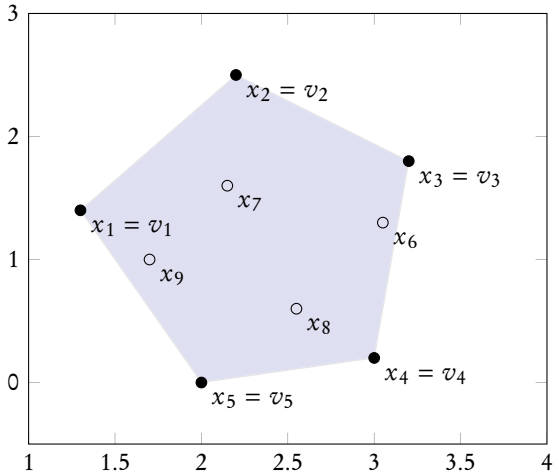


**Figure 2.3.** An example of a set of nine points $x_i$, and the convex hull of those points (the shaded area). The points that constitute the vertices of the convex hull are the vertex representation of the shaded polytope.

of a polytope is known as the *vertex enumeration problem* and the opposite (vertex to halfspace representation) is known as the *facet enumeration problem*. These are complicated and possibly very computational expensive tasks and in Avis and Fukuda [1992] the authors present one algorithm for performing those.

Let us now continue with some basic polytope operations which will be useful in the following chapters.

For two polytopic sets, $\mathcal{P}_1$ and $\mathcal{P}_2$, the *Minkovsky sum* is the set

$$\mathcal{P}_1 \oplus \mathcal{P}_2 = \{x_1 + x_2 \mid x_1 \in \mathcal{P}_1,\ x_2 \in \mathcal{P}_2\}$$

and correspondingly the *Pontryagin difference* of two sets is the set

$$\mathcal{P}_1 \ominus \mathcal{P}_2 = \{x_1 \mid x_1 + x_2 \in \mathcal{P}_1, \forall x_2 \in \mathcal{P}_2\}$$

An illustration of these two polytope operations are shown in Figure 2.4.

Furthermore, the *Cartesian product* of the two sets is

$$\mathcal{P}_1 \times \mathcal{P}_2 = \{(x_1, x_2) \mid x_1 \in \mathcal{P}_1,\ x_2 \in \mathcal{P}_2\}$$

If the dimension of $\mathcal{P}_1$ is $\mathbb{R}^n$ and $\mathcal{P}_2$ is $\mathbb{R}^m$, then the dimension of the product is $\mathbb{R}^{m+n}$.

**Definition 2.10.** For a polytopic set $\mathcal{P}$ with the halfspace representation

$$\mathcal{P} = \{x \mid Ax \leq b\}$$

we define

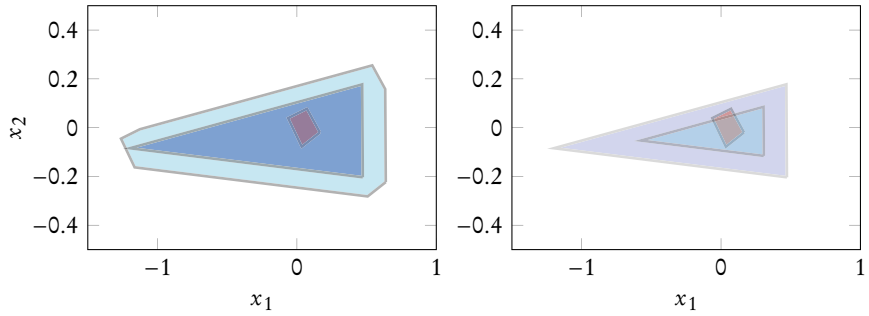


**Figure 2.4.** The left axis shows an example of the Minkovsky sum. The two most inner polytopes are $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively. The outer polytope is the Minkovsky sum of $\mathcal{P}_1$ and $\mathcal{P}_2$. The right axis shows the same two polytopes and the Pontryagin difference as the smaller triangular polytope.

- the *scaling* of the set with a positive scalar $\alpha$ as

$$\alpha\mathcal{P} = \{x \mid Ax \leq \alpha b\}$$

- the *translation* of the set to an arbitrary point $y$ as

$$\mathcal{P}(y) = \mathcal{P} \oplus y = \{x \mid A(x - y) \leq b\}$$

Note that the translation of a polytope to an arbitrary point is the same as the Minkovsky sum of the set, $\mathcal{P}$, and a set consisting of only one point, $y$. It is straightforward to derive the vertex representation of scaling and translation as

$$\alpha\mathcal{P} = \alpha \sum_{i=1}^{v_p} \beta_i v_i$$

and

$$\mathcal{P}(y) = y + \sum_{i=1}^{v_p} \beta_i v_i$$

Given a polytope, $\mathcal{P}$, the interior of the set, $\text{int}(\mathcal{P})$, is defined as all those points $y \in \mathcal{P}$ for which there exists an $\epsilon > 0$ such that for any point $x$ in $\mathcal{P}$, $y$ is within an $\epsilon$-radius of $x$.

$$\text{int}(\mathcal{P}) = \left\{y \mid \|x - y\|_2 \leq \epsilon, x \in \mathcal{P}\right\} \subseteq \mathcal{P}$$

One could also more loosely describe these points as all points in $\mathcal{P}$, except the border. Since the interior is an open set it is practical to work with the closely related *$\epsilon$-interior* to a set.

**Definition 2.11.** For a given constant $0 \leq \epsilon \leq 1$ and set $\mathcal{P}$ containing the

origin in its interior, let $\text{int}_\epsilon(\mathcal{P})$ denote the $\epsilon$-interior of $\mathcal{P}$, i.e.,

$$\text{int}_\epsilon(\mathcal{P}) = (1-\epsilon)\mathcal{P} = \{x \mid Ax \leq (1-\epsilon)b\}$$

This is a key detail in the stability proof of our controller in Chapter 5.

The affine transformation, $F(\cdot)$, of a set, $\mathcal{P}$, defined by a matrix $A$ and vector $b$ is an affine map of all elements in $\mathcal{P}$.

$$F(\mathcal{P}) = \{Ax + b \mid x \in \mathcal{P}\}$$

and we will adopt the short hand notation $F(\mathcal{P}) = A\mathcal{P} + b$ for this. Additionally there exist an inverse affine mapping defined through

$$F^{-1}(\mathcal{P}) = \{x \mid Ax + b \in \mathcal{P}\}$$

This inverse mapping is central to the calculation of *invariant sets* which is an important detail of the stability of model predictive controllers.

Note that the scaling and translation defined above are just special cases of the affine mapping.

We argued in the beginning of this section that the intersection of two polytopes is a new polytope, but in fact, all the polytope operations we have defined will result in a new polytope [Boyd and Vandenberghe, 2004].

For our intended application it is also needed to be able to calculate some kind of center point of a polytope. The center point that we consider is the so called *Chebychev center*, which is the center point of the largest ball (Chebychev ball) that can be inscribed in the polytope.

A *Ball* is all points $x$ that are within a radius $R$ from a center point $x_c$

$$\mathcal{B}(x_c, R) = \{x_c + x \mid ||x||_2 \leq R\}$$

For a polytope $\mathcal{P}$ we find the Chebychev center from the following optimization problem

$$\begin{aligned} &\underset{x_c, R}{\text{maximize}} \quad R \\ &\text{subj. to} \\ &\mathcal{B}(x_c, R) \subseteq \mathcal{P} \end{aligned}$$

To see how this abstract formulation can be written as a simple linear program first note that if $\mathcal{B}(x_c, R) \subseteq \mathcal{P}$ it must hold for all facets of the polytope. This means that $a_i^T(x_c + x) \leq b_i$, $\forall i = 1, \ldots, m$, where $m$ is the number of facets in $\mathcal{P}$. Also note that this constraint shall hold for all $||x||_2 \leq R$ and so it must hold for the worst case $x$. Thus we obtain

$$\sup_x \left\{a_i^T x \mid ||x||_2 \leq R\right\} = \left\|a_i^T\right\|_2 R$$

and we can write the constraint as

$$a_i^T x_c + \left\| a_i^T \right\|_2 R \leq b_i \quad \forall\, i = 1, \ldots, m$$

The resulting LP becomes

$$\underset{x_c, R}{\text{maximize}} \quad R \tag{2.12a}$$

$$\text{subj. to}$$

$$a_i^T x_c + \left\| a_i^T \right\|_2 R \leq b_i \quad \forall\, i = 1, \ldots, m \tag{2.12b}$$

# 3

# Aircraft flight dynamics and flight control design

In this chapter we will derive the necessary equations to give the reader enough background to the dynamics of flight to understand the applications of the research. Since the goal is not to be a comprehensive source of information regarding flight dynamics we will cut some corners in the presentation and also make some elementary assumptions. For a full treatment of the subject the reader is pointed to the excellent books of Stevens and Lewis [2003] and Nelson [1998].

The first assumption is that the aircraft is a single engine aircraft of military type that is symmetric in its x-z plane, see Figure 3.1. This is not a significant assumption but not having to take civil aircraft configurations into account will simplify some of the equations and arguments. Secondly we assume that the flat earth approximation is valid and neglect the earth rotation since this will significantly simplify the equations and it serves our purpose well. The final assumption is that the aircraft is a rigid body. This will also greatly simplify the equations since we do not have to take any structural dynamics into account.

First we will derive the full 6 DOF nonlinear equations of motion of the aircraft. These equations will then be simplified to a set of linear equations that are suitable for control law design and explain the objectives of the control law design. The last section will describe the model and simulation environments used in this thesis.

## 3.1 The nonlinear dynamics

In order to describe the dynamics of flight we first have to say some words about different coordinate systems. There are several different coordinate systems used to describe the motion of an aircraft, e.g., the body fixed coordinate system, wind axis system, stability axis system and the inertial reference system. We are not going to go into details about these but the reader should be aware of them since several of the interesting variables discussed are defined in the different systems.

The inertial system or the earth fixed reference system is defined in a north, east, down, direction (the so called NED system) with the origin on a fixed point on the earth's surface. The body fixed coordinate system has its origin in the aircraft center of mass and its x-axis is pointing out through the nose of the aircraft. The y-axis is pointing out through the right hand side wing and the z-axis is pointing down, see Figure 3.1

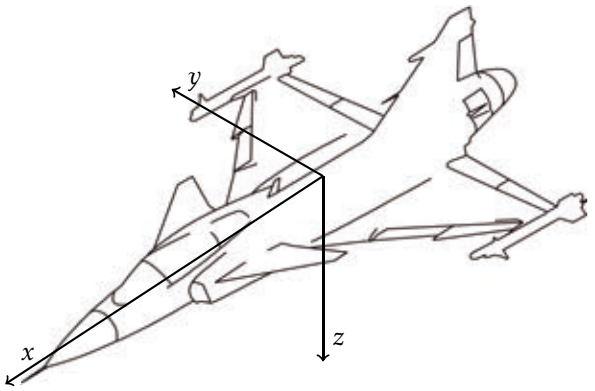


**Figure 3.1.** Definition of the body fixed coordinate system.

The relation between the NED and the body fixed coordinate systems are described by the position

$$P_{NED} = [p_N \quad p_E \quad -p_D]$$

and the orientation, the so called Euler angles, of the body fixed frame

$$\Phi = [\phi \quad \theta \quad \psi]$$

where $\phi$ is the roll (or bank) angle, $\theta$ is the pitch angle and $\psi$ is the yaw angle respectively, see Figure 3.2.

The wind axis coordinate system has its x-axis aligned with the incoming airflow, or in direction of the airspeed vector. The relation between the wind axis and the body fixed systems are defined through the angle of attack, $\alpha$, and sideslip angle, $\beta$, see Figure 3.2. Finally the flight path angle, $\gamma$, relates the wind axis system to the NED system.

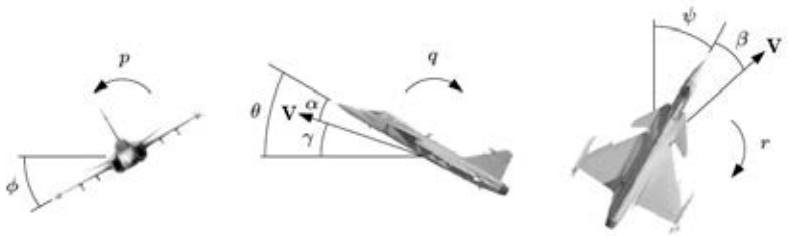


**Figure 3.2.** Definition of the Euler angles, $\phi$, $\theta$, $\psi$, the aerodynamic angles, $\alpha$ and $\beta$ and the flight path angle, $\gamma$, and the angular rates, $p$, $q$ and $r$ [Härkegård, 2003, p. 10].

### 3.1.1 Equations of motion

Newton's second law of motion in a rotating reference frame gives us a starting point for formulating the equations of motion. It states

$$F = m\dot{V} + \omega \times mV \tag{3.1}$$

for the forces acting on the aircraft and

$$T = I\dot{\omega} + \omega \times I\omega \tag{3.2}$$

for the moments. $F$ is the total force acting on the aircraft, $T$ the total moment, $m$ is the aircraft mass, $I$ the inertia, $V = [u \;\; v \;\; w]^T$ is the velocity and finally $\omega = [p \;\; q \;\; r]^T$ is the angular velocity of the aircraft.

The total force acting on the aircraft has three main contributions, the gravitation, $g$, the thrust from the engine, $F_T$, and the aerodynamic forces, $F_{x,y,z}$. The gravity acts in the downward direction of the NED system and thrust is assumed to act only in the direction of the x-axis of the body fixed coordinate system.

The moments will be assumed to mainly come from the aerodynamics. Other effects like thrust vectoring or engine misalignment will not be considered in this thesis.

The aerodynamic forces and moments acting on the aircraft arise due to the airflow around the aircraft as it flies through the atmosphere. The forces and moments depend on many different properties such as the air density, $\rho$, the velocity of the aircraft, $V$, the geometry of the aircraft as well as its orientation relative to the airflow. The geometry of the aircraft is usually characterized by the wing area, $S$, and either wingspan, $b$ or mean aerodynamic chord $\bar{c}$.

The aerodynamic forces can be modeled in the body fixed coordinate system as

$$F_x = \frac{1}{2}\rho \bar{V}^2 S C_x$$

$$F_y = \frac{1}{2}\rho \bar{V}^2 S C_y$$
$$F_z = \frac{1}{2}\rho \bar{V}^2 S C_z$$

where $\bar{V} = \sqrt{u^2 + v^2 + w^2}$, $C_{(\cdot)}$ are the so called aerodynamic coefficients. It is also common to formulate the force equations in the wind axis system using the drag coefficient, $C_D$, the side force coefficient, $C_Y$, and the lift force coefficient, $C_L$. The aerodynamic coefficients depend on many variables such as control surface deflections, $\delta$, aerodynamic angles, $\alpha$, $\beta$, the angular rates, $p$, $q$ and $r$ (see Figure 3.2), as well as their derivatives. As an example, the lift force coefficient, $C_L$, is increasing with increasing angle of attack, up to a certain point, the stall angle, then it drops rapidly, see Figure 3.3.

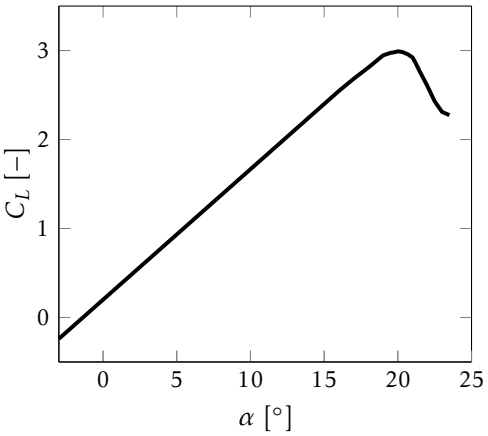


**Figure 3.3.** One example of how the aerodynamic lift coefficient, $C_L$, can depend on the angle of attack, $\alpha$.

The aerodynamic moments are modeled in a similar way with

$$T_y = \frac{1}{2}\rho \bar{V}^2 S \bar{c} C_m$$

for the pitching moment and

$$T_x = \frac{1}{2}\rho \bar{V}^2 S b C_l$$
$$T_z = \frac{1}{2}\rho \bar{V}^2 S b C_n$$

for the rolling and yawing moment respectively.

Modeling of the aerodynamic coefficients is difficult, they are in general nonlinear functions that depend on many variables and vary with Mach number[1]. In the early stages of an aircraft development a first estimate of the

[1]The Mach number id defined as the ratio of the local airflow to the speed of sound, $M = \frac{u}{c}$

coefficients are created using wind tunnel testing and CFD (*Computational Fluid Dynamics*) calculations as well as handbooks with aerodynamic properties of known aircraft geometries. In later stages flight testing and system identification is used to refine the models.

Now, combining the equations for the aerodynamic forces and moments with Newton's law of motion we obtain the nonlinear equations for the dynamics of flight

$$F_T + \frac{1}{2}\rho\bar{V}^2 S C_x - mg\sin\theta = m(\dot{u} + qw - rv) \tag{3.3a}$$

$$\frac{1}{2}\rho\bar{V}^2 S C_y + mg\sin\phi\cos\theta = m(\dot{v} + ru - pw) \tag{3.3b}$$

$$\frac{1}{2}\rho\bar{V}^2 C_z + mg\cos\phi\cos\theta = m(\dot{w} + pv - qu) \tag{3.3c}$$

$$\frac{1}{2}\rho\bar{V}^2 S b C_l = I_x\dot{p} - I_{xz}\dot{r} + (I_z - I_y)qr - I_{xz}pq \tag{3.4a}$$

$$\frac{1}{2}\rho\bar{V}^2 S\bar{c} C_m = I_y\dot{q} + (I_x - I_z)pr + I_{xz}(p^2 - r^2) \tag{3.4b}$$

$$\frac{1}{2}\rho\bar{V}^2 S b C_n = I_z\dot{r} - I_{xz}\dot{p} + (I_y - I_x)pq + I_{xz}qr \tag{3.4c}$$

The equations (3.3) are the force equations in body axis coordinates and (3.4) are the body axis moment equations. $I_{(\cdot)}$ represent the appropriate element in the inertia matrix, $I$.

To obtain the total description we also need the attitude equations

$$\dot{\theta} = q\cos\phi - r\sin\phi \tag{3.5a}$$

$$\dot{\phi} = p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \tag{3.5b}$$

$$\dot{\psi} = (q\sin\phi + r\cos\phi)\sec\theta \tag{3.5c}$$

and the navigation equations

$$\dot{P}_{NED} = T_{nav}V \tag{3.6}$$

where

$$T_{nav} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

The equations for the forces in the X and Z dimension, (3.3a) and (3.3c) together with the pitching moment (3.4b) and the attitude equation (3.5a) form the so called longitudinal dynamics. Similarly the equations for Y-force (3.3b) together with the roll (3.4a) and yaw (3.4c) moments and the attitude equations (3.5b) and (3.5c) form the lateral dynamics.

The free response of the longitudinal dynamics, i.e., the solution to the

differential equations with no control surface input, is characterized by two types of motions, the slower *Phugoid mode* and the high frequency *Short period mode*. In the Phugoid mode the aircraft has a slowly varying pitch angle, $\theta$, with fairly constant angle of attack, $\alpha$, while the Short period mode instead is a high frequency oscillation in the angle of attack under almost constant $\theta$. The corresponding lateral modes are the *Roll mode*, *Spiral mode* and the *Dutch roll mode*.

All of the above modes can either be stable, i.e., returning to an equilibrium point, or unstable, i.e., diverging from an equilibrium point. One of the objectives of the flight control system is to stabilize any unstable modes and to give the modes desirable properties in terms of amplitude, damping and frequency.

## 3.2 The linearized dynamics

If we intend to utilize linear design techniques to design the flight control laws then the nonlinear equations of motion derived in the previous section are not suitable but needs to be simplified. To do this one rearranges the equations (3.3) and (3.4) and then use *small disturbance theory* to derive a set of linear differential equations

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{3.7}$$

where $x$ represent the deviation in the states from a steady flight condition and $u$ is the deviation from steady state value of the selected control inputs. We will not go into details on how this is done, just state the results, the interested reader is referred to the excellent exposition in Nelson [1998].

It should be pointed out that the states here do not necessarily need to be only the states in the dynamics of motion derived in the previous section but can also include, e.g., control surface deflection angles as states. The control inputs are then the commanded control surface deflections and the servo dynamics are normally modeled as a first order system.

### Longitudinal model

For the design of the inner loop control laws which we will discuss in the next section the phugoid mode is most often neglected and only the short period mode is modeled.

The short period dynamics, which is the dynamics we will mainly be focusing on in this thesis, can then be modeled as a two state linear system with the angle of attack, $\alpha$, as one state and the pitch angular rate, $q$, as the other. The input to the system is the elevator control surface deflection, $\delta_e$.

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} \frac{Z_\alpha}{u} & 1 \\ \left(M_\alpha + M_{\dot{\alpha}} \frac{Z_\alpha}{u}\right) & \left(M_q + M_{\dot{\alpha}}\right) \end{bmatrix} \begin{bmatrix} \alpha \\ q \end{bmatrix} + \begin{bmatrix} \frac{Z_{\delta_e}}{u} \\ M_{\delta_e} + \frac{M_{\dot{\alpha}}}{u} Z_{\delta_e} \end{bmatrix} \delta_e \tag{3.8}$$

Here $M_{(\cdot)}$ is the derivative of the moment equation (3.4b) with respect to the different variables and $Z_{(\cdot)}$ is the derivative of the force equation (3.3c) with respect to the different variables. The derivatives have the form [Stevens and Lewis, 2003]

$$Z_\alpha = \frac{-C_{N_\alpha} QS}{m} \qquad Z_{\delta_e} = \frac{-C_{N_{\delta_e}} QS}{m}$$
$$M_\alpha = \frac{C_{m_a} QS\bar{c}}{I_y} \qquad M_q = \frac{C_{m_q} QS\bar{c}^2}{2uI_y}$$
$$M_{\delta_e} = \frac{C_{m_{\delta_e}} QS\bar{c}}{I_y} \qquad M_{\dot{\alpha}} = \frac{C_{m_{\dot{\alpha}}} QS\bar{c}^2}{2uI_y}$$

where we have introduced the *dynamic pressure*, $Q = \frac{1}{2}\rho \bar{V}^2$.

The normal acceleration (or the load factor) in the center of gravity is defined as the ratio between the aircraft lift force and aircraft weight and it is given by

$$n_{z,cg} = -\frac{F_z}{mg}$$

but the load factor experienced by the pilot becomes

$$n_z = n_{z,cg} + \frac{\dot{q}\Delta l}{g}$$

where $\Delta l$ is the distance along the body fixed x-axis from the center of gravity to the pilot. This can be incorporated into the linear dynamics as an output from (3.8).

### Lateral model

The corresponding differential equations for the lateral dynamics, i.e., the roll mode and the dutch roll mode, is a three state dynamical system with the states, roll rate, $p$, sideslip angle, $\beta$ and yaw rate, $r$, and with the control inputs aileron deflection, $\delta_a$ and rudder deflection, $\delta_r$. The state space model can be written as

$$\begin{bmatrix} \dot{\beta} \\ \dot{p} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{Y_\beta}{u} & \frac{Y_p}{u} & \frac{Y_r}{u} - 1 \\ L'_\beta & L'_p & L'_r \\ N'_\beta & N'_p & N'_r \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \end{bmatrix} + \begin{bmatrix} 0 & \frac{Y_{\delta_r}}{u} \\ L'_{\delta_a} & L'_{\delta_r} \\ N'_{\delta_a} & N'_{\delta_r} \end{bmatrix} \begin{bmatrix} \delta_a \\ \delta_r \end{bmatrix} \tag{3.9}$$

where

$$L'_{(\cdot)} = L_{(\cdot)} + \frac{I_{xz}}{I_x} N_{(\cdot)}, \quad N'_{(\cdot)} = N_{(\cdot)} + \frac{I_{xz}}{I_z} L_{(\cdot)}$$

and where the derivatives are given by

$$Y_\beta = \frac{QSC_{y_\beta}}{m} \qquad Y_p = \frac{QSbC_{y_p}}{2mu}$$

$$Y_r = \frac{QSbC_{y_r}}{2mu} \qquad Y_{\delta_r} = \frac{QSC_{y_{\delta_r}}}{m}$$

$$L_\beta = \frac{QSbC_{l_\beta}}{I_x - \frac{I_{xz}^2}{I_z}} \qquad L_p = \frac{QSb^2C_{l_p}}{2u(I_x - \frac{I_{xz}^2}{I_z})}$$

$$L_r = \frac{QSb^2C_{l_r}}{2u(I_x - \frac{I_{xz}^2}{I_z})} \qquad L_{\delta_a} = \frac{QSbC_{l_{\delta_a}}}{I_x - \frac{I_{xz}^2}{I_z}}$$

$$L_{\delta_r} = \frac{QSbC_{l_{\delta_r}}}{I_x - \frac{I_{xz}^2}{I_z}} \qquad N_\beta = \frac{QSbC_{n_\beta}}{I_z - \frac{I_{xz}^2}{I_x}}$$

$$N_p = \frac{QSb2C_{n_p}}{2u(I_z - \frac{I_{xz}^2}{I_x})} \qquad N_r = \frac{QSb2C_{n_r}}{2u(I_z - \frac{I_{xz}^2}{I_x})}$$

$$N_{\delta_a} = \frac{QSbC_{n_{\delta_a}}}{I_z - \frac{I_{xz}^2}{I_x}} \qquad N_{\delta_r} = \frac{QSbC_{n_{\delta_r}}}{I_z - \frac{I_{xz}^2}{I_x}}$$

Note the zero element in the B-matrix of (3.9). The effect of aileron deflection on the side force is neglectable and hence very often set to zero.

## 3.3 Fighter aircraft flight control law design

The design of flight control laws is a very challenging task, especially for fighter aircraft. The flight control system shall provide stabilization, disturbance rejection and desired characteristics of the short period, roll, dutch roll and spiral modes as well as automatic trimming in all axes and superior pilot handling qualities in all modes of operation.

The aircraft dynamics varies significantly over the full operating envelope. The speed ranges from low subsonic speed where the aircraft usually are open loop unstable to supersonic speed where they are stable. Furthermore, different combinations of external stores change both mass, inertia, aerodynamics as well as the center of gravity of the aircraft. For modern military aircraft there can be over a thousand different external store combinations. The amount of fuel and how the different tanks are emptied also affects the mass and center of gravity location. In addition to this, a fighter aircraft is maneuvered aggressively over a large span of angle of attack, angle of sideslip and roll rate which makes the nonlinearities of the dynamics non negligible.

The flight control system shall be designed to give maximum attainable performance in all conditions while at the same time respecting bandwidth limitations of actuators and sensors. Also the bandwidth of the flight control system must be low enough to not excite any structural vibrations and flutter. The flight control system must also be designed such that there are no PIO (Pilot Induced Oscillations) tendencies.

All the above must be met also in a number of failure cases of, e.g., sensors or

actuators. Therefore the flight control system must be able to reconfigure and redistribute control commands in case of actuator failures or manage any input data loss in order to provide safe continued flight and landing. The system safety requirements on flight control systems are very high, no single failure may cause a loss of the aircraft so the flight control system must incorporate an advanced system for test and monitoring of all parts of the system.

It is difficult to obtain a good overview of the industry state of art design principles for flight control systems of advanced fighter aircraft since it is most often considered intellectual property rights and very little material is published on the matter. Although, one good review paper is the one written by Balas [2003]. In this paper it is evident that the industry utilize both linear and nonlinear design techniques. One common approach seem to be the combination of Nonlinear Dynamic Inversion (NDI) together with classical proportional and integral control techniques and this has been used in e.g., the F-35 Joint strike fighter aircraft, the Boeing X-36 aircraft and in some versions of the VAAC Harrier. Other common approaches are to use gain scheduled $\mathcal{H}_\infty$ or LQ techniques and these has been used in, e.g., the Boeing F/A-18 aircraft, the Saab JAS 39 Gripen and the VAAC Harrier.

Let us now discuss, in a bit more detail, one procedure of designing flight control laws using linear and gain scheduled techniques.

Due to the vast complexity of the overall design task one has to divide the problem into smaller simpler design tasks. A common approach is to linearize the dynamics in a number of different operating points of Mach, altitude and angle of attack and in each of these tune the control laws for different configurations such as external stores combinations, mass and center of gravity locations. Usually one starts of with a small operating envelope and a single external store and design and tune the control laws for this case and then one gradually extend the flight envelope and adds additional stores combinations. It is crucial for the quality of the design that there is a close cooperation between the control design engineers and the pilots since most often the pilots' opinions are far more stringent then the requirements set out in any documents.

It is quite common to separate the control law design of the longitudinal dynamics and the lateral dynamics into two separate design tasks. The basic structure of the flight control laws for both the longitudinal and lateral dynamics are shown in Figure 3.4 and in consists of a feedback part, a feed forward part and an integral control part.

The reference input is calculated from the pilot control stick and pedal input. For agile fighter aircraft it is most suitable according to Stevens and Lewis [2003] that the reference input for the longitudinal dynamics is either pitch rate or load factor. There is a close relation between the load factor and the angle of attack. For a given angle of attack the load factor increases quadratically as a function of speed, which means that the maximum angle of attack, before the aircraft stalls, limits the attainable load factor for low Mach

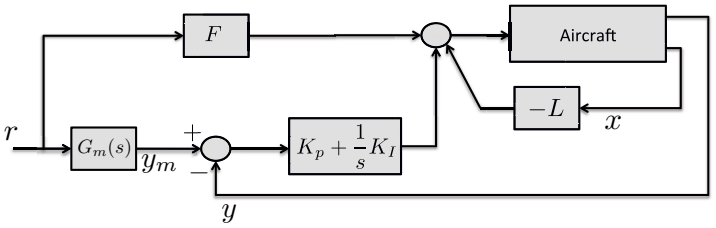


**Figure 3.4.** The nominal LQ controller of the ARES model.

numbers. While for higher Mach numbers the structural load on the airframe and the pilot will limit the maximum load factor, see Figure 3.5. The speed

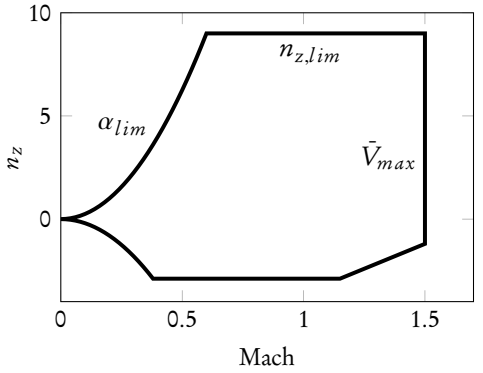


**Figure 3.5.** The attainable normal load factor envelope as a function of speed (Mach). At low speeds the maximum (or minimum) $C_L$ will limit the attainable load factor.

where the angle of attack limit meets the load factor limit is known as the *maximum maneuvering speed* or *corner speed*.

Due to this relation it can be convenient to use a combination of load factor and angle of attack as reference input, using load factor above corner speed, when the load on the pilot and airframe is high, and angle of attack below corner speed, when the loads are small but a high angle of attack risks to stall the aircraft. For the lateral dynamics it is most common to use roll rate, $p$, and sideslip angle, $\beta$, as the controlled variables.

The feedback loop is designed to achieve required damping and frequency of the short period, roll and dutch roll modes such that they meet the level 1 requirements in MIL-F-8785C [1980] and to fulfill the flying and handling qualities criteria specified in, e.g., Gibson [1999]. It shall also meet requirements regarding disturbance rejection and flutter. The inner feedback loops shall be designed explicitly taking the pilot feedback loop into account, since this has proven to give exceptional handling qualities and robustness once the real pilot is closing the loop. The feedback loop also need to compensate for the change

in dynamics from the different external stores. Due to the vast amount of combinations it is not tractable to design a separate feedback for every store combination but instead more efficient techniques must be used.

The feedforward loop should be designed concurrently with the feedback loop in order to achieve excellent pilot handling qualities. It usually consists of either a static feedforward gain, scheduled over the flight envelope, or a dynamic prefilter for shaping the pilot command, and rate limit filters to remove any PIO tendencies. The rate limit filters can advantageously be designed with phase compensation using feedback such as those in Rundqwist and Ståhl-Gunnarsson [1996] to improve stability margins compared to classic rate limit filters. To achieve carefree maneuvering and reduce any overshoots of, e.g., angle of attack and load factor limits, command limiting functions are also added to the feedforward path. The roll command from the pilot is translated into both an aileron command and a rudder command in the feedforward path of the lateral dynamics such that the aircraft rolls around the velocity vector or the bullet trajectory since this most often is more suitable than rolling around the x-axis.

The integral control is designed as a PI controller and the objective of the integral part is to account for any model uncertainties and gain scheduling errors. The PI controller integrates the error between an ideal output from a response model and the actual output.

For aircraft with a delta-canard configuration such as the JAS 39 Gripen, Eurofighter Typhoon and the Dassault Rafale, the control commands can be distributed to the control surfaces in different ways. The control surfaces on the trailing edge of the main wing are used for both pitch and roll motion and special care must be taken to account for this in the design. Also the pitch command can be actuated with both the trailing edge control surfaces and the canard wings. The trade off between these should be made in a drag optimal way.

Once the linear control laws are designed it is common to add additional nonlinear feedback terms to account for the nonlinearities in, e.g., high angle of attack situations.

In the design it is very important to minimize all types of time delays not only in the control system itself but also to the display system and the head up display (HUD) since the pilot is closing the loop with the display system. Investigations and experience from earlier design projects has shown that pilots in general act as a proportional feedback gain in the control loop. Furthermore, in order to reduce the pilot workload, the flight control system shall be designed such that the longitudinal and lateral dynamics are de-coupled to the extent possible also for very asymmetric external store combinations.

Due to the iterative nature of the design process flight control laws should be designed to enable new functionality or performance to be added piece by piece. The performance and safety of the flight control laws are verified through

extensive analysis and a huge amount of linear and nonlinear simulations. This requires very accurate models to be developed and maintained throughout the development. The first iteration of the control laws are based on models derived from physical modeling and wind tunnel testing. Once the aircraft proceeds to flight testing the models are continuously updated with real flight data.

The industry has collected a set of best practices for flight control design that can be found in Nato [2000].

## 3.4 The ARES and ADMIRE models

In this thesis we will mainly use two different aircraft simulation tools to derive linear models for control system design and to implement and simulate the closed loop system. For the simpler examples used in the theoretical development in chapters 5 to 7 we have used the ADMIRE tool and for the implementation studies in complex environments presented in chapter 8 and 9 we have used an in-house tool, ARES, at Saab Aeronautics.

Both simulation tools model a single engine fighter aircraft with a delta-canard wing configuration, see Figure 3.1. The canard wings are the two small wing-shaped control surfaces just behind the cockpit. They are separately controlled but they are in general only commanded equally to control the pitching moment. The elevon control surfaces are the control surfaces on the trailing edge of the main wing and they are used as combined elevator and aileron control surfaces, generating both pitching and rolling moment.

The ADMIRE (Aero-Data Model In Research Environment) tool is a Simulink based simulation environment that implements a nonlinear rigid body motion with nonlinear aerodynamics based on the *Generic Aerodata Model* (GAM), developed by Saab and the Swedish Defense Research Agency. It has simplified models for engine, actuators and sensors.

For a complete description of the ADMIRE tools the reader is referred to Forssell and Nilsson [2005].

The ARES (Aircraft Rigid-Body Engineering System) tool is the main simulation tool used at Saab Aeronautics. It is an in-house developed simulation environment that can be used both for desktop simulations as well as for software and hardware simulator testing. The ARES tool is used in the JAS 39 Gripen development program for all types of flight dynamical simulations such as, performance investigations, pilot training and flight control law design and clearance.

The ARES tool consists of a core simulation environment in which different submodels are linked together, see Figure 3.6. The different submodels can be replaced to simulate different aircraft and configurations.

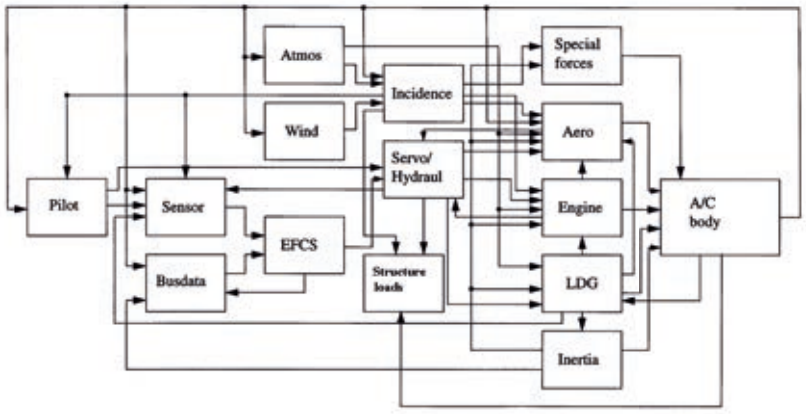


**Figure 3.6.** The structure of the ARES simulation environment and its submodels.

Let us shortly describe the purpose and complexity of the different submodels in order to give an overview of the realism of the ARES tool.

**A/C body:** This model implements the rigid body equations (3.1) and (3.2) and based on the forces and moments calculate the velocities and angular velocities.

**Inertia:** Calculates the aircraft mass, inertia matrix and center of mass based on, e.g., the amount of fuel and external stores.

**Aero:** Calculates the aerodynamic forces and moments from tables of the aerodynamic data that is scheduled over speed, altitude as well as angle of attack and sideslip. The tables have been constructed from wind tunnel testing, CFD[2] calculations and flight testing. The aerodata model used in this thesis is based on the GAM.

**Engine:** A linear dynamical model of the engine simulating engine thrust, airflow and engine speed.

**LDG:** Landing gear model simulating the gear extension and retraction as well as the suspension and nose wheel steering.

**Special forces:** This submodel models other forces affecting the rigid body motion than the standard ones described above such as, e.g., automatic cannon recoil.

**Structure loads:** Calculates the structural loads on the different parts of the airframe.

**Servo/Hydraul:** A simulation model for the control surface actuator servos

[2]Computational Fluid Dynamics

and the hydraulic system. It calculates pressure and flow in the hydraulic system from pump models and calculates hydraulic consumption in the servos.

**Incidence:** This submodel calculates the wind axis variables, angle of attack, $\alpha$, and sideslip angle, $\beta$.

**Atmos:** This is the atmosphere model. It is the International Standard Atmosphere (ISA), which models the pressure, density, temperature, viscosity etc. of the atmosphere as a function of altitude.

**Wind:** The wind model is the Dryden turbulence model developed by the United States Department of Defense.

**EFCS:** Models the algorithm of the Electronic Flight Control System (EFCS). It is in this model we implement the control algorithms studied in this thesis.

**Sensor:** This is a model of the sensors supplying data to the flight control system. It models acceleration, angular rate, angle of attack and angle of sideslip sensors.

**Busdata:** Models the bus communication between the flight control computer and other computers in the aircraft.

**Pilot:** This model implements a number of different maneuvers that a pilot performs. It is intended to be used in Monte-Carlo simulations.

The different submodels are implemented in C, C++ or Fortran and all the models are automatically linked together in a correct way during compilation of the environment. The compilation can be done for desktop and Monte-Carlo simulations or to interface with real hardware in the simulators.

The ARES model can be trimmed in different steady state flight conditions as initial conditions for simulation or to extract linear models such as (3.8) and (3.9) for control law design and analysis.

### 3.4.1 ARES baseline LQ controller

The nominal control system that is implemented in ARES for the GAM aerodynamics is a gain scheduled controller with similar principle structure as the controller in the JAS 39 Gripen aircraft. It has an LQ designed feedback from the states together with a proportional and integral feedback from the tracking error and a static feedforward from the pilot command. A simplified schematic of the controller structure is shown in Figure 3.4.

For the control law design the ARES model is linearized around trimmed level flight at 25 different envelope points of Mach and altitude in the subsonic region and the design is made separately for the longitudinal and lateral dynamics.

For the longitudinal control law design the short period dynamics (3.8) is augmented with the actuator dynamics for the canard and elevator control

surfaces. The actuator dynamics is modeled as a first order system with a time constant of 0.05 seconds from control surface command, $u(t)$, to control surface deflection angles, $\delta_c$ and $\delta_e$ respectively

$$\begin{bmatrix} \dot{\delta}_c(t) \\ \dot{\delta}_e(t) \end{bmatrix} = \begin{bmatrix} -20 & 0 \\ 0 & -20 \end{bmatrix} \begin{bmatrix} \delta_c(t) \\ \delta_e(t) \end{bmatrix} + \begin{bmatrix} 20 & 0 \\ 0 & 20 \end{bmatrix} u(t) \tag{3.10}$$

The output, $y(t)$, is either angle of attack or normal load factor, depending on which variable we want to control.

From these linearized models the LQ feedback gain, $L$, and feedforward gain, $F$, are calculated in each envelope point and linear interpolation is made between the design points. The feedforward term is a static gain calculated such that the closed loop system should have a unit static gain in each design point, i.e.,

$$C(sI - A + BL)^{-1}F + DF = 1$$

The feedback and feedforward gains are tuned to ideally give the closed loop response $y = G_m(s)r$ where, when the actuator dynamics are neglected, $G_m(s)$ is

$$G_m(s) = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} \tag{3.11}$$

and where the damping, $\zeta$, and frequency, $\omega_0$, vary with speed and altitude. The parameters $\zeta$ and $\omega$ are design parameters and the closed loop system is tuned such that they meet requirements in, e.g., MIL-F-8785C [1980] and pilot experience.

The pilot command to the controller is the increment in load factor, $\Delta n_{z,cmd}$, from trimmed flight. The reference input, $r$, to the controller is then calculated from the pilot command as both a total load factor command

$$n_{z,cmd} = \Delta n_{z,cmd} + \cos\theta$$

and an angle of attack command,

$$\alpha_{cmd} = K_{\alpha/n_z}\Delta n_{z,cmd} + \alpha_{trim}$$

The pilot command, $\Delta n_{z,cmd}$, is limited such that the load factor command and angle of attack command ideally stays within the specified design limits

$$-3 \le n_{z,cmd} \le 9$$
$$-8 \le \alpha_{cmd} \le 18$$

The controller then tracks either the angle of attack command, $\alpha_{cmd}$, or the load factor, $n_{z,cmd}$, command based on the current speed (over corner speed the controller tracks the load factor command and below corner speed it tracks the angle of attack command). From the reference command a nominal response, $y_m$ is calculated as

$$y_m = G_m(s)r$$

where the model, $G_m(s)$, is the second order model (3.11) and is the same for

both angle of attack and load factor reference commands.

The nominal response is then compared to the actual response of the aircraft to calculate the model following error, $e = y_m - y$. The feedback term from the model following error is a proportional and integral term trying to integrate out all model errors such that the true closed loop response is as close as possible to the desired response throughout the whole flight envelope.

The lateral controller has the same principle structure as the longitudinal controller and is designed for the lateral dynamics (3.9) augmented with the actuator models for ailerons and rudder control surfaces. The pilot command roll rate, $p$, with the control stick, and sideslip angle, $\beta$, with the pedals. The feedforward from the pilot commands has cross coupling terms, i.e., the roll rate command has a feedforward to the rudder control surface command and vice versa.

Finally the commands $\delta_e$, $\delta_c$, $\delta_a$ and $\delta_r$ are limited and sent to the control surface servo loops. Note that in the ARES model the elevator command and aileron command are sent to the combined control surfaces, the elevons. This means that he commands are added together before they are sent to the corresponding actuator.

# 4

# Introduction to model predictive control

In this chapter we give the background to the *Model Predictive Control* (MPC) problem formulation and present some variants and extensions of the theory. Section 4.1 gives a short historical background to the development of MPC. In Section 4.2 we derive the MPC controller formulation for linear systems and discuss stability and robustness issues as well as reference tracking. We also derive the explicit controller formulation. Finally the concepts of using MPC for nonlinear systems are briefly discussed in Section 4.3.

## 4.1 Introduction

As described in the introduction, adding constraints to the infinite horizon linear quadratic optimal control problem makes it in general extremely difficult to solve explicitly, i.e., finding an explicit formulation for $\kappa(x)$. Model Predictive Control offers a way to overcome these difficulties by defining an open loop optimal control problem, which is easier to solve than the original problem, and then iteratively solves it online, using the current state of the system as initial condition, giving a closed loop feedback controller.

Model predictive control research originated in the early 1960's with the work of Propoi [1963] but in the 1970's and 1980's much of the development came from the process industry with methods such as DMC, GPC, IDCOM, QDMC and SMOC. These techniques were extensively implemented in industry and hence solved real world problems with constraints and nonlinearities, however they did not build on firm theoretical foundations, e.g., they did not guarantee stability and required careful tuning and stable models. In the 1990's, several influential papers were published that rigorously developed the stability theory

and reference tracking formulation. A big leap in the theory development occurred in the early 2000's with the introduction of Explicit MPC that opened up a new area of research within the community. In the last two decades a large body of research has been done on nonlinear MPC and its theoretical foundations, and today linear and nonlinear MPC is one of the most promising advanced control techniques to handle constrained multivariable systems.

## 4.2 Linear MPC

The most common formulation of MPC is the discrete time version and therefore we will limit our discussion to discrete time systems and the discrete time formulation of the MPC controller (with the exception of Chapter 6 where we also discuss continuous time nonlinear systems). Hence let us for the remainder of this section assume that the system dynamics can be described by a linear discrete time difference equation

$$x_{i+1} = Ax_i + Bu_i \tag{4.1}$$

and the subscript $i$ is short for the time index, i.e., $x_i = x(iT_s)$ where $T_s$ is the sample time of the system.

Consider the discrete time constrained optimal control problem of the form

$$\underset{u_i, x_i}{\text{minimize}} \quad \sum_{i=0}^{\infty} \left( x_i^T Q x_i + u_i^T R u_i \right) \tag{4.2a}$$

subj. to

$$x_0 = x(t) \tag{4.2b}$$

$$x_{i+1} = Ax_i + Bu_i \tag{4.2c}$$

$$x_i \in \mathcal{X} \tag{4.2d}$$

$$u_i \in \mathcal{U} \tag{4.2e}$$

and assume that the sets $\mathcal{X}$ and $\mathcal{U}$ are convex. Then this can be viewed as a general convex optimization problem with an infinite number of decision variables $x_i$, $u_i$ $\forall\, i > 0$. Instead of solving the infinite dimensional optimization problem (4.2) it is possible to recast it as a finite problem by splitting the objective function (4.2a) into two parts

$$\underset{u_i, x_i}{\text{minimize}} \quad \sum_{i=0}^{N-1} \left( x_i^T Q x_i + u_i^T R u_i \right) + \sum_{i=N}^{\infty} \left( x_i^T Q x_i + u_i^T R u_i \right)$$

and try to find a way to approximate the last part of the objective function, or at least to bound it from above by some function $\Psi(x_N)$

$$\sum_{i=N}^{\infty} \left( x_i^T Q x_i + u_i^T R u_i \right) \leq \Psi(x_N) \tag{4.3}$$

Then the resulting optimal control problem will have a finite dimension.

$$\underset{u_i, x_i}{\text{minimize}} \quad \sum_{i=0}^{N-1} \left( x_i^T Q x_i + u_i^T R u_i \right) + \Psi(x_N) \tag{4.4a}$$

subj. to

$$x_0 = x(t) \tag{4.4b}$$

$$x_{i+1} = Ax_i + Bu_i \quad \forall\, i = 0, \ldots, N-1 \tag{4.4c}$$

$$x_i \in \mathcal{X} \quad \forall\, i = 0, \ldots, N-1 \tag{4.4d}$$

$$u_i \in \mathcal{U} \quad \forall\, i = 0, \ldots, N-1 \tag{4.4e}$$

$$x_N \in \mathcal{T} \tag{4.4f}$$

In order to make the approximation or bound (4.3) valid there might be some extra constraints (4.4f) on the state at the end of the horizon, $x_N$. We will discuss further how to choose $\Psi(x_N)$ and $\mathcal{T}$ in Section 4.2.1 since it relates closely to stability of the closed loop system.

By solving the optimization problem (4.4) with the current state as $x_0$, the solution is a sequence of $N$ optimal control inputs $u_i^*$, denoted $\{u_i^*\}_{i=0}^{N-1}$, for which the objective (4.4a) is minimized. Implementing this sequence of controls as inputs to the real system will result in an open loop controller. However it is widely known that closed loop control is preferred since all real systems suffer from disturbances and all models have errors [Skogestad and Postlethwaite, 2005].

To achieve closed loop control we implement, at each time step $k$, only the first control signal, $u_0^*$, in the optimal sequence $\{u_i^*\}_{i=0}^{N-1}$ and then in the next time step, $k+1$, measure the current state and redo the optimization with the new current state as $x_0$. This strategy is referred to as *Receding Horizon Control* since the control problem is solved over a future horizon that is progressing into the future as time evolves.

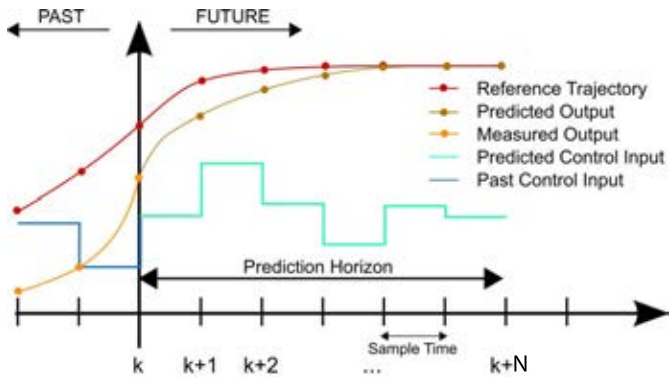


**Figure 4.1.** The receding horizon concept of Model Predictive Control (By Martin Behrendt, via Wikimedia Commons).

This could also be explained as at each discrete time instant $t = kT_s$ (where $T_s$ is the sample time of the controller) the current state of the system is measured $x(t) = x(kT_s) = x_k$ and the controller predicts the response of the system $N$ time steps into the future (see Figure 4.1). The optimal control input is selected based on the predicted future behavior of the system by solving the optimization problem (4.4). In conclusion, the Model Predictive Control law, $\kappa(x_k)$, is defined by the solution to the following optimization problem

$$V_k^* = \underset{u_{k+i}, x_{k+i}}{\text{minimize}} \sum_{i=0}^{N-1} \ell(x_{k+i}, u_{k+i}) + \Psi(x_{k+N}) \tag{4.5a}$$

subj. to

$$x_k = x(t) \tag{4.5b}$$

$$x_{k+i+1} = Ax_{k+i} + Bu_{k+i} \quad \forall\, i = 0, \ldots, N-1 \tag{4.5c}$$

$$x_{k+i} \in \mathcal{X} \quad \forall\, i = 0, \ldots, N-1 \tag{4.5d}$$

$$u_{k+i} \in \mathcal{U} \quad \forall\, i = 0, \ldots, N-1 \tag{4.5e}$$

$$x_{k+N} \in \mathcal{T} \tag{4.5f}$$

and by applying the first element, $u_k^*$, of the optimal solution sequence $\{u_{k+i}^*\}_{i=0}^{N-1}$ as control input, i.e.,

$$\kappa(x_k) = u_k^* \tag{4.5g}$$

In (4.5) we have generalized the notation by replacing the quadratic cost function, $x_{k+i}^T Q x_{k+i} + u_{k+i}^T R u_{k+i}$, with the more general expression, $\ell(x_k, u_k)$. We have also introduced the notation $V_k^*$ as the optimal objective function value. Note also that in (4.5), $k$ denotes the current time step and $i$ is used to denote the prediction steps into the future. This is only for notational convenience, in a real implementation there is no difference between (4.4) and (4.5). Although $\mathcal{X}$ and $\mathcal{U}$ can be any arbitrary convex sets we will in this thesis mainly consider polytopic constraint sets, i.e.,

$$\mathcal{X} = \{x \mid F_{\mathcal{X}} x \leq b_{\mathcal{X}}\}, \quad \mathcal{U} = \{u \mid F_{\mathcal{U}} u \leq b_{\mathcal{U}}\}$$

The formulation (4.5) is the standard linear MPC formulation and will, with a few exceptions, be used throughout this thesis, but there exist several other formulations with different objective functions and constraint formulations.

### 4.2.1 Stability

Due to the presence of constraints, the MPC formulation is a nonlinear control problem and to show stability for the closed loop system it is common to use *Lyapunov Stability Theory* [Khalil, 2002]. In this section we will give a brief outline of the stability properties for the MPC formulation. We assume in this section that the model of the system is perfect and that there are no disturbances acting on the system.

To begin with let us first briefly sketch how one can argue stability for the infinite horizon case. The infinite horizon MPC formulation (4.2), although it in general is impossible to solve, is stabilizing due to the *Principle of Optimality*. If $\ell(x_k, u_k)$ is a positive definite function such that $\ell(0,0) = 0$ and $\ell(x_k, u_k) \to \infty$ when $x_k \to \infty$, $u_k \to \infty$, then it follows that the optimal cost at time $k+1$, is

$$\begin{aligned} V_{k+1}^* &= \sum_{i=0}^{\infty} \ell(x_{k+1+i}^*, u_{k+1+i}^*) \\ &= \sum_{i=0}^{\infty} \ell(x_{k+1+i}^*, u_{k+1+i}^*) + \ell(x_k^*, u_k^*) - \ell(x_k^*, u_k^*) \\ &= \sum_{i=0}^{\infty} \ell(x_{k+i}^*, u_{k+i}^*) - \ell(x_k^*, u_k^*) \\ &= V_k^* - \ell(x_k^*, u_k^*) \\ &\leq V_k^* \end{aligned}$$

which means that for all $x_k \neq 0$ and $u_k \neq 0$, then the objective function (4.2a) is strictly decreasing as time progresses.

Furthermore since $V_k^*$ is bounded below by zero it must follow that $\ell(x_k^*, u_k^*) \to 0$ as $k \to \infty$ and hence $x_k^* \to 0$ and $u_k^* \to 0$.

For the finite time horizon formulation (4.5) this type of argument can not be used since the horizon is progressing over time, taking new information into account. The solution at time $k+1$ can therefore be very different from that at time $k$. For this case one has to carefully chose the function $\Psi(x_{k+N})$ and the terminal constraint $x_{k+N} \in \mathcal{T}$ to ensure stability. There exist several such choices and we will give a brief review of some different techniques proposed in the literature. For a more extensive survey we refer to the excellent review paper by Mayne et al. [2000].

If the system is stable, a pragmatic approach would be to assume all control signals $u_{k+i} = 0$ for $i \geq N$ and calculate (4.3) as the uncontrolled response of the system starting from the state $x_{k+N}$. This is approach proposed in [Muske and Rawlings, 1993] and [Rawlings and Muske, 1993]. Keerthi and Gilbert [1988] proposes instead to impose a new constraint, $x_{k+N} = 0$ (i.e., $\mathcal{T} = 0$), which for linear systems will give $\Psi(x_{k+N}) = 0$. With this setup they show that the closed loop system is stable. Additionally they show that the cost of the finite horizon optimal control problem (4.5) approaches that of the infinite horizon problem (4.2) as $N$ increases.

A natural relaxation of this approach would be to constrain the final state $x_{k+N}$, not to the origin, but instead to some small region, $\mathcal{T}$, around the origin in which a local controller, $u_k = \kappa(x_k)$, can take over and drive the system to the origin. This approach is called *Dual Mode Control* since, in theory, the controller is divided in two modes, one solving the receding horizon control

problem and one that is a local control law. However in practice the local controller is never implemented, it is only a theoretical construction to show stability. Combining this approach with a cost $\Psi(x_{k+N})$ which is the cost for the local controller to drive the states to the origin is, at least in academia, the most widely used formulation of the MPC controller (4.5).

Before we can state the formal stability theorem of the dual mode MPC, we first need a definition, which can also be found in Blanchini [1999].

**Definition 4.1.** The set $\mathcal{T} \subset \mathbb{R}^n$ is said to be *positively invariant* for a system $x_{k+1} = f(x_k)$ if for all $x_k \in \mathcal{T}$ the solution $x_{k+i} \in \mathcal{T}$ for $i > 0$.

With this definition in place we are now ready to state the main stability theorem.

**Theorem 4.2 (Mayne et al. [2000]).** *If the system (4.1) is controlled with the MPC law (4.5), where $\ell(x_k, u_k) \geq c(|(x_k, u_k)|)^2$ and $\ell(0,0) = 0$, the closed loop system is stable and asymptotically converges to the origin if the following conditions hold:*

1. *$\mathcal{T}$ is a positively invariant (see def. 4.1) set of the system (4.1) controlled with the feedback $u_k = \kappa(x_k)$ where $\kappa(x_k) \in \mathcal{U} \; \forall \, x_k \in \mathcal{T}$*
2. *$\mathcal{T} \subseteq \mathcal{X}$ with $0 \in \mathcal{T}$*
3. *$\Delta\Psi(x_k) + \ell(x_k, \kappa(x_k)) \leq 0, \; \forall \, x_k \in \mathcal{T}$, where $\Delta\Psi(x_k) = \Psi(x_{k+1}) - \Psi(x_k)$.*

The first two conditions ensures that the problem is *recursively feasible*, i.e., given that there exist a solution at one time, then there will exist a solution for all future time steps. The last condition ensures that the system asymptotically converges to the origin. The properties of $\Psi$ that fulfill condition 3 can be obtained if $\Psi$ is chosen to be a Lyapunov function upper bounding the infinite horizon cost when using the controller $\kappa(x_k)$.

To show the convergence let us assume that there exists a solution at time $k$ with an optimal cost $V_k^*$. Then, at the following time step we apply the feasible but possibly suboptimal control sequence $\{\hat{u}_{k+i}\}_{i=1}^N = \{u_{k+1}^*, \ldots, u_{k+N-1}^*, \kappa(x_{k+N}^*)\}$. Then the suboptimal cost, $V_{k+1}$, is

$$
\begin{aligned}
V_{k+1} &= \sum_{i=0}^{N-1} \ell(x_{k+1+i}^*, u_{k+1+i}) + \Psi(x_{k+1+N}) \\
&= \sum_{i=0}^{N-2} \ell(x_{k+1+i}^*, u_{k+1+i}^*) + \ell(x_{k+N}^*, u_{k+N}) + \Psi(x_{k+1+N}) \\
&\quad + \ell(x_k^*, u_k^*) - \ell(x_k^*, u_k^*) + \Psi(x_{k+N}^*) - \Psi(x_{k+N}^*)
\end{aligned}
$$

$$
\begin{aligned}
&= \underbrace{\sum_{i=0}^{N-1} \ell(x_{k+i}^*, u_{k+i}^*) + \Psi(x_{k+N}^*)}_{V_k^*} \\
&- \ell(x_k^*, u_k^*) + \ell(x_{k+N}^*, u_{k+N}) + \Psi(x_{k+1+N}) - \Psi(x_{k+N}^*) \\
&= V_k^* \underbrace{-\ell(x_k^*, u_k^*)}_{<0} + \underbrace{\ell(x_{k+N}^*, \kappa(x_{k+N})) + \Delta\Psi(x_{k+N})}_{\leq 0} \\
&< V_k^*
\end{aligned}
$$

The last inequality follows from the third property of the theorem and from this we conclude that the optimal cost at time $k+1$ fulfills

$$V_{k+1}^* \leq V_{k+1} < V_k^*$$

In other words, $V_k^*$ is strictly decreasing as long as $x_k^* \neq 0$ and $u_k^* \neq 0$. Hence $x_k^* \to 0$ and $u_k^* \to 0$.

The above derivation assumes that there exists a solution to (4.5) at time $k+1$, i.e., that $\hat{\mathbf{u}}$ is feasible, this is not trivial and it must be proven that there always exist a solution to the open loop problem, so called *recursive feasibility*. To show this let $\mathcal{X}_N$ be the set of $x$ where (4.5) is feasible. Assume $x_k \in \mathcal{X}_N$ with an optimal solution given by the sequence $\{u_{k+i}^*\}_{i=0}^{N-1}$ and with predicted optimal state trajectory $\{x_{k+i}^*\}_{i=1}^{N}$.

At the next time step $\{\hat{u}_{k+i}\}_{i=1}^{N} = \{u_{k+1}^*, u_{k+2}^*, \ldots, u_{k+N-1}^*, \kappa(x_{k+N}^*)\}$ is a feasible control sequence, since $x_{k+N}^* \in \mathcal{T}$ and hence $\kappa(x_{k+N}^*) \in \mathcal{U}$ according to Condition 1 of Theorem 4.2. The new, possibly suboptimal, state sequence at time $k+1$ is $\{\hat{x}_{k+i}\}_{i=2}^{N+1} = \{x_{k+2}^*, x_{k+3}^*, \ldots, x_{k+N}^*, x_{k+N+1}\}$ where $x_{k+N+1} = Ax_{k+N}^* + B\kappa(x_{k+N}^*)$. Since $\mathcal{T}$ is positively invariant w.r.t the system $x_{k+1} = Ax_k + B\kappa(x_k)$, it follows that $x_{k+N+1}$ stays in $\mathcal{T}$, i.e., the terminal state constraint $x_{k+N+1} \in \mathcal{T}$ is satisfied. This argument can be used recursively and shows that the problem is feasible at all $k$.

It should be noted that the property of recursive feasibility require that the model of the system is perfect and that no disturbances act on the system. We will discuss more on how to ensure recursive feasibility in the presence of model uncertainties and disturbances in Section 4.2.4.

Sznaier and Damborg [1987] proposes a dual mode formulation for linear systems where the local controller $\kappa(x_k) = -Kx_k$ and $\Psi(x_{k+N}) = x_{k+N}^T P x_{k+N}$ is the solution to a discrete time version of the unconstrained infinite horizon LQ problem, i.e.,

$$K = (R + B^T P B)^{-1} B^T P A \tag{4.6a}$$

where $P$ is the solution to the Riccati-equation

$$(A - BK)^T P (A - BK) - P = -Q - K^T R K \tag{4.6b}$$

The authors show that by iteratively solving (4.5) for an increasing horizon, $N$, until (4.4f) is satisfied (where $\mathcal{T}$ is an invariant set of the system controlled with the local controller, see 4.1) then the solution is also the solution to the constrained LQ problem (4.2). Later versions of this approach have explicitly incorporated (4.4f) as a constraint and use a fix horizon, $N$.

To see that this choice of local controller and terminal state penalty indeed satisfy the third property of Theorem 4.2 we note that since

$$\ell(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k$$

$$\Psi(x_{k+N}) = x_{k+N}^T P x_{k+N}$$

$$x_{k+1} = (A - BK)x_k$$

we obtain

$$\begin{aligned}
&\ell(x_{k+N}^*, \kappa(x_{k+N}^*)) + \Psi(x_{k+1+N}) - \Psi(x_{k+N}^*) \\
&= x_{k+N}^{*T} Q x_{k+N}^* + (K x_{k+N}^*)^T R K x_{k+N}^* + x_{k+1+N}^T P x_{k+1+N} - x_{k+N}^{*T} P x_{k+N}^* \\
&= x_{k+N}^{*T} Q x_{k+N}^* + x_{k+N}^{*T} K^T R K x_{k+N}^* + x_{k+N}^{*T} (A - BK)^T P (A - BK) x_{k+N}^* \\
&- x_{k+N}^{*T} P x_{k+N}^* \\
&= x_{k+N}^{*T} \underbrace{\left( Q + K^T R K + (A - BK)^T P (A - BK) - P \right)}_{(4.6b)} x_{k+N}^* \\
&= 0
\end{aligned}$$

where the last equality follows from the fact that $P$ and $K$ are the solution to the LQ problem and hence satisfy (4.6b). This shows that the dual mode MPC formulation with the LQ solution as local controller and terminal state penalty is recursively feasible, stabilizes a linear system and is optimal in the sense that it solves the infinite horizon constrained LQ problem when the final state constraint is inactive. These desirable properties make this formulation very attractive.

### 4.2.2 Reference tracking

When tracking a reference signal, i.e., the so called servo problem, the system shall not converge to the origin but settle at some steady state $(x_r, u_r)$ different from the origin, yielding the desired output.

At this steady state it must hold that

$$y_k = r \tag{4.7}$$

where $r$ is the external reference to be followed and since it is a steady state it must hold that

$$x_{k+1} = x_k = x_r \tag{4.8}$$

Given a controllable linear discrete time system

$$x_{k+1} = Ax_k + Bu_k \tag{4.9a}$$
$$y_k = Cx_k + Du_k \tag{4.9b}$$

inserting (4.7) and (4.8) into (4.9) we obtain

$$x_r = Ax_r + Bu_r$$
$$r = Cx_r + Du_r$$

Rearranging the equations gives the relation

$$\begin{bmatrix} A - I & B \\ C & D \end{bmatrix} \begin{bmatrix} x_r \\ u_r \end{bmatrix} = \begin{bmatrix} 0 \\ r \end{bmatrix} \tag{4.10}$$

This relation determines the steady state $x_r$ and control $u_r$ given a reference input $r$. However the combination of input and steady state that result in $y_k = r$ might be non-unique, i.e., if the matrix on the left hand side of (4.10) is singular. In that case, a reasonable choice is to choose a steady state which is the minimal norm input, which can be formulated as a convex problem [Meadows and Badgwell, 1998, Muske and Rawlings, 1993]

$$\underset{x_r, u_r}{\text{minimize}} \; u_r^T W u_r \tag{4.11a}$$
$$\text{subj. to}$$
$$\begin{bmatrix} A - I & B \\ C & D \end{bmatrix} \begin{bmatrix} x_r \\ u_r \end{bmatrix} = \begin{bmatrix} 0 \\ r \end{bmatrix} \tag{4.11b}$$
$$u_r \in \mathcal{U} \tag{4.11c}$$
$$x_r \in \mathcal{X} \tag{4.11d}$$

In the case that a reference signal $r$ results in an infeasible optimization problem (4.11), then one can instead solve

$$\underset{x_r, u_r}{\text{minimize}} \; (y_k - r)^T W (y_k - r)$$
$$\text{subj. to}$$
$$\begin{bmatrix} A - I & B \end{bmatrix} \begin{bmatrix} x_r \\ u_r \end{bmatrix} = 0$$
$$u_r \in \mathcal{U}$$
$$x_r \in \mathcal{X}$$

resulting in the steady state that keeps the output as close as possible to $r$ [Muske and Rawlings, 1993]. An alternative to this approach is to use a pseudo reference which we will discuss further later in this section.

Now a pragmatic approach to implement the reference tracking case is to simply shift the origin of the problem to the new setpoint and apply the standard MPC scheme on the translated system, i.e., in the original coordinate

system, penalize deviations from the steady state setpoints.

$$\underset{u_{k+i},x_{k+i}}{\text{minimize}} \quad \sum_{i=0}^{N-1} \ell(x_{k+i} - x_r, u_{k+i} - u_r) + \Psi(x_{k+N} - x_r) \tag{4.12a}$$

subj. to

$$x_{k+i+1} = Ax_{k+i} + Bu_{k+i} \tag{4.12b}$$
$$x_{k+i} \in \mathcal{X} \tag{4.12c}$$
$$u_{k+i} \in \mathcal{U} \tag{4.12d}$$
$$x_{k+N} \in \mathcal{T}(x_r) \tag{4.12e}$$

This formulation is the standard procedure of solving tracking problems in the MPC framework, see, e.g., Lee and Cooley [1997], Mayne et al. [2000], Meadows and Badgwell [1998], Muske and Rawlings [1993], Rao and Rawlings [1999], Rawlings [2000], Rawlings et al. [1994]. In this setup the terminal state constraint set now depends on the setpoint in the way that it shall be an invariant set for the translated system. This will in general lead to a terminal constraint set that has a different size and shape for every steady state setpoint. This is a complicating fact since it could require a recalculation of the terminal constraint set on-line. We will elaborate more on this topic in Chapter 5.

A further extension to the tracking concept is to use a so called *pseudo setpoint* or *pseudo reference* [Rossiter, 2006]. Instead of using the true reference $r$ in (4.11) one introduces a new optimization variable $\bar{r}$ which gives a corresponding steady state and control, $\bar{x}$ and $\bar{u}$, in the optimization problem (4.12), and then penalize the deviation between the desired reference $r$ and the pseudo reference $\bar{r}$ using a positive definite function $\phi(\bar{r} - r)$ in the objective function. By using this pseudo reference, the feasible region of the problem can be increased. The dual mode MPC problem when using a pseudo reference then has the form

$$\begin{aligned}\underset{u_{k+i},x_{k+i},\bar{x}_k,\bar{u}_k,\bar{r}_k}{\text{minimize}} \quad & \sum_{i=0}^{N-1} \ell(x_{k+i} - \bar{x}_k, u_{k+i} - \bar{u}_k) \\ & + \Psi(x_{k+N} - \bar{x}_k) + \phi(\bar{r}_k - r)\end{aligned} \tag{4.13a}$$

subj. to

$$x_{k+i+1} = Ax_{k+i} + Bu_{k+i} \tag{4.13b}$$
$$x_{k+i} \in \mathcal{X} \tag{4.13c}$$
$$u_{k+i} \in \mathcal{U} \tag{4.13d}$$
$$x_{k+N} \in \mathcal{T}(\bar{x}_k) \tag{4.13e}$$
$$\begin{bmatrix} A - I & B \\ C & D \end{bmatrix} \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} = \begin{bmatrix} 0 \\ \bar{r}_k \end{bmatrix} \tag{4.13f}$$

Note that the only constraint on $\bar{r}$ is that it fulfills (4.13f), but since also $\bar{x}$ and $\bar{u}$ are variables in the optimization problem they can be chosen and the only constraint on how they must be chosen is (4.13e). The limiting constraint

is that $\bar{x}$ must be chosen such that the final state at the end of the prediction horizon is in the terminal set, $x_{k+N} \in \mathcal{T}(\bar{x})$. This means that an arbitrary reference, $r$, can not lead to infeasibility since we can always chose $\bar{r}$ such that $x_{k+N} \in \mathcal{T}(\bar{x})$ and the optimization problem remains feasible, but it might be at a high cost. In this way the pseudo reference acts as a prefiltering of the true reference such that feasibility can be guaranteed when changing references.

Both the formulation (4.12) and (4.13) requires a perfect model of the system otherwise there will be a nonzero steady state offset. Additionally, if there are disturbances acting on the system there will also be an offset between the desired output and the achieved output. Therefore one has to introduce integral action in the MPC controller for any practical purposes.

### 4.2.3 Integral control

A standard way of introducing integral control is to augment the system model with new states, $\epsilon$, which is the integral of the tracking errors

$$\epsilon_k = \epsilon_{k-1} + r_k - y_k$$

and then penalize the integral state in the cost function. However this may not be a suitable approach in MPC since, firstly, the computational cost increases as the cube of the state dimension and secondly due to the presence of constraints there is a need for an anti-windup structure in order to avoid performance degradation [Muske and Badgwell, 2002].

Instead the standard procedure in MPC is to augment the system model with a constant disturbance, $d_k$,

$$x_{k+1} = Ax_k + Bu_k + Ed_k \tag{4.14a}$$
$$d_{k+1} = d_k \tag{4.14b}$$
$$y_k = Cx_k \tag{4.14c}$$

and use a disturbance observer to estimate this disturbance. Then the desired steady state is compensated for the disturbance.

$$\begin{bmatrix} A - I & B \\ C & 0 \end{bmatrix} \begin{bmatrix} x_r \\ u_r \end{bmatrix} = \begin{bmatrix} -E\hat{d}_k \\ r \end{bmatrix} \tag{4.15}$$

Many papers have been published related to integral control in MPC and almost all are variants of this setup. One of the pioneering contributions by Muske and Rawlings [1993] consider both input and output constant disturbances and use the Kalman filter to estimate the disturbances. In Meadows and Badgwell [1998] this concept is analyzed in the context of nonlinear MPC for constant output disturbances. Pannocchia and Rawlings [2003] and Pannocchia and Kerrigan [2003] extend the theory to a more general disturbance model and non square systems and Maeder and Morari [2010] extends the theory further to handle non constant disturbances such as sinusoids and ramps.

We will in this thesis only consider the standard setup with a constant

disturbance model and following the derivation in Åkesson and Hagander [2003] we show that the use of a disturbance observer in MPC will, in fact, result in integral action.

For simplicity and clarity of the derivation we consider an integral form of MPC formulation (4.12) with a fixed reference, i.e., the terminal constraint set (4.12e) is a fixed set and can be described as a polytope. Instead of the state prediction used in (4.12) we use the state prediction equation $x_{k+i+1} = Ax_{k+i} + Bu_{k+i} + E\hat{d}_k$ with $x_k = \hat{x}_k$ where $\hat{x}_k$ and $\hat{d}_k$ are the observed state and disturbance. The disturbance observer for the system (4.14) is given by

$$\begin{bmatrix} \hat{x}_{k+1} \\ \hat{d}_{k+1} \end{bmatrix} = \begin{bmatrix} A & E \\ 0 & I \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ \hat{d}_k \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_k + L(y - C\hat{x}_k) \tag{4.16}$$

where $L$ is the observer feedback gain.

We also assume that the steady state and control are given by

$$\begin{bmatrix} A - I & B \\ C & 0 \end{bmatrix} \begin{bmatrix} x_r \\ u_r \end{bmatrix} = \begin{bmatrix} -E\hat{d}_k \\ r \end{bmatrix}$$

$$\begin{bmatrix} x_r \\ u_r \end{bmatrix} = \begin{bmatrix} A - I & B \\ C & 0 \end{bmatrix}^{-1} \begin{bmatrix} -E\hat{d}_k \\ r \end{bmatrix} = \begin{bmatrix} \Pi_{xd} & \Pi_{xr} \\ \Pi_{ud} & \Pi_{ur} \end{bmatrix} \begin{bmatrix} \hat{d}_k \\ r \end{bmatrix} \tag{4.17}$$

By defining $X = \left[x_{k+1}^T,\ x_{k+2}^T,\ \ldots,\ x_{k+N}^T\right]^T$ and $U_k = \left[u_k^T,\ u_{k+1}^T,\ \ldots,\ u_{k+N-1}^T\right]^T$ we can write (4.12) as

$$\underset{U_k}{\text{minimize}} \quad (X - \mathbb{I}x_r)^T \mathcal{Q} (X - \mathbb{I}x_r) + (U_k - \mathbb{I}u_r)^T \mathcal{R} (U_k - \mathbb{I}u_r) \tag{4.18}$$

$$\text{subj. to} \tag{4.19}$$

$$X = \mathcal{A}\hat{x}_k + \mathcal{B}U_k + \mathcal{E}\hat{d}_k \tag{4.20}$$

$$\mathcal{F}_x X \leq \Upsilon_x \tag{4.21}$$

$$\mathcal{F}_u U_k \leq \Upsilon_u \tag{4.22}$$

where the matrices in the objective function are defined as

$$\mathbb{I} = \begin{bmatrix} I \\ I \\ \vdots \\ I \end{bmatrix}, \quad \mathcal{Q} = \begin{bmatrix} Q & & 0 & 0 \\ & Q & & 0 \\ 0 & & \ddots & \\ 0 & 0 & & P \end{bmatrix}, \quad \mathcal{R} = \begin{bmatrix} R & & 0 & 0 \\ & R & & 0 \\ 0 & & \ddots & \\ 0 & 0 & & R \end{bmatrix}$$

and for the system dynamics as

$$\mathcal{A} = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}, \quad \mathcal{B} = \begin{bmatrix} B & 0 & \ldots & 0 \\ AB & B & \ldots & 0 \\ \vdots & & \ddots & \\ A^{N-1}B & A^{N-2}B & \ldots & B \end{bmatrix}, \quad \mathcal{E} = \begin{bmatrix} E \\ AE + E \\ \vdots \\ \sum_{j=0}^{N-1} A^j E \end{bmatrix}$$

The state and control constraint matrices defined as

$$\Upsilon_x = [b_{\mathcal{X}}^T \;\; b_{\mathcal{X}}^T \;\; \dots \;\; b_{\mathcal{T}}^T]^T, \quad \Upsilon_u = [b_{\mathcal{U}}^T \;\; b_{\mathcal{U}}^T \;\; \dots \;\; b_{\mathcal{U}}^T]^T$$

$$\mathcal{F}_x = \begin{bmatrix} F_{\mathcal{X}} & & 0 & 0 \\ & F_{\mathcal{X}} & & 0 \\ 0 & & \ddots & \\ 0 & 0 & & F_{\mathcal{T}} \end{bmatrix}, \quad \mathcal{F}_u = \begin{bmatrix} F_{\mathcal{U}} & & 0 & 0 \\ & F_{\mathcal{U}} & & 0 \\ 0 & & \ddots & \\ 0 & 0 & & F_{\mathcal{U}} \end{bmatrix}$$

We can eliminate the state variables from the above formulation with the use of the state dynamic equation. This results in the following optimization problem

$$\begin{aligned} \underset{U_k}{\text{minimize}} \; & U_k^T \left(\mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R}\right) U_k + 2U_k^T \mathcal{B}^T \mathcal{Q} \mathcal{A} \hat{x}_k + 2U_k^T \mathcal{B}^T \mathcal{Q} \mathcal{E} \hat{d}_k \\ & - 2U_k^T \mathcal{B}^T \mathcal{Q} \mathbb{I} x_r - 2U_k^T \mathcal{R}^T \mathbb{I} u_r + \Phi(\hat{x}_k, \hat{d}_k, x_r, u_r) \end{aligned}$$

subj. to

$$\begin{bmatrix} \mathcal{F}_x \mathcal{B} \\ \mathcal{F}_u \end{bmatrix} U_k \leq \begin{bmatrix} \Upsilon_x - \mathcal{F}_x \mathcal{A} \hat{x}_k - \mathcal{F}_x \mathcal{E} \hat{d}_k \\ \Upsilon_u \end{bmatrix}$$

where we have collected all the terms of the objective function that are independent of $u_{k+i}$ in $\Phi(\cdot)$. From Section 2.1.2 we know that at the optimum it must hold that the gradient of the Lagrangian is equal to zero, hence form the Lagrangian and differentiate it w.r.t. $U_k$.

$$\begin{aligned} \frac{d\mathcal{L}}{d\, U_k} = & \; 2\left(\mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R}\right) U_k^* + 2\mathcal{B}^T \mathcal{Q} \mathcal{A} \hat{x}_k + 2\mathcal{B}^T \mathcal{Q} \mathcal{E} \hat{d}_k \\ & - 2\mathcal{B}^T \mathcal{Q} \mathbb{I} x_r - 2\mathcal{R}^T \mathbb{I} u_r + \begin{bmatrix} \mathcal{F}_u \\ \mathcal{F}_x \mathcal{B} \end{bmatrix}^T \Lambda^* = 0 \end{aligned}$$

From this we can easily solve for $U_k$

$$U_k^* = -\left(\mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R}\right)^{-1} \left( \mathcal{B}^T \mathcal{Q} \left( \mathcal{A} \hat{x}_k + \mathcal{E} \hat{d}_k - \mathbb{I} x_r \right) - \mathcal{R}^T \mathbb{I} u_r + \frac{1}{2} \begin{bmatrix} \mathcal{F}_u \\ \mathcal{F}_x \mathcal{B} \end{bmatrix}^T \Lambda^* \right)$$

and the optimal control, $u_k^*$, is defined by the first row of the equation as

$$u_k^* = -K_x \hat{x}_k - K_d \hat{d}_k + K_{x_r} x_r + K_{u_r} u_r - K_\lambda \lambda^* \tag{4.23}$$

for an appropriate definition of the matrices, $K_{(\cdot)}$. Using the relation (4.17) we can write $u_k^*$ as

$$\begin{aligned} u_k^* &= -K_x \hat{x}_k - \underbrace{\left(K_d - K_{x_r} \Pi_{xd} - K_{u_r} \Pi_{ud}\right)}_{\tilde{K}_d} \hat{d}_k + \underbrace{\left(K_{x_r} \Pi_{xr} + K_{u_r} \Pi_{ur}\right)}_{\tilde{K}_r} r - K_\lambda \lambda^* \\ &= -K_x \hat{x}_k - \tilde{K}_d \hat{d}_k + \tilde{K}_r r - K_\lambda \lambda^* \end{aligned} \tag{4.24}$$

Inserting (4.24) into the observer equations (4.16) we obtain

$$\begin{bmatrix}\hat{x}_{k+1}\\ \hat{d}_{k+1}\end{bmatrix} = \begin{bmatrix}A - BK_x - L_xC & E - B\tilde{K}_d\\ -L_dC & I\end{bmatrix}\begin{bmatrix}\hat{x}_k\\ \hat{d}_k\end{bmatrix} + \begin{bmatrix}B\tilde{K}_r\\ 0\end{bmatrix} r + \begin{bmatrix}L_x\\ L_d\end{bmatrix} y_k + \begin{bmatrix}BK_\lambda\\ 0\end{bmatrix}\lambda^*$$
$$\triangleq \begin{bmatrix}\tilde{A}_{11} & \tilde{A}_{12}\\ -\tilde{A}_{21} & I\end{bmatrix}\begin{bmatrix}\hat{x}_k\\ \hat{d}_k\end{bmatrix} + \begin{bmatrix}B\tilde{K}_r\\ 0\end{bmatrix} r + \begin{bmatrix}L_x\\ L_d\end{bmatrix} y_k + \begin{bmatrix}BK_\lambda\\ 0\end{bmatrix}\lambda^*$$

Rewriting the observer equations into the z-transform domain and into transfer matrix form yield

$$\begin{bmatrix}\hat{x}(z)\\ \hat{d}(z)\end{bmatrix} = \begin{bmatrix}z^{-1}I - \tilde{A}_{11} & -\tilde{A}_{12}\\ \tilde{A}_{21} & \frac{1-z}{z}I\end{bmatrix}^{-1}\left(\begin{bmatrix}B\tilde{K}_r\\ 0\end{bmatrix} r(z) + \begin{bmatrix}L_x\\ L_d\end{bmatrix} y(z) + \begin{bmatrix}BK_\lambda\\ 0\end{bmatrix}\lambda^*\right) \quad (4.25)$$

and the matrix inverse is

$$\begin{bmatrix}z^{-1}I - \tilde{A}_{11} & -\tilde{A}_{12}\\ \tilde{A}_{21} & \frac{1-z}{z}I\end{bmatrix}^{-1} = \begin{bmatrix}\Psi(z) & \frac{z}{1-z}\Psi(z)\tilde{A}_{12}\\ -\frac{z}{1-z}\tilde{A}_{21}\Psi(z) & \frac{z}{1-z}\left(I - \frac{z}{1-z}\tilde{A}_{21}\Psi(z)\tilde{A}_{12}\right)\end{bmatrix}$$

where, $\Psi(z) = z^{-1}I - \tilde{A}_{11} + \frac{z}{1-z}\tilde{A}_{12}\tilde{A}_{21}$, denotes the Schur complement to $\frac{1-z}{z}I$.

Inserting (4.25) into the z-transform of (4.24) we finally obtain

$$\begin{aligned} u(z) &= -K_x\hat{x}(z) - \tilde{K}_d\hat{d}(z) + \tilde{K}_r r(z) - K_\lambda\lambda^*\\ &= -K_x\Psi(z)L_x y(z) - \left(K_x\Psi(z)B - I\right)\tilde{K}_r r(z)\\ &\quad + \frac{z}{1-z}\Phi(z)(\Gamma(z)r(z) - y(z))\\ &\quad - \left(K_x\Psi(z)B - \frac{z}{1-z}\tilde{K}_d\tilde{A}_{21}\Psi(z)B - I\right)K_\lambda\lambda^* \end{aligned} \quad (4.26)$$

with

$$\Phi(z) = K_x\Psi(z)\tilde{A}_{12}L_d - \tilde{K}_d\tilde{A}_{21}\Psi(z)L_x + \tilde{K}_d - \frac{z}{1-z}\tilde{K}_d\tilde{A}_{21}\Psi(z)\tilde{A}_{12}L_d$$

$$\Gamma(z) = \Phi^{-1}(z)\tilde{K}_d\tilde{A}_{21}\Psi(z)B\tilde{K}_r$$

Equation (4.26) shows that the MPC setup with a disturbance observer for a constant disturbance will result in an output feedback term, $-K_x(z)\Psi(z)L_xy(z)$, a reference feedforward term, $-\left(K_x(z)\Psi(z)\tilde{B}(z) - \tilde{K}_r(z)\right)r(z)$, a term depending on the optimal dual variables, $-\left(K_x\Psi(z)B - \frac{z}{1-z}\tilde{K}_d\tilde{A}_{21}\Psi(z)B - I\right)K_\lambda\lambda^*$, and an integral term of the output error, $\frac{z}{1-z}\Phi(z)(\Gamma(z)r(z) - y(z))$. It is worth noting that if no constraints are active then the complementary slackness condition (2.9) implies that the optimal dual variables, $\lambda^*$, are zero and hence that term has no affect on the control signal (as one would expect).

In Figure 4.2 we apply this integral control algorithm to the pitch dynamics of an aircraft as defined in Section 3.2, but here subject to a constant state disturbance. The figure clearly shows the benefits of adding integral control since without the integral action there is a large offset from the reference signal

which is completely eliminated with the use of integral control.

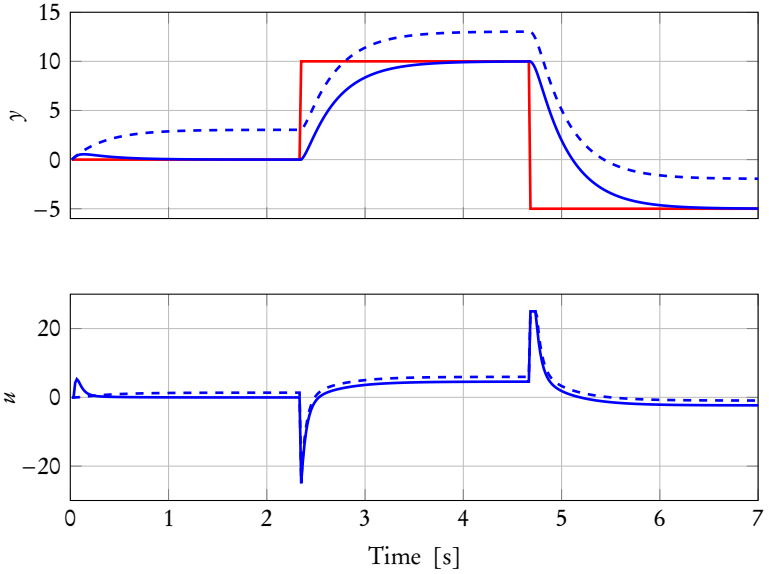


**Figure 4.2.** The step response of a system subject to a constant state disturbance when controlled with an MPC controller without integral action and with integral action. The dashed lines are the output and control signal of an MPC controller without integral action and the solid lines are the output and control signal when integral action is added.

Note that we have not discussed anything about the stability properties of this output feedback MPC algorithm. As a matter of fact, the well known *principle of separation* does not hold and a separate state estimator and feedback design does not necessary stabilize nonlinear and constrained systems. We will not go into detail on the stabilizing properties of output feedback MPC in this thesis, instead we refer to Rawlings and Mayne [2009] for a more in depth treatment.

### 4.2.4 Slack variables

Due to the iterative nature of the MPC method it can happen, due to, e.g., model errors or disturbances, that at some point in time the optimization problem becomes infeasible. There exist several ways to handle this, one ad hoc way is to simply use the next control signal, $u_{k+1}$, from the optimal sequence in the previous time step; but there is no guarantee that the problem becomes feasible again at a later point in time.

A better and more systematic way to handle infeasibility issues is to add an extra optimization variable, a so called *slack variable*, $\varepsilon$, to the problem. The slack variable is a non-negative vector that is added to the right hand side of

the state constraints

$$F_{\mathcal{X}} x \leq b_{\mathcal{X}} + \varepsilon$$

The effect of adding the slack variable to the right hand side of the inequality constraints is that if an element of the slack variable is larger than zero then the corresponding constraint is relaxed, i.e., feasibility can be regained.

Usually, the slack variables are only added to the state constraints and not to the input constraint [Maciejowski, 2002], since the input constraints usually are real hard constraints with physical limits, hence it makes no sense to relax those constraints. Note however that in theory one could also add slack to the input constraints if it makes any sense, e.g., if the input signal limits have some safety margin to the real physical limit which one do not want to break unless it is extremely important.

In addition to adding the slack to the constraints it is also added in the cost function with some positive definite function, $\varphi(\varepsilon)$, that penalizes the constraint violations. In de Oliveira and Biegler [1994] it is suggested to use a quadratic cost function of the form

$$\varphi(\varepsilon) = \rho \, ||\varepsilon||^2$$

The drawback with this type of penalty function is that if any constraints are active in the optimum, then a quadratic penalty on the slack variable will always cause the constraints to be violated to some small extent. This comes from the fact that the increased cost from the penalty function is of order $\mathcal{O}(\varepsilon^2)$ and smaller than the reduced cost from the other parts of the cost function, which is of order $\mathcal{O}(\varepsilon)$, for small values of $\varepsilon$ [Maciejowski, 2002]. If instead a linear penalty function is used then the increase in cost is also of order $\mathcal{O}(\varepsilon)$ and if $\rho$ is large enough, then the constraints will only be violated if there does not exist a feasible solution to the problem with slack variables equal to zero [Maciejowski, 2002].

This is known as an *exact penalty* but the question remains how to chose the gain $\rho$. Unfortunately, in order to have exact penalty the choice of gain, $\rho$, depends on the optimal dual variables (which are not known until the problem is solved). Fletcher [1987] shows that the gain must be larger than the dual norm of the Lagrange multipliers (the dual variables).

$$\rho > ||\lambda^*||_D$$

In order to overcome this problem one has to estimate a bound on the dual variables and in Kerrigan and Maciejowski [2000] the authors derive a fixed lower bound on the gain.

Finally the resulting MPC problem with integral action, pseudo setpoint and

slack variables can be formulated as

$$\underset{u_{k+i},x_{k+i},\bar{x}_k,\bar{u}_k,\bar{r}_k,\varepsilon}{\text{minimize}} \quad \sum_{i=0}^{N-1} \ell(x_{k+i} - \bar{x}_k, u_{k+i} - \bar{u}_k) + \Psi(x_{k+N} - \bar{x}_k) + \phi(\bar{r}_k - r) + \rho \, ||\varepsilon||_\infty \tag{4.27a}$$

subj. to

$$x_{k+i+1} = Ax_{k+i} + Bu_{k+i} + E\hat{d}_k \tag{4.27b}$$
$$F_{\mathcal{X}} x_{k+i} \leq b_{\mathcal{X}} + \varepsilon \tag{4.27c}$$
$$F_{\mathcal{U}} u_{k+i} \leq b_{\mathcal{U}} \tag{4.27d}$$
$$F_{\mathcal{T}} x_{k+N} \leq b_{\mathcal{T}} \tag{4.27e}$$
$$\begin{bmatrix} A - I & B \\ C & D \end{bmatrix} \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} = \begin{bmatrix} -E\hat{d}_k \\ \bar{r}_k \end{bmatrix} \tag{4.27f}$$

where $x_k$, $r$ and $\hat{d}_k$ are the input variables and $\hat{d}_k$ is estimated with a disturbance observer like (4.16). The applied control law is $\kappa(x_k) = u_k^*$.

### 4.2.5 The explicit solution

For many years it has been the general conception that an explicit solution to the constrained optimal control problem (4.2) is extremely difficult to calculate [Mayne et al., 2000]. However in the last two decades this notion has been overthrown by what is called *Explicit Model Predictive Control*.

The development of explicit MPC has been motivated mainly by the need for MPC algorithms to control systems with very fast dynamics and a need for algorithms to be implemented on low cost hardware, where an online optimization is not possible. However yet another aspect is that of verifiability of the algorithm for safety critical applications; since in the recent years very fast online algorithms have been developed, verifiability might be the most important aspect today for using explicit MPC in aeronautical applications.

Explicit MPC builds on the concept of multi-parametric programming, which has been a research topic within the optimization community since the 1960's [Gal, 1980], but it was not until the papers of Bemporad et al. [2002a] and Bemporad et al. [2002b] that these techniques became widely known within the MPC community.

A multi-parametric program can in its general form be written as

$$z^*(\theta) = \arg\min \; f(z, \theta)$$

$$\text{subj. to} \; g(z, \theta) \leq 0$$

where $z$ is the optimization variable and $\theta$ is a vector of parameters. Both the objective function and the constraint function can depend on the parameter

vector and of course also the solution $z^*(\theta)$ is dependent on the parameter values.

We will in this section derive the explicit formulation for the quadratic program (4.5) where the constraint sets $\mathcal{X}$, $\mathcal{U}$ and $\mathcal{T}$ are polytopic sets.

In the same manner as we did in Section 4.2.3 we can rewrite the basic MPC formulation (4.5) by defining the concatenated vectors

$$X = \begin{bmatrix} x_{k+1}^T & x_{k+2}^T & \dots & x_{k+N}^T \end{bmatrix}^T$$

and

$$U_k = \begin{bmatrix} u_k^T & u_{k+1}^T & \dots & u_{k+N-1}^T \end{bmatrix}^T$$

which results in

$$\underset{U_k}{\text{minimize}} \; X^T \mathcal{Q} X + U_k^T \mathcal{R} U_k \tag{4.28a}$$

subj. to

$$X = \mathcal{A} x_k + \mathcal{B} U_k \tag{4.28b}$$

$$\mathcal{F}_x X \leq \Upsilon_x \tag{4.28c}$$

$$\mathcal{F}_u U_k \leq \Upsilon_u \tag{4.28d}$$

From this we can again use the state dynamics equation (4.28b) to eliminate the state variable from (4.28) and the resulting optimization problem is a multi-parametric problem

$$\underset{U_k}{\text{minimize}} \; U_k^T \left( \mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R} \right) U_k + 2 U_k^T \left( \mathcal{B}^T \mathcal{Q} \mathcal{A} \right) x_k + x_k^T \left( \mathcal{A}^T \mathcal{Q} \mathcal{A} \right) x_k \tag{4.29a}$$

$$\text{subj. to} \begin{bmatrix} \mathcal{F}_u \\ \mathcal{F}_x \mathcal{B} \end{bmatrix} U_k \leq \begin{bmatrix} \Upsilon_u \\ \Upsilon_x - \mathcal{F}_x \mathcal{A} x_k \end{bmatrix} \tag{4.29b}$$

where $U_k$ is the optimization variable and $x_k$ the parameter.

If we introduce the variable

$$z = U_k + \left( \mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R} \right)^{-1} \mathcal{B}^T \mathcal{Q} \mathcal{A} x_k$$

we can rewrite (4.29) to

$$\underset{z}{\text{minimize}} \; z^T H z \tag{4.30a}$$

$$\text{subj. to} \quad M z \leq N + S x_k \tag{4.30b}$$

with

$$H = \mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R}, \quad M = \begin{bmatrix} \mathcal{F}_u \\ \mathcal{F}_x \mathcal{B} \end{bmatrix}$$

$$N = \begin{bmatrix} \Upsilon_u \\ \Upsilon_x \end{bmatrix}, \quad S = \begin{bmatrix} \mathcal{F}_u \left( \mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R} \right)^{-1} \mathcal{B}^T \mathcal{Q} \mathcal{A} \\ -\mathcal{F}_x \mathcal{A} + \mathcal{F}_x \mathcal{B} \left( \mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R} \right)^{-1} \mathcal{B}^T \mathcal{Q} \mathcal{A} \end{bmatrix}$$

where now only the right hand side of the constraints depends on the parameter.

Given a value on the parameter $x_k$ the solution to the multi-parametric QP (4.30) is given by the KKT conditions

$$2Hz + M^T\lambda = 0 \tag{4.31a}$$
$$Mz - (N + Sx_k) \leq 0 \tag{4.31b}$$
$$\lambda \geq 0 \tag{4.31c}$$
$$\lambda_i\Big(M_i z - (N + Sx_k)_i\Big) = 0 \tag{4.31d}$$

where the subscript $i$ denotes the i:th row of the matrices. From (4.31a) we can solve for $z$

$$z^* = -H^{-1}M^T\lambda$$

which if we insert it into the complementary slackness condition (4.31d) gives

$$\lambda\Big(MH^{-1}M^T\lambda + (N + Sx_k)\Big) = 0$$

From Section 2.1.2 we know that for all active constraints it must hold that

$$M_{\mathcal{A}}H^{-1}M_{\mathcal{A}}^T\lambda_{\mathcal{A}} + N_{\mathcal{A}} + S_{\mathcal{A}}x_k = 0$$

where the subscript $\mathcal{A}$ denotes the part of the matrices for which the corresponding constraint is active. From this equation we can determine the dual variable and substitute it into the solution for, $z^*(x_k)$, yielding

$$z^* = H^{-1}M^T\left(M_{\mathcal{A}}H^{-1}M_{\mathcal{A}}^T\right)^{-1}(N_{\mathcal{A}} + S_{\mathcal{A}}x_k) \tag{4.32}$$

From (4.32) we observe that the optimal solution $z^*(x_k)$ to the KKT conditions (4.31) is an affine function of the parameter $x_k$

$$z^*(x_k) = F_i x_k + g_i$$

where $F_i$ and $g_i$ depends on the set of active constraints and hence the solution $z^*(x_k)$ is a different affine function depending on the set of active constraints. The set of the active constraints, $\mathcal{X}_i$, can easily be determined if we note that the optimal solution $z^*(x_k)$ must fulfill the primal feasibility constraint (4.31b), i.e.,

$$MH^{-1}M^T\left(M_{\mathcal{A}}H^{-1}M_{\mathcal{A}}^T\right)^{-1}(N_{\mathcal{A}} + S_{\mathcal{A}}x_k) \leq N + Sx_k \tag{4.33}$$

The inequality constraint (4.33) specifies the polytopic subset, $\mathcal{X}_i$, of the state space where the optimal solution, $z^*(x_k)$, is valid.

It can be shown that the optimal solution $z^*(x_k)$ is, in fact, a continuous piecewise affine function (in the sense of definition 4.3) of the parameter $x_k$ [Bemporad et al., 2002b]. Each affine function valid in a different polytopic subset, $\mathcal{X}_i$, of the state space where the active constraints does not change.

**Definition 4.3.** A function $z^*(x)$ is said to be piecewise affine if it is possible

to partition the polyhedral state constraint set $\mathcal{X}$ into convex polyhedral regions $\mathcal{X}_i$ such that $z^*(x) = F_i x + g_i \ \forall \ x \in \mathcal{X}_i$

The calculations of the explicit controller is an iterative procedure where the entire feasible set is partitioned into convex polyhedral regions, each with an affine feedback solution. Here we will only outline the main structure of one such algorithm and refer to Baotic [2005] for the details. Once an initial $x_k$ has been chosen and the affine solution and the polytopic subset, $\mathcal{X}_i$, has been calculated, then the remainder of the state space can be partitioned into convex regions, $\mathcal{P}_j$. In each convex region a new $x_k$ is chosen and a new solution, $z^*(x_k)$ and the polytope of active constraints, $\mathcal{X}_i$ are calculated. The algorithm gradually explores the different regions $\mathcal{P}_j$, finding new partitions, $\mathcal{X}_i$, and dividing the remaining region into new smaller regions. This procedure continues until all regions have been divided into partitions and no unexplored regions remain. Then the entire feasible set will be divided into convex polyhedral partitions, see Figure 4.3.

We summarize the above discussion in Algorithm 1.

**Algorithm 1** Offline calculations of the explicit MPC feedback solution

1: Given an initial set of constraints (in the first iteration the state constraint set $\mathcal{X}$), select an initial state $x_k$ belonging to the set, for example the Chebychev center (2.12) of the set [Bemporad et al., 2002b].
2: Solve the KKT equations 4.31.
3: Determine the set of active constraints.
4: Calculate $F_i$ and $g_i$ from (4.32).
5: Determine the set $\mathcal{X}_i$ from (4.33).
6: Partition the remainder of the state constraint set into non overlapping convex regions $\mathcal{P}_j$.
7: Repeat from 1, now with the set $\mathcal{P}_j$ as initial set of constraints.

Note that with the procedure for exploring the state space by dividing it into regions $\mathcal{P}_j$, which has been described in both Dua and Pistikopoulos [2001] and Bemporad et al. [2002b] it can happen, due to the nature of the algorithm, that several adjacent partitions have the same feedback solution. If the partitions constitute a convex set, then they can be merged into one partition, reducing the complexity of the controller, see Figure 4.3.

The online complexity of the explicit solution is mainly affected by the task of finding the partition, $\mathcal{X}_i$, in which the current state, $x_k$, is located (the so called *point location problem*) and hence dependent on the number of partitions, $N_{\mathcal{X}}$, that are required to describe the solution. The runtime complexity is in worst case $\mathcal{O}(N_{\mathcal{X}})$ if a straightforward simple search routine is implemented but with advanced search algorithms the runtime can be reduced to $\mathcal{O}(\log_2 N_{\mathcal{X}})$ [Jones et al., 2006, Kvasnica, 2009].

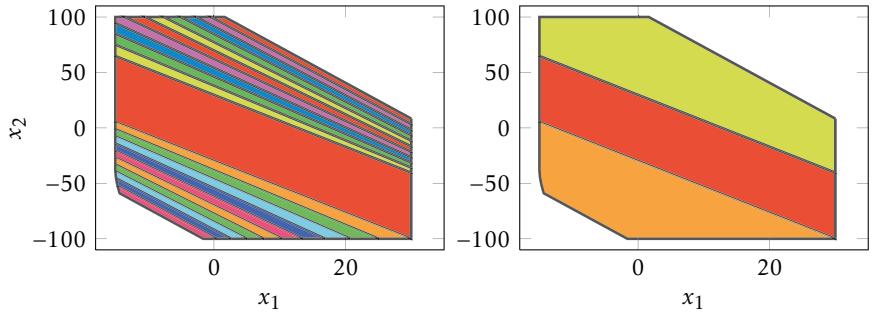


**Figure 4.3.** In the left axis the total number of partitions, $\mathcal{X}_i$, generated by the algorithm and in the right axis the number of partitions after reduction.

Since the number of state space partitions so greatly affect the applicability of explicit MPC, a large amount of research has been performed on the task of defining a simple state space partition structure in order to reduce the complexity in the point location problem, see, e.g., Genuit et al. [2011], Raimondo et al. [2012], Rubagotti et al. [2012].

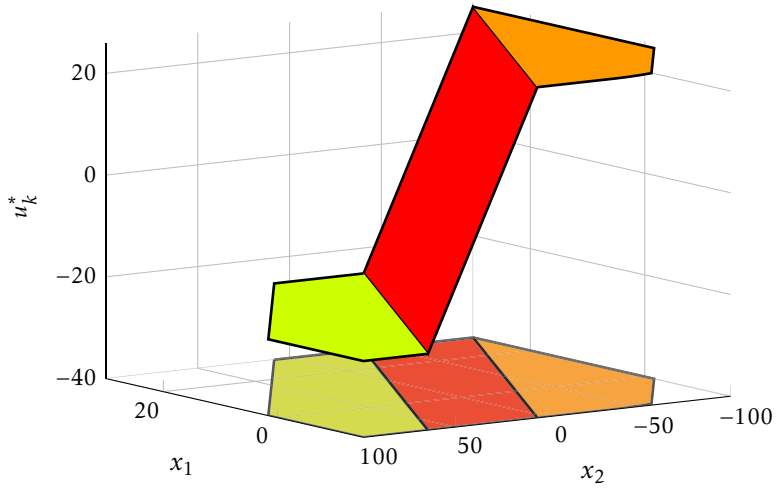


**Figure 4.4.** The optimal explicit MPC feedback solution plotted over the polytopic regions where each affine solution is valid.

In Figure 4.3 and 4.4 we have used the pitch dynamics of from Section 3.2 to calculate a simple explicit MPC controller for stabilizing the unstable dynamics around the origin. In Figure 4.4 we can see that the optimal feedback solution is a piecewise affine function with three regions. One region where the solution is the discrete LQ solution and two regions where it is the saturated control input.

In Chapter 5 we develop a new MPC formulation for reference tracking purposes, which has the property of greatly reducing the number of state space partitions needed to describe the explicit solution.

## 4.3 Nonlinear MPC

Even though all MPC formulations, because of the constraints, result in nonlinear controllers, the term nonlinear MPC usually refers to the case where the system dynamics is nonlinear

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

and the MPC problem formulation (4.5) now looks like

$$\underset{u_{k+i}, x_{k+i}}{\text{minimize}} \quad \sum_{i=0}^{N-1} \ell(x_{k+i}, u_{k+i}) + \Psi(x_{k+N}) \tag{4.34a}$$

$$\text{subj. to}$$

$$x_{k+i+1} = f(x_{k+i}, u_{k+i}) \tag{4.34b}$$

$$x_{k+i} \in \mathcal{X} \tag{4.34c}$$

$$u_{k+i} \in \mathcal{U} \tag{4.34d}$$

$$x_{k+N} \in \mathcal{T} \tag{4.34e}$$

Even though (4.34) looks very much like (4.5) and the analysis in Section 4.2.1, such as Theorem 4.2, can be extended to hold also for nonlinear systems there are some very important subtle differences.

In general it is extremely difficult to find the terminal state penalty, $\Psi(x_{k+N})$, and constraint set, $\mathcal{T}$, such that the problem corresponds to the infinite horizon optimal control problem. In the linear case this is relatively straightforward and a quadratic penalty term can be used with the unconstrained LQ solution, $P$, as penalty.

Instead several different approaches on finding good approximate solutions to the infinite horizon problem have been proposed in the literature. Many of the methods, their stability properties, performance and implementation are analyzed in the easily accessible paper by Nicolao et al. [2000].

One popular approach proposed by, e.g., Chen and Allgöwer [1998], Michalska and Mayne [1993] and Magni et al. [2001], is to use a linear controller that is stabilizing the linearization of the system around the origin, to define terminal state constraint and cost. Another approach found in Primbs et al. [1999] and Jadbabaie et al. [2001] uses the theory of Control Lyapunov Functions to define an appropriate terminal cost function. Nicolao et al. [1996, 1998] propose to use a nonquadratic terminal state cost which is infinite outside an implicit terminal region, forcing the terminal state into a region where a linear state feedback stabilizes the system. In the paper by De Oliveira Kothare and Morari [2000] the authors introduce *contractive constraints* that forces the norm

of the state at the prediction horizon to be smaller than the norm of the current state. Using this the authors can show, under feasibility assumptions, that the closed loop system is exponentially stable.

Other relevant papers analyzing stability of nonlinear MPC are Henson [1998], Mayne [2000], Mayne and Michalska [1990], Michalska [1997], Morari and Lee [1999] and Fontes [2001]. In Scokaert et al. [1999] two suboptimal MPC algorithms are analyzed which can guarantee stability even in the absence of an optimal solution. The main benefit of the proposed suboptimal schemes is the low computational complexity compared to other algorithms.

Regardless which of the above mentioned stabilizing algorithms one consider, the fact that the system dynamics is nonlinear makes the MPC formulation (4.34) a nonconvex program. In general, nonconvex programs are much harder to solve than the convex program that a linear dynamics and quadratic cost function results in [Boyd and Vandenberghe, 2004]. In the survey of Cannon [2004] several different techniques to solve the nonlinear MPC problem in a computational efficient way are reviewed. These methods include *Sequential Quadratic Programming*, (SQP), *Euler-Lagrange-* and *Hamilton-Jacobi-Bellman* approaches as well as *Cost and constraint approximation*.

The SQP method tries to find a solution for (4.34) by solving a sequence of approximations defined as quadratic optimization problems. The solution to each QP gives a search direction for the original problem (4.34). The main drawback with the SQP method is that the sequence of iterates can be extremely long and secondly that there are no guarantees that the sequence converges to the global minimum.

The Euler-Lagrange and Hamiltion-Jacobi-Bellman techniques circumvent the troublesome task of solving the nonconvex optimization problem by viewing the nonlinear MPC problem formulation in the light of optimal control. The Euler-Lagrange method numerically solves the two point boundary value problem that arise from Pontryagin's Maximum Principle and the Hamilton-Jacobi-Bellman approach tries to numerically solve the Hamilton-Jacobi-Bellman partial differential equation. These techniques either lack any stability guarantees or are extremely computational expensive.

Another approach to reduce the computational burden is to approximate the cost and constraint functions such that a convex program can be solved instead. This approach of course yields a suboptimal solution to the original problem (4.34) and for some approximation schemes not even recursive feasibility can be guaranteed.

One fairly popular approach to remove the nonlinearity of the system dynamics is to first design a linearizing inner feedback loop and then add a linear MPC controller for the linearized system. This alternative also has some major drawbacks and in Chapter 6 we investigate this approach further.

# 5

# A low complexity reference tracking MPC algorithm

Among the many different formulations of MPC with guaranteed stability, one that has attracted much attention is the formulation with a terminal cost and terminal constraint set, i.e., the dual mode formulation. Despite its popularity only little has been published concerning stability properties for reference tracking applications. In this chapter we build on the dual mode formulation of MPC and our goal is to make minimal changes to this framework, for the linear polytopic case, in order to develop a flexible reference tracking algorithm with guaranteed stability and low complexity, which is intuitive and easily understood.

The main idea is to introduce a scaling variable that dynamically scales the terminal constraint set and therefore allows it to be centered around an arbitrary setpoint without violating the stability conditions. The main benefit of the algorithm is reduced complexity of the resulting QP compared to other state of art methods without loosing performance.

This chapter is based on the journal paper:

> Daniel Simon, Johan Löfberg, and Torkel Glad. Reference Tracking MPC using Dynamic Terminal Set Transformation. *IEEE Transactions on Automatic Control*, 59(10):2790–2795, 2014.

which is an extension of the work previously published in the conference paper

> Daniel Simon, Johan Löfberg, and Torkel Glad. Reference tracking MPC using terminal set scaling. In *51st IEEE Conference on Decision and Control (CDC)*, pages 4543–4548, dec 2012.

and in this chapter we have also added simulation results which were presented

on the IFAC world congress 2014, as an extended abstract.

## 5.1 Introduction

As described in Section 4.2.2 it is often argued that with a simple change of coordinates the origin can represent any suitable setpoint for the controller. A pragmatic approach to implement the reference tracking case would therefore be to apply the standard MPC scheme (4.5) on the translated system, i.e., in the original coordinate system, penalizing deviations from the steady state setpoint.

However for the dual mode formulation the possibility to guarantee recursive feasibility and stability when tracking a reference is dependent on the possibility to guarantee that the terminal set, which depends on the chosen setpoint, still is a valid positively invariant set. In order to guarantee stability it must hold that the translated terminal set still fulfills the conditions (1) and (2) in Theorem 4.2. What could easily happen if a simple change of coordinates is used is that $\mathcal{T}(x_r) \nsubseteq \mathcal{X}$, i.e., a translation of $\mathcal{T}$ moves parts of it outside $\mathcal{X}$, see Figure 5.1, thus invalidating any claim of positive invariance (and similarly w.r.t to control constraints for the nominal controller in $\mathcal{T}(x_r)$).

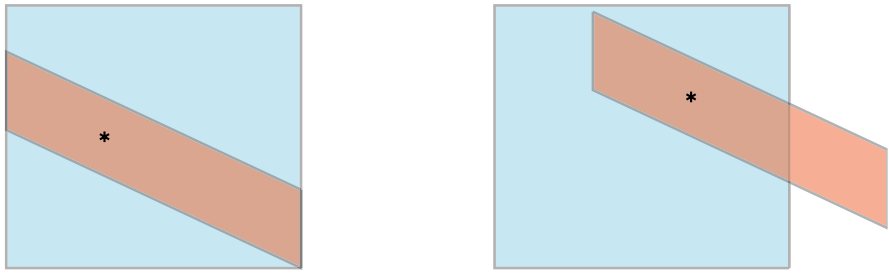

**Figure 5.1.** Example of a terminal set $\mathcal{T}$ (dark shaded polytope) calculated with the setpoint at the origin (marked with a star) and the same terminal set shifted to a new setpoint $x_r$. When translating the terminal set, parts of it may leave the set $\mathcal{X}$ (the square polytope).

Common assumptions when proving stability of linear MPC algorithms for tracking applications are to assume that the terminal constraint set is small enough and that the desired setpoints are located far into the interior of the feasible set such that the translated terminal constraint set still is positively invariant. This is a major drawback since many applications have optimal operating points on or close to the border of the feasible set.

To overcome the problem a straightforward way would be to recalculate the terminal constraint set online based on the current setpoint, see Figure 5.2, which is implicitly assumed in, e.g., Rao and Rawlings [1999], where the

authors develop a method to handle active constraints at setpoints. To recalculate the terminal constraint set can however require complex calculations to be performed online and might thus not be suitable for systems where the setpoint changes often.

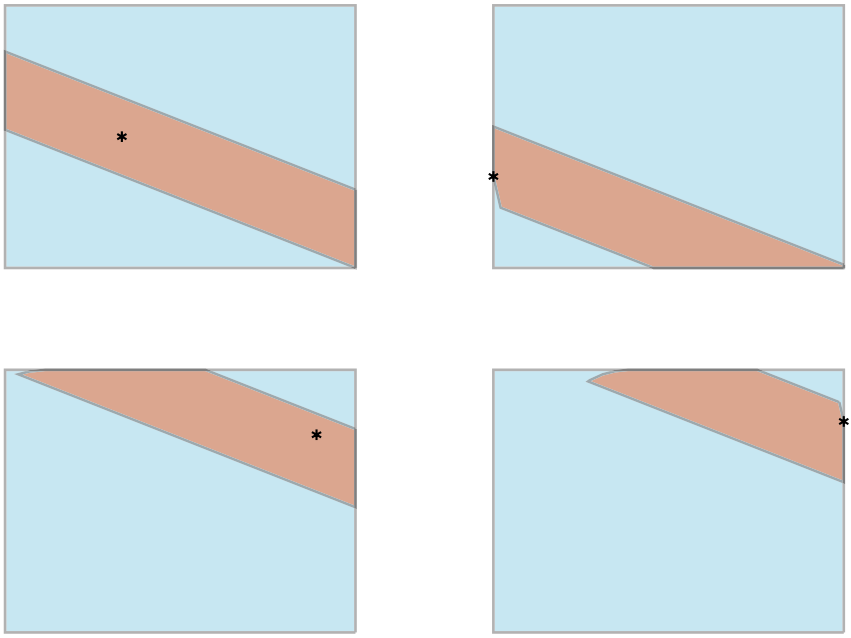

**Figure 5.2.** Example of terminal state constraint set recalculation for different setpoints. Upper left axis is the terminal set calculated with the origin as the setpoint (marked with a star). The three other axis show the terminal constraint set calculated for three different setpoints other than the origin.

A far better approach has been proposed in different forms by several authors Chisci and Zappa [2003], Limon et al. [2008] and Ferramosca et al. [2009]. In Chisci and Zappa [2003] the authors augment the system with a new constant state which corresponds to the reference signal providing a terminal constraint set in the higher dimension $(x, r)$ which contains the whole feasible equilibrium set for any given reference, i.e., the terminal constraint set is a fixed set which can be pre-calculated. Instead of using the reference as augmented state the authors of Ferramosca et al. [2009], Limon et al. [2008] introduce a new optimization variable, $\theta$, which spans the null space of steady state equation (4.11b). The resulting controller guarantees feasibility and stability and has a larger domain of attraction than the standard MPC tracking controller.

The main drawback with these augmented state controllers is that they can be relatively complex even for small systems, making any explicit implementations impractical. This motivates an extension of the theory which is developed in

detail in the remaining sections of this chapter. We develop a reference tracking algorithm with a modified terminal constraint set which can guarantee stability for arbitrary setpoints in the entire feasible set. The developed algorithm has the potential to be much simpler than existing methods.

## 5.2 The proposed controller

We will start with the MPC formulation (4.13) and in this section derive a high level representation of the proposed controller.

Instead of augmenting the system with a new state as in Chisci and Zappa [2003], Limon et al. [2008] the main concept here is to introduce an extra optimization variable, $\lambda_k$, which scales the terminal set, $\mathcal{T}$. The scaling allows us to move the terminal constraint set to an arbitrary setpoint, $\bar{x}_k$, since the terminal set can be scaled down to a single point, i.e., $\mathcal{T} = \bar{x}_k$, eliminating the need for any online recalculation of $\mathcal{T}$ as in Rao and Rawlings [1999] and the terminal constraint set can possibly be far simpler than the terminal set for an augmented system.

The proposed terminal state constraint is a scaled and translated version of the terminal constraint set (4.4f)

$$x_{k+N} \in \lambda_k \mathcal{T}(\bar{x}_k) \tag{5.1}$$

where $\lambda_k$ is a non-negative scalar. The constraints on how $\lambda_k$ can be chosen are such that the conditions of Theorem 4.2 hold, i.e., the terminal set must be positively invariant (which we will argue that it is in Lemma 5.5) and the state and control constraints must be satisfied in $\mathcal{T}$. To ensure this the following constraints must be added

$$\lambda_k \mathcal{T}(\bar{x}_k) \subseteq \mathcal{X} \tag{5.2a}$$

$$\kappa(x, \bar{x}_k, \bar{u}_k) \in \mathcal{U} \quad \forall\, x \in \lambda_k \mathcal{T}(\bar{x}_k) \tag{5.2b}$$

The first constraint states that the scaled and translated terminal set is a subset of the feasible state space, i.e., state constraints are satisfied in $\lambda_k \mathcal{T}(\bar{x}_k)$, and the second constraint states that the predefined stabilizing controller $\kappa(\cdot)$ fulfills the control constraints for all $x$ in $\lambda_k \mathcal{T}(\bar{x}_k)$.

Note that $\lambda_k$ is not necessarily less than 1, it might actually enlarge the terminal set if this is possible with respect to the conditions of Theorem 4.2. Let us illustrate this with an example.

**Example 5.1**

Consider a discrete time system (artificially constructed to illustrate the behavior with terminal set enlargement) with $A$ and $B$ matrices

$$A = \begin{bmatrix} 0.9 & 0.5 \\ 0 & -0.8 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0.6 \end{bmatrix}$$

and with the state constraints

$$\mathcal{X} = \left\{ x \mid \|x - p\|_1 \leq 10, \quad p = (7.5 \;\; 0)^T \right\}$$

and the control constraints

$$\mathcal{U} = \{ u \mid -3 \leq u \leq 5 \}$$

The terminal constraint set, $\mathcal{T}$, is calculated as an invariant set for an LQ state feedback control law

$$\kappa(x_k) = -K x_k$$

where the feedback gain is

$$K = -(I + B^T P B)^{-1} B^T P A$$

and $P$ is the solution to the Riccati equation with $Q = I$ and $R = 1$

$$P = A^T P (I + B B^T P)^{-1} A + C^T C$$

The terminal set for the nominal case, i.e., when the setpoint is the origin, will for this problem setup result in a relatively small terminal set polytope, $\mathcal{T}$, located in the small left part of the state constraint set, see Figure 5.3.

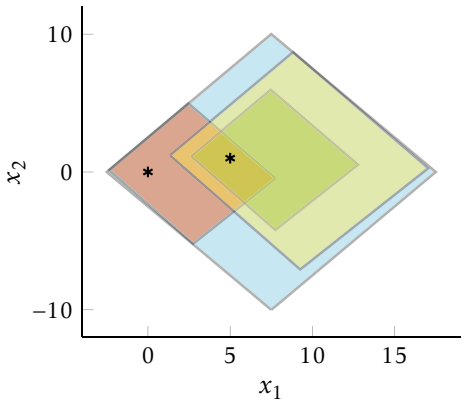


**Figure 5.3.** Illustration of how the terminal set $\mathcal{T}$ can be scaled with a $\lambda > 1$. The state constraint set $\mathcal{X}$ is the outermost polytope. The original terminal state constraint set, $\mathcal{T}$, for zero reference, is the dark shaded polytope to the left which has the origin (marked with a black star) in its interior. This set translated to the new setpoint (also marked with a star) without any scaling is the inner most of the two polytopes located around $x_1 = 5$. The outer of the two polytopes is the translated set, scaled with $\lambda = 1.55$.

When the system tracks a reference signal, the change of reference will then cause the terminal set to be shifted accordingly. From Figure 5.3 it is evident that $\mathcal{T}(\bar{x}) \subset \mathcal{X}$, i.e., the shifted terminal set lies in the strict interior of the feasible set. This means that it is possible to scale up the terminal set without violating the state constraints. However also the local control law needs to be

feasible in the scaled terminal set, i.e., $\kappa(x) \in \mathcal{U} \ \forall \ x \in \lambda\mathcal{T}(\bar{x})$.

In this example $\mathcal{T}(\bar{x})$ can be scaled with a $\lambda \leq 1.55$ before the state constraints limit the scaling.

Modifying the MPC tracking algorithm (4.13) with the above constraints we arrive at the following high-level representation of the optimization problem

$$\underset{u,x,\lambda_k,\bar{x}_k,\bar{u}_k,\bar{r}_k}{\text{minimize}} \ \sum_{i=0}^{N-1} \ell(x_{k+i} - \bar{x}_k, u_{k+i} - \bar{u}_k) + \Psi(x_{k+N} - \bar{x}_k) + \phi(\bar{r}_k - r) \quad (5.3a)$$

subj. to

$$x_{k+i+1} = Ax_{k+i} + Bu_{k+i} \quad \forall\, i = 0, \ldots, N-1 \quad (5.3b)$$

$$x_{k+i} \in \mathcal{X} \quad \forall\, i = 0, \ldots, N-1 \quad (5.3c)$$

$$u_{k+i} \in \mathcal{U} \quad \forall\, i = 0, \ldots, N-1 \quad (5.3d)$$

$$x_{k+N} \in \lambda_k\mathcal{T}(\bar{x}_k) \quad (5.3e)$$

$$\lambda_k\mathcal{T}(\bar{x}_k) \subseteq \mathcal{X} \quad (5.3f)$$

$$\bar{u}_k - K(x - \bar{x}_k) \in \mathcal{U} \ \forall x \in \lambda_k\mathcal{T}(\bar{x}_k) \quad (5.3g)$$

$$\begin{bmatrix} A - I & B \\ C & D \end{bmatrix} \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} = \begin{bmatrix} 0 \\ \bar{r}_k \end{bmatrix} \quad (5.3h)$$

$$\bar{x}_k \in \text{int}_\epsilon(\mathcal{X}) \quad (5.3i)$$

$$\bar{u}_k \in \text{int}_\epsilon(\mathcal{U}) \quad (5.3j)$$

In order to guarantee the convergence properties of the controller, our proof in section 5.2.4 requires the steady states and controls to be strictly feasible, i.e., the terminal set has to be placed in the strict interior of the feasible set. From a practical point of view we achieve this by constraining them to an $\epsilon$-interior of the feasible set, constraints (5.3i) and (5.3j). Simulations indicate that for practical purposes any arbitrarily small $\epsilon$ can be chosen, even $\epsilon = 0$.

Unfortunately, as the problem is written in (5.3), it is not suitable for implementation. For example as the constraints (5.1) and (5.2a) are written now, it is not obvious that they are linear in $\lambda_k$ and $\bar{x}_k$, and (5.2b) is an infinite-dimensional constraint. Hence, a reformulation is needed.

### 5.2.1 Vertex enumeration reformulation

As a first step we will show that the constraints in fact are linear in the variables and that the constraints can be formulated using a finite (although possibly large) number of linear constraints.

Let us begin with (5.3e) which constrains the terminal state to be inside the scaled and translated terminal set. Since the terminal set is polytopic, we have a representation of the form $\mathcal{T} = \{x \mid F_\mathcal{T} x \leq b_\mathcal{T}\}$. With our definitions of

translations and scaling from Chapter 2.2, we obtain a linear constraint.

$$F_{\mathcal{T}}(x_{k+N} - \bar{x}_k) \leq \lambda_k b_{\mathcal{T}} \tag{5.4}$$

The constraints (5.3f), which ensure the scaled and translated terminal set to be state feasible, and (5.3g), which ensures that the nominal control law $\bar{u}_k - K(x - \bar{x}_k)$ is feasible for any $x$ in the scaled and translated terminal set, can be rewritten using the vertex form of the terminal set $\mathcal{T}$.

Let $\mathcal{T}$ have $v_p$ vertices $v_j$ and it follows that $\lambda_k \mathcal{T}(\bar{x}_k)$ can be represented as the convex hull of the vertices $\bar{x}_k + \lambda_k v_j$. By convexity, this polytope is a subset of $\mathcal{X}$ if and only if all vertices are. With $\mathcal{X} = \{x \mid F_{\mathcal{X}} x \leq b_{\mathcal{X}}\}$ we obtain

$$F_{\mathcal{X}} x \leq b_{\mathcal{X}}, \quad \forall \left\{x \mid x = \bar{x}_k + \lambda_k v_j, \ \forall\, j = 1, \ldots, v_p\right\} \tag{5.5}$$

and we arrive at

$$F_{\mathcal{X}}(\bar{x}_k + \lambda_k v_j) \leq b_{\mathcal{X}} \ \forall\, j = 1, \ldots, v_p \tag{5.6}$$

Similarly for the control constraints, we have to ensure that

$$F_{\mathcal{U}}(\bar{u}_k - K(x - \bar{x}_k)) \leq b_{\mathcal{U}}, \quad \forall \left\{x \mid x = \bar{x}_k + \lambda_k v_j, \ \forall\, j = 1, \ldots, v_p\right\} \tag{5.7}$$

Inserting the vertices leads to

$$F_{\mathcal{U}}(\bar{u}_k - \lambda_k K v_j) \leq b_{\mathcal{U}} \ \forall\, j = 1, \ldots, v_p \tag{5.8}$$

As the constraints (5.6) and (5.8) are derived now the main drawback is that they require the enumeration of all vertices in the terminal set. This can, even for simple polytopes, be extremely expensive [Grünbaum, 2003], e.g., in the case when $\mathcal{T}$ is a simple box in $\mathbb{R}^n$ it has $2n$ facets but $2^n$ vertices and for higher dimensions this difference becomes crucial from a computational complexity point of view. Therefore we seek to avoid performing the vertex calculations, which can be done by rewriting the constraints (5.5) and (5.7) using duality theory.

### 5.2.2 Dual formulation of terminal set constraints

First note that the constraints (5.6) and (5.8) can be interpreted as uncertain constraints, i.e., they must hold for all $x \in \lambda_k \mathcal{T}(\bar{x}_k)$ and we can write them as

$$F_{\mathcal{U}}\left(\bar{u}_k - K\left(x - \bar{x}_k\right)\right) \leq b_{\mathcal{U}} \tag{5.9a}$$

$$F_{\mathcal{X}} x \leq b_{\mathcal{X}} \tag{5.9b}$$

for all $x$ such that

$$\{x \mid F_{\mathcal{T}}\left(x - \bar{x}_k\right) \leq \lambda_k b_{\mathcal{T}}\}$$

Now consider a general uncertain affine constraint of the form

$$\left(c^T x + d\right) + (Ax + b)^T p \leq 0 \ \forall \left\{x \mid F^T x \leq E\right\} \tag{5.10}$$

where $x$ is an uncertain, but bounded, variable and $p$ is the decision variable. This constraint must hold for all possible values of $x$, i.e., it must hold for the worst case $x$, in this case, the one that maximize the left hand side.

This maximum of the left hand side can be determined by the following optimization problem

$$\max_x. \ \left(A^T p + c\right)^T x + \left(b^T p + d\right) \ \text{ subj. to } F^T x \leq E \tag{5.11}$$

Additionally we know that for a solution $p$ to satisfy (5.10) it must hold that the left hand side is negative and hence also gives a negative optimal cost for the optimization problem (5.11).

From duality theory, see, e.g., Boyd and Vandenberghe [2004] or Nesterov and Nemirovskii [1994], the dual problem of (5.11) can be formulated in the dual variable $\zeta$ as

$$\min_\zeta. \ E^T \zeta + \left(b^T p + d\right) \ \text{ subj. to } \ \zeta \geq 0, \ F\zeta = A^T p + c \tag{5.12}$$

Since the optimal value to the primal problem (5.11) is negative it then follows that the optimal value of the dual problem, which is a tight upper bound to the primal problem, is negative. Thus we can conclude that given a solution $p$ and the existence of a dual variable $\zeta \geq 0$ such that

$$E^T \zeta + \left(b^T p + d\right) \leq 0, \ \ F\zeta = A^T p + c \tag{5.13}$$

is equivalent to that $p$ satisfies the constraint (5.10) for all values of the uncertain variable $x$. This derivation has previously been presented in, e.g., Ben-Tal and Nemirovski [2002] in the context of robust optimization.

Let us now use this result to rewrite the constraints (5.6) and (5.8) into the dual form. Denote the decision variable $p_k = (\bar{x}_k, \bar{u}_k)^T$ and write each row of (5.9) generically as

$$f_x^T x + f_p^T p_k + d \leq 0 \ \forall \ \{x \mid F_\mathcal{T} (x - \bar{x}_k) \leq \lambda_k b_\mathcal{T}\}$$

where $f_x^T$ denotes the rows of the matrix

$$F_x = \begin{bmatrix} -F_\mathcal{U} K \\ F_\mathcal{X} \end{bmatrix}$$

and $f_p^T$ denotes the rows of

$$F_p = \begin{bmatrix} F_\mathcal{U} K & F_\mathcal{U} \\ 0 & 0 \end{bmatrix}$$

and $d$ is the corresponding row element of the vector $\begin{pmatrix} -b_\mathcal{U}^T & -b_\mathcal{X}^T \end{pmatrix}^T$.

Writing this in the same structure as (5.10) gives

$$\left(f_x^T x + d\right) + \left(0x + f_p\right)^T p_k \leq 0 \ \forall \ \{x \mid F_\mathcal{T} x \leq \lambda_k b_\mathcal{T} + F_\mathcal{T} \bar{x}_k\}$$

and the dual form then becomes

$$(\lambda_k b_{\mathcal{T}} + F_{\mathcal{T}} \bar{x}_k)^T \zeta + f_p^T p_k + d \leq 0, \;\; F_{\mathcal{T}}^T \zeta = f_x, \; \zeta \geq 0$$

Expand the parenthesis and use the fact that $F_{\mathcal{T}}^T \zeta = f_x$, this gives

$$\lambda_k b_{\mathcal{T}}^T \zeta + \bar{x}_k^T f_x + f_p^T p_k + d \leq 0, \;\; F_{\mathcal{T}}^T \zeta = f_x, \; \zeta \geq 0$$

Hence, we should have a global solution $(\lambda_k, p_k)$ such that there exist a $\zeta$ for every row satisfying this set of equations. Note that the term $b_{\mathcal{T}}^T \zeta$ is non-negative (both $b_{\mathcal{T}}^T$ and $\zeta$ are non-negative) since the terminal set contains the origin in its interior. Hence, it is beneficial to make this term minimal for every row. Note also that this choice of $b_{\mathcal{T}}^T \zeta$ is independent of the variables $\lambda_k$, $p_k$. Therefore let

$$\gamma = \min_{\zeta}. \; b_{\mathcal{T}}^T \zeta \;\; \text{subj. to } F_{\mathcal{T}}^T \zeta = f_x, \; \zeta \geq 0 \tag{5.14}$$

and replace the term $b_{\mathcal{T}}^T \zeta$ with the solution $\gamma$, finally giving

$$\lambda_k \gamma + \bar{x}_k^T f_x + f_p^T p_k + d \leq 0 \tag{5.15}$$

We can now replace each row of the uncertain constraints (5.6) and (5.8) with (5.15) and if we write this into a matrix form we obtain

$$\lambda_k \Gamma + \begin{bmatrix} -F_{\mathcal{U}} K \\ F_{\mathcal{X}} \end{bmatrix} \bar{x}_k + \begin{bmatrix} F_{\mathcal{U}} K & F_{\mathcal{U}} \\ 0 & 0 \end{bmatrix} \begin{pmatrix} \bar{x}_k \\ \bar{u}_k \end{pmatrix} \leq \begin{pmatrix} b_{\mathcal{U}} \\ b_{\mathcal{X}} \end{pmatrix} \tag{5.16}$$

where the vector $\Gamma$ has the solution $\gamma$ to (5.14), for each $f_x$ of $F_x$, as its elements.

The equations (5.4) and (5.16) gives a complete description of the terminal state constraint set.

### 5.2.3 The QP formulation

We can now summarize all the pieces into the quadratic program formulation of the proposed controller.

First select the different parts of the cost function in (5.3) as

$$\ell(x_{k+i} - \bar{x}_k, u_{k+i} - \bar{u}_k) = \|x_{k+i} - \bar{x}_k\|_Q^2 + \|u_{k+i} - \bar{u}_k\|_R^2 \tag{5.17a}$$

$$\Psi(x_{k+N} - \bar{x}_k) = \|x_{k+N} - \bar{x}_k\|_P^2 \tag{5.17b}$$

$$\phi(\bar{r}_k - r) = \beta \|\bar{r}_k - r\|_\infty \tag{5.17c}$$

In (5.17a), $Q$ and $R$ are positive definite penalty matrices, used also to define the Lyapunov cost matrix $P$ in (5.17b) and nominal state feedback $K$ through

$$(A - BK)^T P (A - BK) - P = -Q - K^T R K \tag{5.18}$$

In (5.17c), $\beta$ is a positive scalar, $r$ is the desired reference to track and $\bar{r}_k$ is the pseudo reference variable. Later in this chapter we will discuss on how to choose the scalar $\beta$ in order to achieve desired properties.

The proposed controller is then defined by the solution to the following quadratic programming problem

$$\underset{u,x,\lambda_k,\bar{x}_k,\bar{u}_k,\bar{r}_k}{\text{minimize}} \quad \Psi(x_{k+N} - \bar{x}_k) + \phi(\bar{r}_k - r) + \sum_{i=0}^{N-1} \ell(x_{k+i} - \bar{x}_k, u_{k+i} - \bar{u}_k) \tag{5.19a}$$

subj. to

$$x_{k+i+1} = Ax_{k+i} + Bu_{k+i} \quad \forall\, i = 0, \ldots, N-1 \tag{5.19b}$$

$$F_{\mathcal{X}} x_{k+i} \leq b_{\mathcal{X}} \quad \forall\, i = 0, \ldots, N-1 \tag{5.19c}$$

$$F_{\mathcal{U}} u_{k+i} \leq b_{\mathcal{U}} \quad \forall\, i = 0, \ldots, N-1 \tag{5.19d}$$

$$F_{\mathcal{T}}(x_{k+N} - \bar{x}_k) \leq \lambda_k b_{\mathcal{T}} \tag{5.19e}$$

$$\lambda_k \Gamma + F_x \bar{x}_k + F_p \begin{pmatrix} \bar{x}_k \\ \bar{u}_k \end{pmatrix} \leq \begin{pmatrix} b_{\mathcal{U}} \\ b_{\mathcal{X}} \end{pmatrix} \tag{5.19f}$$

$$\begin{bmatrix} A - I & B \\ C & D \end{bmatrix} \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} = \begin{bmatrix} 0 \\ \bar{r}_k \end{bmatrix} \tag{5.19g}$$

$$F_{\mathcal{X}}\,(\bar{x}_k) \leq (1 - \epsilon)\, b_{\mathcal{X}}, \quad \epsilon > 0 \tag{5.19h}$$

$$F_{\mathcal{U}}\,(\bar{u}_k) \leq (1 - \epsilon)\, b_{\mathcal{U}}, \quad \epsilon > 0 \tag{5.19i}$$

Solving the above problem gives an optimal control input sequence $\{u^*_{k+i}\}_{i=0}^{N-1}$ and the applied MPC law is the first element of this optimal sequence

$$\kappa(x_k) = u^*_k \tag{5.20}$$

It should be noted that the above formulation, in strict sense, is not a quadratic program on standard form since the cost function contains the infinity norm. However it is a trivial task to reformulate this into the standard QP form (2.2).

### 5.2.4 Stability and feasibility of the proposed algorithm

In the following theorem we establish the stability and convergence properties of the proposed controller that was derived in the previous section. Before we outline the stability properties of our proposed algorithm we must make some necessary assumptions and definitions.

**Definition 5.2.** $r_\perp$ is defined as the closest strictly feasible point to the reference $r$ in a distance measure determined by the function $\phi(\cdot)$

$$r_\perp = \arg\min_{\bar{r}} \ \phi(\bar{r} - r)$$

subj. to

$$\begin{bmatrix} A - I & B \\ C & D \end{bmatrix} \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix} = \begin{bmatrix} 0 \\ \bar{r} \end{bmatrix}$$
$$\bar{x} \in \text{int}_\epsilon(\mathcal{X})$$
$$\bar{u} \in \text{int}_\epsilon(\mathcal{U})$$

**Assumption 5.3.** The matrix

$$\begin{bmatrix} A - I & B \\ C & D \end{bmatrix}$$

is such that there exists a solution to (4.11) for any reference $r$.

**Assumption 5.4.** $\mathcal{X}$, $\mathcal{U}$ and $\mathcal{T}$ contain the origin in their interior.

**Lemma 5.5.** *Let $\mathcal{T}$ be a positively invariant set for a stable discrete time linear system, $x_{k+1} = A_c x_k$. Then for any scalar $\lambda \geq 0$, the scaled set, $\lambda\mathcal{T}$ is also a positively invariant set for the system.*

**Proof:** The lemma follows directly from the scaling invariance of linear systems, see, e.g., Blanchini [1999]. □

With these definitions and assumptions in order, we are ready to formulate the main stability theorem for the proposed controller.

**Theorem 5.6.** *For any feasible initial state $x_0$ and Assumptions 5.3 and 5.4 , the MPC algorithm defined through* (5.19) - (5.20) *remains feasible and stabilizes the system* (4.9)*. Additionally, $x_k$ asymptotically converges to a setpoint given by the projection of the reference $r$ onto the ($\epsilon$-contracted) feasible set.*

The proof of the theorem is relatively straightforward and recursive feasibility and convergence of the state to a stationary point follow standard proofs found in the literature [Mayne et al., 2000]. In a second step, to show that the setpoint to which the state converges is the setpoint associated with the given reference $r$, if feasible, or the setpoint corresponding to the closest feasible reference has previously been proven in similar ways in Ferramosca et al. [2009].

**Proof:** Let $\mathcal{X}_N$ be the set of $x$ where (5.19) is feasible. Assume $x_k \in \mathcal{X}_N$ with an optimal solution given by the sequence $\{u^*_{k+i}\}_{i=0}^{N-1}$ and $\lambda^*_k$ and $\bar{r}^*_k$, with predicted state trajectory $\{x^*_{k+i}\}_{i=1}^{N}$.

At the next time step $\{\hat{u}_{k+i}\}_{i=1}^{N} = \{u^*_{k+1}, u^*_{k+2}, \ldots, u^*_{k+N-1}, \bar{u}^*_k - K(x^*_{k+N} - \bar{x}^*_k)\}$ is a feasible control sequence, since $\bar{u}^*_k - K(x^*_{k+N} - \bar{x}^*_k)$ is feasible according to (5.3g). Furthermore, we use $\lambda_{k+1} = \lambda^*_k$ and $\bar{r}_{k+1} = \bar{r}^*_k$. Keeping $\lambda_{k+1}$ and $\bar{r}_{k+1}$ unchanged means that we keep the scaled and translated terminal set

unchanged. The new, possibly suboptimal, state sequence is $\{\hat{x}_{k+i}\}_{i=2}^{N+1}$ where $x_{k+N+1} - \bar{x}_k^* = (A-BK)(x_{k+N}^* - \bar{x}_k^*)$. Since $\lambda_k^*\mathcal{T}$ is positively invariant w.r.t. the system $x_{k+1} = (A-BK)x_k$ according to Lemma 5.5, it follows that $x_{k+N+1} - \bar{x}_k^*$ stays in $\lambda_k^*\mathcal{T}$, i.e., the terminal state constraint $x_{k+N+1} \in \lambda_k^*\mathcal{T}(\bar{x}_k^*)$ is satisfied. Since $\lambda_k^*\mathcal{T}(\bar{x}_k^*) \subseteq \mathcal{X}$, state constraints are trivially satisfied.

Now assume that we have an optimal solution at time $k$ and denote the optimal cost

$$V_k^* = \sum_{i=0}^{N-1} \left( \left\| x_{k+i}^* - \bar{x}_k^* \right\|_Q^2 + \left\| u_{k+i}^* - \bar{u}_k^* \right\|_R^2 \right) + \left\| x_{k+N}^* - \bar{x}_k^* \right\|_P^2 + \phi(\bar{r}_k^* - r)$$

Applying the control sequence $\{\hat{u}_{k+i}\}_{i=1}^{N}$ defined in the previous section gives the suboptimal cost $V_{k+1}$ as

$$\begin{aligned}
&\sum_{i=0}^{N-1} \left( \left\| x_{k+1+i}^* - \bar{x}_k^* \right\|_Q^2 + \left\| u_{k+1+i}^* - \bar{u}_k^* \right\|_R^2 \right) + \left\| x_{k+1+N} - \bar{x}_k^* \right\|_P^2 + \phi(\bar{r}_k^* - r) \\
&= \sum_{i=0}^{N-2} \left( \left\| x_{k+1+i}^* - \bar{x}_k^* \right\|_Q^2 + \left\| u_{k+1+i}^* - \bar{u}_k^* \right\|_R^2 \right) + \left\| x_{k+N}^* - \bar{x}_k^* \right\|_Q^2 \\
&+ \left\| \bar{u}_k^* - K(x_{k+N} - \bar{x}_k^*) - \bar{u}_k^* \right\|_R^2 + \left\| x_{k+1+N} - \bar{x}_k^* \right\|_P^2 + \phi(\bar{r}_k^* - r) \\
&+ \left\| x_k^* - \bar{x}_k^* \right\|_Q^2 + \left\| u_k^* - \bar{u}_k^* \right\|_R^2 + \left\| x_{k+N}^* - \bar{x}_k^* \right\|_P^2 \\
&- \left\| x_k^* - \bar{x}_k^* \right\|_Q^2 - \left\| u_k^* - \bar{u}_k^* \right\|_R^2 - \left\| x_{k+N}^* - \bar{x}_k^* \right\|_P^2 \\
&= \underbrace{\sum_{i=0}^{N-1} \left( \left\| x_{k+i}^* - \bar{x}_k^* \right\|_Q^2 + \left\| u_{k+i}^* - \bar{u}_k^* \right\|_R^2 \right) + \left\| x_{k+N}^* - \bar{x}_k^* \right\|_P^2 + \phi(\bar{r}_k^* - r)}_{V_k^*} \\
&\underbrace{+ \left\| x_{k+N}^* - \bar{x}_k^* \right\|_Q^2 + \left\| K(x_{k+N}^* - \bar{x}_k^*) \right\|_R^2 + \left\| x_{k+1+N} - \bar{x}_k^* \right\|_P^2 - \left\| x_{k+N}^* - \bar{x}_k^* \right\|_P^2}_{=0 \text{ due to (5.18)}} \\
&- \left\| x_k^* - \bar{x}_k^* \right\|_Q^2 - \left\| u_k^* - \bar{u}_k^* \right\|_R^2
\end{aligned}$$

and thus, it follows that the suboptimal cost is

$$V_{k+1} = V_k^* - \left\| x_k^* - \bar{x}_k^* \right\|_Q^2 - \left\| u_k^* - \bar{u}_k^* \right\|_R^2$$

which implies that the optimal cost $V_{k+1}^*$ fulfills

$$V_{k+1}^* \leq V_{k+1} = V_k^* - \left\| x_k^* - \bar{x}_k^* \right\|_Q^2 - \left\| u_k^* - \bar{u}_k^* \right\|_R^2 < V_k^*$$

In other words, $V_k^*$ is strictly decreasing as long as $x_k^* \neq \bar{x}_k^*$ and $u_k^* \neq \bar{u}_k^*$. Hence $x_k^* \to \bar{x}_k^*$ and $u_k^* \to \bar{u}_k^*$. Note that in the limit we have, since $\bar{x}_k^*$ and $\bar{u}_k^*$ represent a stationary pair, that $\bar{x}_k^* = x_k^* = x_{k+1}^* = \bar{x}_{k+1}^*$, i.e., the pseudo setpoint

converges too.

To show convergence of $\bar{r}_k^* \to r_\perp$, assume that the system has settled at a setpoint given by $\bar{x}_k^*, \bar{u}_k^*$, defined by $\bar{r}_k^*$. The proof will proceed by contradiction, so we assume $\bar{r}_k^* \neq r_\perp$. Consider a perturbation ($0 \leq \gamma < 1$) of the pseudo reference $\bar{r}_k^*$ towards $r_\perp$, given by

$$\bar{r}_\gamma = \gamma \bar{r}_k^* + (1-\gamma) r_\perp$$

Our first step is to show that this choice is feasible for $\gamma$ sufficiently close to 1. By convexity, $\bar{r}_\gamma$ is feasible with respect to (5.19h) and (5.19i). We use the constant control sequence corresponding to the steady state control given by $\bar{r}_\gamma$, i.e. $u_{k+i} = \bar{u}_\gamma$ (which is also feasible by convexity) and the predicted states evolve according to

$$\begin{aligned}
x_{k+1} &= A\bar{x}_k^* + B\bar{u}_\gamma \\
&= A\bar{x}_k^* + \gamma B\bar{u}_k^* + (1-\gamma)Bu_\perp \\
&= A(\gamma\bar{x}_k^* + (1-\gamma)x_\perp + \bar{x}_k^* - \gamma\bar{x}_k^* - (1-\gamma)x_\perp) + \gamma B\bar{u}_k^* + (1-\gamma)Bu_\perp \\
&= \gamma(A\bar{x}_k^* + B\bar{u}_k^*) + (1-\gamma)(Ax_\perp + Bu_\perp) + (1-\gamma)A(\bar{x}_k^* - x_\perp) \\
&= \gamma\bar{x}_k^* + (1-\gamma)x_\perp + (1-\gamma)A(\bar{x}_k^* - x_\perp) \\
&= \bar{x}_\gamma + (1-\gamma)A(\bar{x}_k^* - x_\perp)
\end{aligned}$$

Applying the same manipulations recursively leads to

$$x_{k+i} = \bar{x}_\gamma + (1-\gamma)A^i(\bar{x}_k^* - x_\perp)$$

For $\gamma$ sufficiently close to 1 all predicted states are feasible w.r.t. $\mathcal{X}$, since $\bar{x}_k^*$ and $x_\perp$ (and thus $\bar{x}_\gamma$) are strictly inside $\mathcal{X}$ and $(1-\gamma)A^i(\bar{x}_k^* - x_\perp)$ approaches 0 as $\gamma$ goes to 1. What remains to show is that we can select $\lambda_k$ such that $x_{k+N+1} \in \lambda_k \mathcal{T}(\bar{x}_\gamma)$ and $\lambda_k \mathcal{T}(\bar{x}_\gamma) \subseteq \mathcal{X}$.

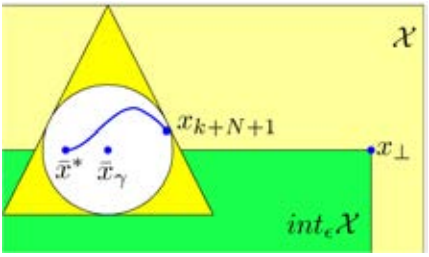


**Figure 5.4.** Illustration of the components in the proof of convergence of the pseudo reference. The figure shows portions of the sets $\mathcal{X}$ and $\text{int}_\epsilon(\mathcal{X})$, and the triangular set $\lambda_k \mathcal{T}(\bar{x}_\gamma)$ with its inscribed Euclidean ball.

Since the pseudo reference is in the strict interior (defined by $\epsilon$), it immediately follows that there exist a constant $\epsilon_\lambda > 0$, determined by the geometry of $\mathcal{T}, \mathcal{X}$, $\mathcal{U}$ and $\epsilon$, such that $\epsilon_\lambda \mathcal{T}(\bar{x}) \subseteq \mathcal{X}$ for any strictly feasible $\bar{x}$. Let $d$ denote

the radius of the largest possible Euclidean ball centered at the origin which can be inscribed in $\mathcal{T}$, see Figure 5.4. Since $\mathcal{T}$ contains 0 in its interior by assumption, $d > 0$. The distance from the terminal state $x_{k+N+1}$ to the new pseudo setpoint is given by $\left\|x_{k+N+1} - \bar{x}_\gamma\right\| = (1-\gamma)\left\|A^{N+1}(\bar{x}_k^* - x_\perp)\right\|$. If this distance is smaller than $\lambda_k d$, the terminal state is inside the scaled and translated terminal set. Hence, if $\gamma \geq 1 - \frac{\epsilon_\lambda d}{\left\|A^{N+1}(\bar{x}_k^* - x_\perp)\right\|}$ the terminal state constraint is fulfilled. Since $\mathcal{X}$ is polytopic, the denominator in the expression has an upper bound.

Returning back to the objective function for our proposed feasible solution, and using the notation $\Psi_i = A^i(\bar{x}_k^* - x_\perp)$, we arrive at

$$V_k = ||(1-\gamma)\Psi_N||_P^2 + \sum_{i=0}^{N-1} ||(1-\gamma)\Psi_i||_Q^2 + \underbrace{\left\|u_{k+i} - \bar{u}_\gamma\right\|_R^2}_{=0} + \phi\left(\gamma\bar{r}_k^* + (1-\gamma)r_\perp - r\right)$$

$$= (1-\gamma)^2 ||\Psi_N||_P^2 + (1-\gamma)^2 \sum_{i=0}^{N-1} ||\Psi_i||_Q^2 + \phi\left(\gamma(\bar{r}_k^* - r_\perp) + (r_\perp - r)\right)$$

Differentiate $\mathcal{J}_k$ with respect to the step size $\gamma$

$$\frac{\partial \mathcal{J}_k}{\partial \gamma} = -2(1-\gamma)\left(||\Psi_N||_P^2 + \sum_{i=0}^{N-1} ||\Psi_i||_Q^2\right) + c^T(\bar{r}_k^* - r_\perp)$$

where $c$ is the subgradient to the function $\phi(\cdot)$ at $\gamma = 1$. Evaluating this at $\gamma = 1$ gives

$$\left.\frac{\partial \mathcal{J}_k}{\partial \gamma}\right|_{\gamma=1} = c^T(\bar{r}_k^* - r_\perp)$$

If this inner product is positive, the cost function decreases as $\gamma$ decreases which in turn implies that the cost can be reduced by moving $\bar{r}_k^*$ closer to $r_\perp$. From the definition of the subgradient it follows that

$$\phi(r_\perp - r) \geq \phi(\bar{r}_k^* - r) + c^T(r_\perp - \bar{r}_k^*)$$

which gives

$$c^T(\bar{r}_k^* - r_\perp) \geq \phi(\bar{r}_k^* - r) - \phi(r_\perp - r)$$

Since $r_\perp$ by definition is the closest feasible point to $r$ in the chosen norm, the right hand side is strictly greater than zero unless $\bar{r}_k^* = r_\perp$. This means that the cost, $V_k$, can be improved by making an arbitrarily small move of $\bar{r}_k^*$ towards $r_\perp$ and hence the only stationary point for $\bar{r}_k$ is $r_\perp$. Since we previously proved that $x_k$ asymptotically converges to $\bar{x}_k$, and now proved that $\bar{x}_k$ converges to $x_\perp$ we can conclude that as $x_k$ comes sufficiently close to $\bar{x}_k$ the cost will be reduced by moving $\bar{x}_k$ closer to $x_\perp$ and hence, $x_k$ will asymptotically converge to $x_\perp$. □

Note that it is not self-evident that this tracking MPC algorithm, using pseudo setpoints, is locally optimal in the sense that it minimizes the infinite horizon LQR cost in a vicinity of the setpoint, or in other words that it gives the same solution as the infinite LQ controller when possible.

However in the recent work by Ferramosca et al. [2011] the authors argue that under certain conditions a similar type of MPC algorithm has the local optimality property. In fact if the pseudo reference penalty is an exact penalty function, then it directly follows that the controller has the local optimality property. Since the same arguments can be used to show that also this algorithm possesses the local optimality property will we only briefly outline the train of thought and leave the details to the reader.

If $\phi(\bar{r}_k - r) = \beta \|\bar{r}_k - r\|_\infty$ and $\beta$ is chosen such that $\phi(\bar{r}_k - r)$ constitutes an exact penalty function, as described in Section 4.2.4, then for all $x_k$ where $\phi(\bar{r}_k - r) = 0$ is a feasible solution to (5.19), we will have $V_k^* = \hat{V}_k^*$, where $\hat{V}_k^*$ is the solution to the related optimization problem

$$\underset{u,x,\lambda_k,\bar{x}_k,\bar{u}_k,\bar{r}_k}{\text{minimize}} \quad \Psi(x_{k+N} - \bar{x}_k) + \sum_{i=0}^{N-1} \ell(x_{k+i} - \bar{x}_k, u_{k+i} - \bar{u}_k) \tag{5.21}$$

subject to constraints (5.19b) - (5.19i) and the constraint $\phi(\bar{r}_k - r) = 0$ and where $\Psi(\cdot)$ is defined by (5.17b) and $\ell(\cdot)$ by (5.17a). Note that the optimization problem (5.21) is a dual mode formulation of the MPC controller (although translated to a new origin).

The results from Sznaier and Damborg [1987] show that the dual mode MPC formulation, with terminal state penalty equal to the infinite horizon unconstrained LQ cost and with a local controller that is the unconstrained LQ controller, has the local optimality property, i.e., the finite horizon cost equals that of the infinite horizon LQ problem, $V_\infty$. From this it is clear that for all $x_k$ where $V_k^* = \hat{V}_k^*$ it holds also that $V_k^* = V_\infty^*$. Note that the set of states, for which the local optimality results hold for the proposed controller, is not maximally large. This can be realized when considering that the scaled terminal set is not the maximum invariant terminal set that can be constructed for a given setpoint.

## 5.3 Examples from the aeronautical industry

In this section we illustrate the proposed method by two examples from the aeronautical industry. The first example considers maneuver limiting of an unstable fighter aircraft and in the second example we consider flight envelope protection for a small scale unmanned helicopter.

### 5.3.1 Maneuver limitations on a fighter aircraft

We will in this example compare the proposed controller to the reference tracking MPC algorithm first proposed in Limon et al. [2008] and then analyzed and extended in Ferramosca et al. [2009] and Ferramosca et al. [2011]. We will refer to this method as the *reference method*. Both controllers have been tuned using the same penalty matrices.

In this example we consider the linearized short period dynamics (3.8) for the ADMIRE model, see 3.4, discretized using a sample-time of 16ms.

$$\begin{bmatrix} \alpha_{k+1} \\ q_{k+1} \end{bmatrix} = \underbrace{\begin{bmatrix} 0.9719 & 0.0155 \\ 0.2097 & 0.9705 \end{bmatrix}}_{A} \begin{bmatrix} \alpha_k \\ q_k \end{bmatrix} + \underbrace{\begin{bmatrix} 0.0071 \\ 0.3263 \end{bmatrix}}_{B} \delta_k$$

$$y = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} \alpha_k \\ q_k \end{bmatrix}$$

The maneuver limits for angle of attack and pitch rate have been set to

$$x_k \in \mathcal{X} = \left\{ (\alpha, q)^T \mid -15 \leq \alpha \leq 30, -100 \leq q \leq 100 \right\}$$

The elevator control surface angle deflection limits have been set to 25°

$$\delta_k \in \mathcal{U} = \{ \delta \mid -25 \leq \delta \leq 25 \}$$

The objective is to have the state $\alpha$ track a reference $r$ given by the pilots control stick input, as close as possible to the boundary of the feasible set $\mathcal{X}$.

The penalty matrices in the costfunction (5.17a) have been chosen as

$$Q = \begin{bmatrix} 10 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1$$

and $P$ in (5.17b) is the Lyapunov cost of the corresponding LQ controller. The prediction horizon is chosen to $N = 20$ and $\epsilon$ is chosen small enough to not have any noticeable effect, $\epsilon = 10^{-5}$.

This example been implemented in MATLAB with the use of the toolboxes YALMIP [Löfberg, 2004] and MPT [Herceg et al., 2013].

#### Nominal performance

The step response for both controllers are shown in Figure 5.5. The response in angle of attack is very similar between the proposed method and the reference method, only a slightly faster convergence can be observed for the proposed controller as the angle of attack approaches the border of the feasible set. Note that $\alpha$ converges to the desired reference if feasible. When setting the reference to $\alpha = 30°$, i.e., when it is located on the border of $\mathcal{X}$ the output will track the reference, but when the reference is set to $\alpha = -20°$,

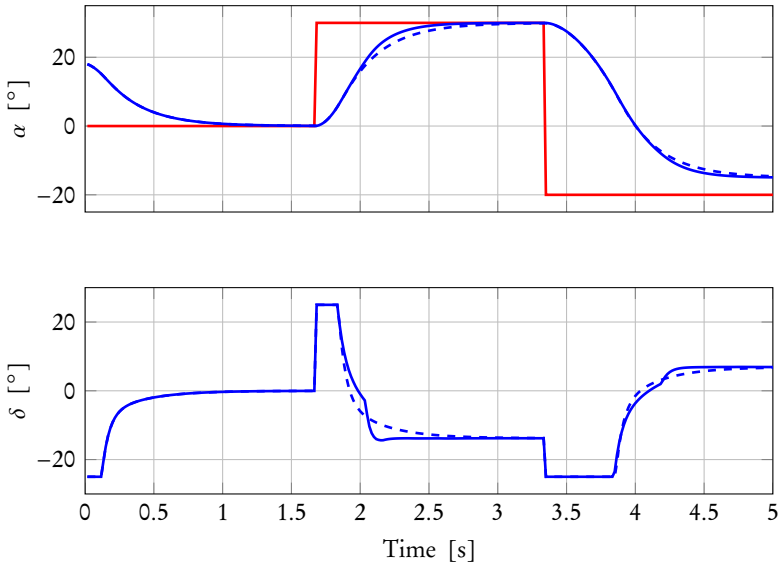


**Figure 5.5.** The upper axis shows the pilot input reference, $r$ (step shaped signal) and the output, $\alpha$, for the two controllers. The proposed controller is the solid line and the reference controller is the dashed line. The lower axis shows the control signal for both controllers.

i.e., outside the feasible set, the output will track the pseudo reference that converges to the closest feasible point.

Since the terminal constraint set is a scaled version of the nominal invariant set, i.e., the one calculated for $r = 0$, it is clearly not the maximal invariant set and hence, it can result in a smaller domain of attraction. This drawback becomes evident only for shorter prediction horizons, e.g., for $N = 5$ the proposed controller has a smaller domain of attraction than the reference method, see Figure 5.6, while for $N = 10$ there is no difference between the two controllers. Therefore one must make a tradeoff between the complexity, i.e., the prediction horizon, and the needed domain of attraction.

### Complexity of explicit solution

Comparing the complexity, i.e., the size of the QP, of the proposed controller with the reference method we can conclude that the proposed controller results in a large reduction in number of constraints. The reference method has 66 variables and 229 constraints while the proposed controller has 67 variables, but only 164 constraints. The significant difference in number of constraints comes from the terminal constraint set which is defined with 72 inequalities, in $\mathbb{R}^3$, for the reference method compared to only 4 inequalities, in $\mathbb{R}^2$, for the proposed controller. The large amount of inequalities needed to describe the terminal set for the reference method comes from the structure of the

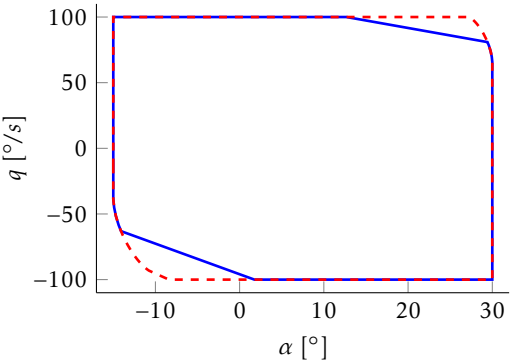


**Figure 5.6.** The domain of attraction for the proposed controller (blue solid) and the reference method (red dashed line) for prediction horizon $N = 5$.

augmented system and has been noted by Limon et al. [2008]. In fact the authors state that it might not be possible to determine the terminal set with a finite number of constraints and if that is the case one has to shrink the terminal set such that the reference is constrained to an interior of the feasible set.

This difference in complexity will result in lower computational effort for online solution of the QP or a reduced complexity of an explicit implementation. When calculating the explicit MPC solution, the resulting controller for the reference method, with a prediction horizon of $N = 10$, has $N_{\mathcal{X}} = 6850$ partitions, compared to the explicit solution for the proposed method which has only $N_{\mathcal{X}} = 840$ partitions, i.e., a reduction of the number of partitions with 87%. In Figure 5.7 the state space partitioning for a zero reference is shown. Compared to the purely stabilizing controller that was calculated in Section 4.2.5 we can see that adding the possibility of reference tracking increases the complexity of the solution also for zero references.

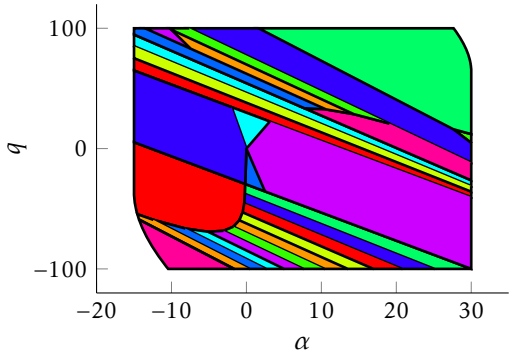


**Figure 5.7.** State space partition of the explicit solution for reference, $r = 0$.

Figure 5.8 shows the optimal control signal as a (piecewise affine) function of the states for different references. It is clear that the optimal feedback varies considerably with the reference. For zero reference (the upper left figure) one can recognize it as the saturated LQ solution from Chapter 4 and for reference values of $r = -10$ (upper right figure) and $r = 20$ (lower left figure) there are small regions, i.e., inside the invariant set, where the LQ solution is still valid. For the reference $r = 30$ (the lower right figure) we can see that the control signal is saturated for the most part of the state space and only for large angular rates there is a varying state feedback. For this reference value the terminal constraint set is scaled down to a single point and hence nothing of the original LQ solution is in the feedback.

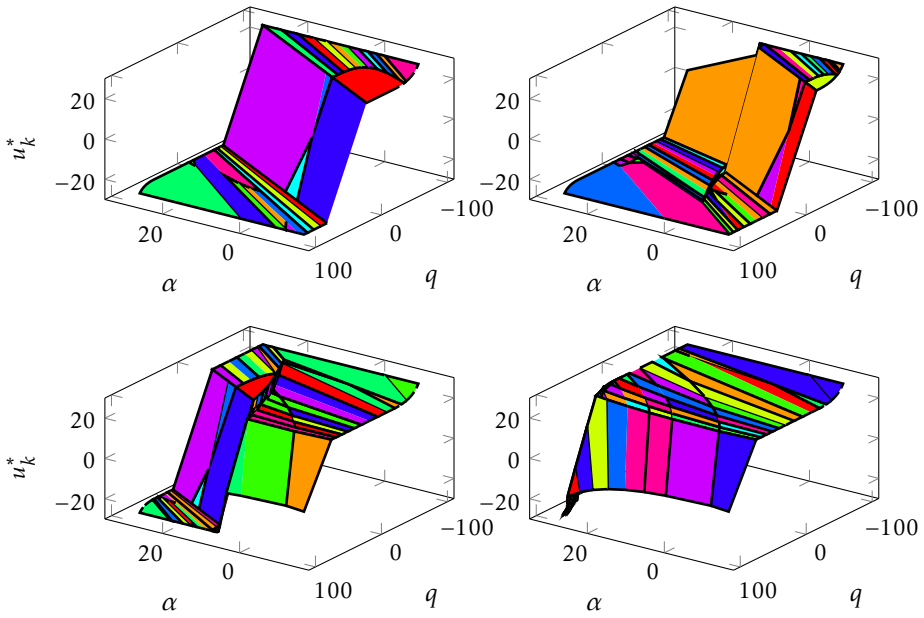


**Figure 5.8.** Explicit feedback solution for different reference signals. Upper left axis show the feedback solution for $r = 0$, upper right axis shows the feedback solution for $r = -10$ and the two axis on the bottom shows the solution for $r = 20$ (left) and $r = 30$ (right).

### Robust performance and integral control

In this section we look at the performance of the proposed controller when the true system is different than that of the prediction model in the controller. Since both the proposed controller and the reference controller have similar performance characteristics we will restrict the discussion to the proposed method.

Let the true system have a 25% larger destabilizing pitching moment, i.e., the

(2, 1) element of the $A$-matrix is 25% larger than in the model. Additionally we let the true $B$-matrix be scaled with a factor of 0.8, i.e., the true system has 20% less control surface effectiveness than modeled.

The time response of the closed loop system with the proposed controller and the true system is shown in Figure 5.9. The initial response is slightly slower than for the case where there are no model errors and the angle of attack response overshoots the reference signal, settling at steady state value that is larger than the reference. This overshoot will result in violations of the state constraints when the reference approaches the boundary of constraint set.

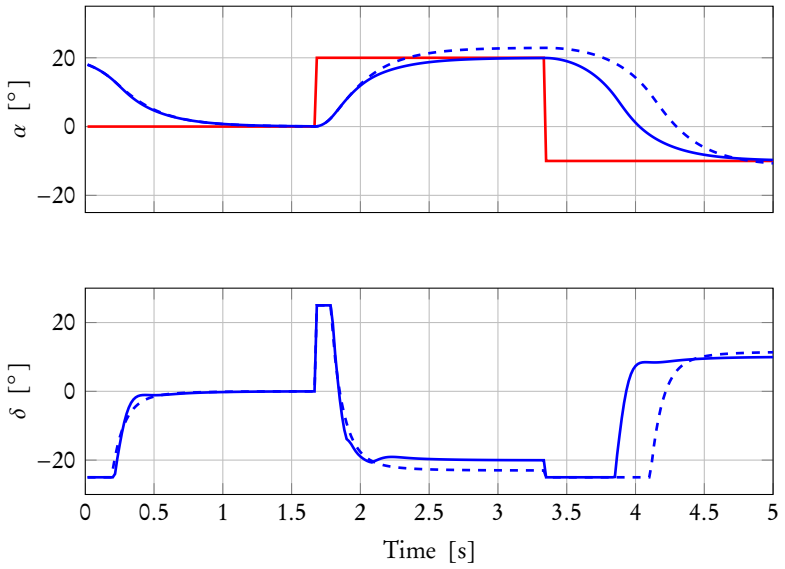


**Figure 5.9.** Time response of the proposed controller (dashed) when the true system is different than that of the prediction model. The solid line is the time response of the proposed controller with added integral control.

To overcome this difficulty we add integral control as described in Section 4.2.3. We can see from Figure 5.9 that the performance is much better and the reference is attained without any steady state error. Even though the disturbance variable that is estimated in the disturbance observer is modeled as a constant disturbance, it can clearly catch dynamic behavior such as model errors.

In this thesis we do not investigate further on the theoretical properties of integral control, but leave this as an open topic for future research.

### 5.3.2 Nonlinear aircraft performance

Now we will consider a more advanced example in which we have applied the developed tracking algorithm to the full nonlinear dynamics of the ADMIRE

model. We have implemented the MPC controller described in the previous sections to the pitch dynamics of the ADMIRE aircraft and used a standard LQ controller for the lateral dynamics.

The MPC controller is tuned for the short period dynamics linearized around trimmed level flight at Mach 0.5 and altitude 3000 meters with penalty matrices

$$Q = \begin{bmatrix} 10 & 0 \\ 0 & 0.5 \end{bmatrix}, \quad R = 1$$

and $P$ from the associated LQ solution. The penalty on the reference has been selected to $\beta = 1000$. The controller has soft state constraints using slack variables and we have used one slack variable for all state constraints at each time step in the prediction horizon. Each slack variable has a linear penalty (effectively a one-norm penalty) and the penalty on each variable is selected to $\beta_\epsilon = 100$.

The performance of the MPC controller has been evaluated and compared to a simple LQ controller with integral action using maximum angle of attack pitch commands. The nonlinearities of the model make the dynamics different at high angles of attack. The effect of the model error is counteracted using integral control with a disturbance observer. The disturbance is modeled as constant and acting on both states, angle of attack and pitch rate. The disturbance is estimated using a Kalman filter with penalty matrices

$$Q_{KF} = 0.2I, \quad R_{KF} = 0.001I$$

in which both states are assumed to be measured.

A comparison of the proposed MPC controller and an LQ controller is shown in Figure 5.10. As we can see from the figure the MPC controller performs better than the LQ controller at high angles of attack. It manages to track the reference and respect the maximum limit on the angle of attack. The LQ controller on the other hand has a large overshoot in the angle of attack even though it is tuned to good performance for the linear dynamics. This is mainly due to the change in the dynamics as the angle of attack increases. This test indicate that it might be sufficient to use a linear MPC controller with integral action to handle the typical nonlinearities that exist over the angle of attack range.

Let us now consider the performance of the proposed linear MPC controller when we change envelope point. Since a change of envelope point, especially the variation in speed, changes the dynamics we can expect the linear MPC controller to perform worse. We tested the controller at Mach 0.4 and altitude 3000 meters and the result is shown in Figure 5.11. From the figure we can see that the performance is slightly worse but still fairly good. The shift in speed caused the pressure center on the aircraft to move forward making it a bit more unstable and from the response we can see that it is slightly less damped with larger oscillations in the control signal. Despite the worse performance the MPC controller still manages to keep the angle of attack response within

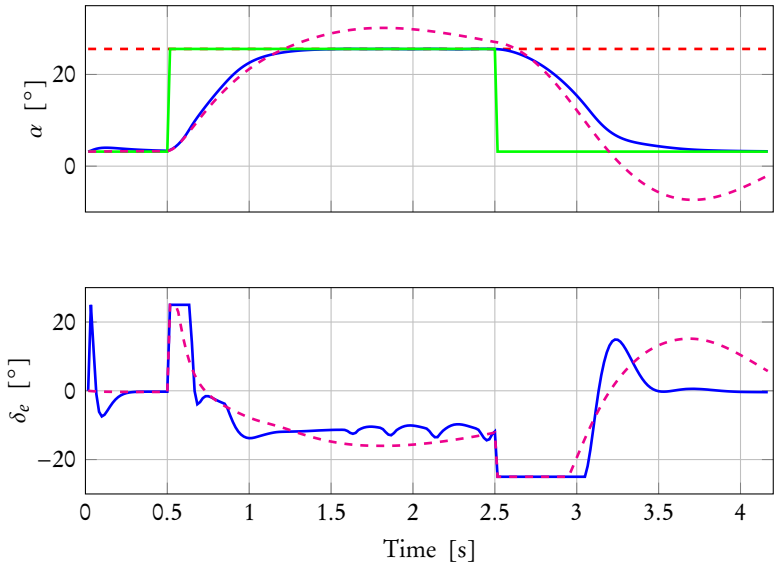


**Figure 5.10.** A comparison between the MPC controller (solid line) and a simple feedback feedforward LQ controller (dashed line) in high angle of attack maneuvers on the nonlinear ADMIRE model.

the desired maximum limit. To overcome this performance degradation we suggest that the MPC controller is scheduled over the flight envelope, with a new dynamic model and penalty matrices in each selected design point.

### 5.3.3 Helicopter flight envelope protection

In this example we apply the developed method to control the forward speed of a Yamaha R-max helicopter using the continuous five state model linearized around hover flight from Mettler [2002]

$$\begin{bmatrix} \dot{v} \\ \dot{q} \\ \dot{\theta} \\ \dot{a} \\ \dot{c} \end{bmatrix} = \begin{bmatrix} -0.0505 & 0 & -9.81 & -9.81 & 0 \\ -0.0561 & 0 & 0 & 82.6 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & -21.7391 & 14 \\ 0 & -1 & 0 & 0 & -0.342 \end{bmatrix} \begin{bmatrix} v \\ q \\ \theta \\ a \\ c \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ -2.174 \\ -0.7573 \end{bmatrix} \delta_s \qquad (5.22)$$

which we have discretized with a sample rate of 60Hz. The states consist of the fuselage motion, i.e., the forward speed, $v$, the pitch rate, $q$ and the pitch angle, $\theta$, and also two states for the rotor dynamics. The first state of the rotor dynamics, $a$, is the pitch angle of the virtual rotor disc that is formed from the rotor blade rotation. The second state, $c$, is corresponding angle for the stabilizer bar. The control signal input, $\delta_s$, is the so called swash plate angle.

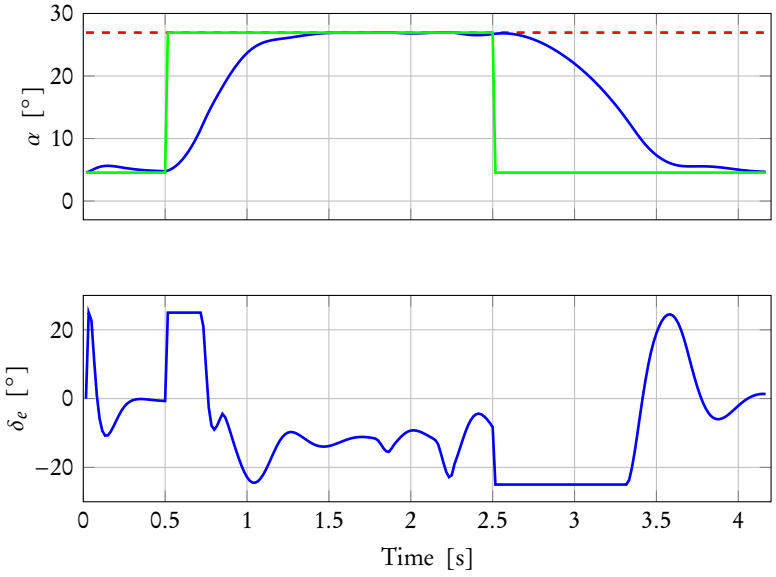


**Figure 5.11.** The proposed MPC tracking controller tuned for Mach 0.5 at altitude 3000 m and simulated in closed loop with the nonlinear ADMIRE model at Mach 0.4 and altitude 3000 m.

The state constraints are formed by upper and lower limits on each variable

$$-5 \leq v \leq 10, \quad |q| \leq 5, \quad |\theta| \leq 3, \quad |a| \leq 1, \quad |c| \leq 2$$

additionally there are upper and lower limits on the control signal, $|\delta_s| \leq 5$.

In this example we have set the prediction horizon to $N = 10$, the penalties in the cost function to

$$W = 10^4, \quad Q = \begin{bmatrix} 50 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}, \quad R = 1$$

and $P$ again as the Lyapunov cost of the corresponding LQ controller. In this setup the QP problem has 72 variables and 296 constraints, out of which 96 constraints come from the terminal constraint set. As a comparison the terminal constraint set from the method of Limon et al. [2008] has 354 constraints.

Figure 5.12 shows how the variable $\lambda_k$ varies over time when the system tracks a given reference input. From the figure we can see that the main changes in the scaling variable come from the changes in the reference signal (compare with Figure 5.13). When the reference is changed from 0 to 9 m/s at time 1 second, then $\lambda$ is increased to its maximum size in order to minimize the

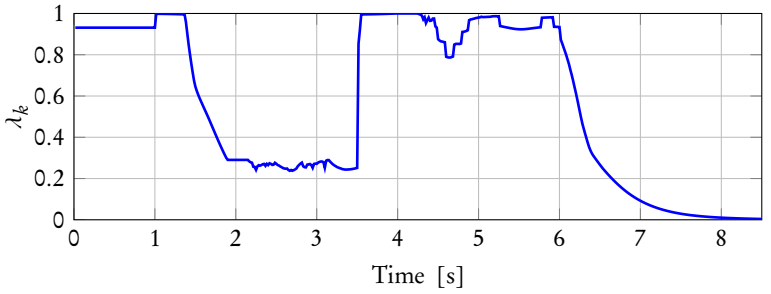


**Figure 5.12.** The variation of the scaling variable, $\lambda$, over time as the system tracks a given speed reference.

difference between the pseudo reference and the true reference. The same happens at time 3.5 seconds when the reference is set back to 0. Towards the end of the simulation the pseudo reference pushes the set point out to the border of the feasible set and hence $\lambda$ is reduced to zero, scaling the terminal constraint set down to a single point. The small changes, or scattering, in $\lambda$ that occur between 2 and 3.5 seconds and also between 4 and 6 seconds depends on that the choice of $\lambda$ is not unique and hence depends on the algorithm used to solve the QP. Note that this scattering in $\lambda$ is internal in the controller and does not affect the applied control signal.

The step responses for the different state variables and also the optimal control signal are shown in Figure 5.13. The upper figure show the speed reference in red, the pseudo reference in green, the actual speed, $v$, in blue and the pitch angle, $\theta$, and pitch rate, $q$, in magenta and cyan respectively. The middle figure shows the rotor disc pitch angle, $a$, in blue and the stabilizer bar pitch angle, $c$, in green. The bottom figure shows the control input, $\delta_s$.

The controller is able to track the speed reference as long as it is a feasible reference. At 6 seconds the speed reference changes to -10 m/s, which is outside the feasible set and then the pseudo reference converges to the closest feasible value.

On the contrary to the previous example it is not the output, $v$, that is the critical variable to limit. Instead it is the pitch angle, $\theta$, and pitch rate, $q$, that we want to limit while tracking the speed as good as possible. From the figure we can see that the pitch rate reaches its upper and lower limits during the transients of the different step responses.

It is worth noting that the control signal seems to have a fairly aggressive nature, but this comes primarily from the tuning of the controller and can be reduced if the controller is retuned. However we have tuned the controller such that the properties of the algorithm should be prominent.

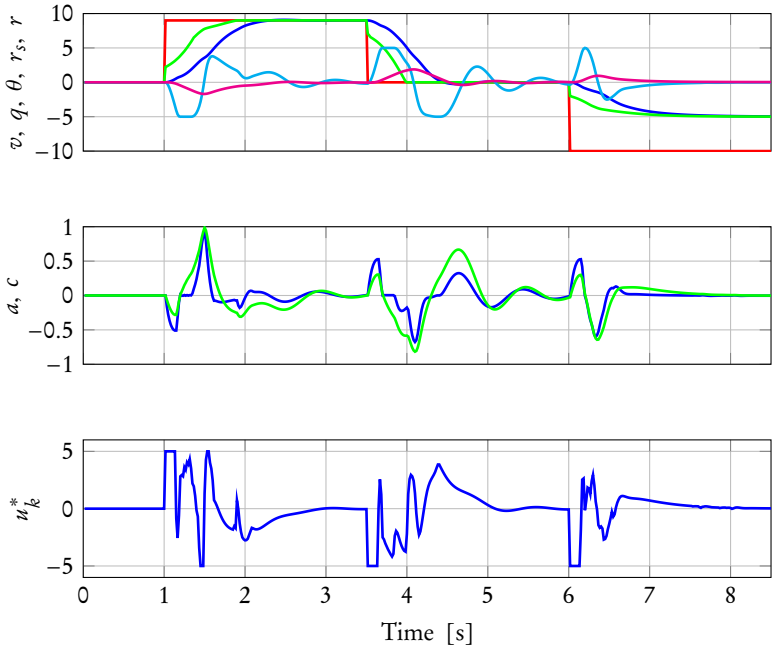


**Figure 5.13.** Response of the helicopter subjected to step changes in the speed reference. The upper most axis show the fuselage states and the reference, the middle axis show the rotor states and the bottom axis show the control signal.

# 6

# Method for guaranteed stability and recursive feasibility in nonlinear MPC

The main drawback with MPC is that it requires an iterative online solution of an optimization problem. This is in general fairly computationally expensive and has so far limited MPC's practical use for nonlinear systems.

To reduce the computational burden of nonlinear MPC, feedback linearization together with linear MPC has been successfully used to control nonlinear systems. The feedback linearization is used as an inner loop to obtain linear dynamics from input to output of the reference system in the MPC controller. The MPC controller is then used as an outer loop to obtain desired dynamics and constraint satisfaction. The main drawback is that this in general results in an optimization problem with nonlinear constraints on the control signal.

Several methods to approximate the nonlinear constraints have been proposed in the literature, many working in an ad hoc fashion, resulting in conservatism, or worse, inability to guarantee recursive feasibility. Also several methods work in an iterative manner, which can be quite time consuming making them inappropriate for fast real time applications.

In this chapter we propose a method to handle the nonlinear constraints, using a set of dynamically generated local inner polytopic approximations. The main benefit of the proposed method is that while computationally cheap it can still guarantee recursive feasibility and convergence.

This chapter is an edited and extended version of the following conference paper

> Daniel Simon, Johan Löfberg, and Torkel Glad. Nonlinear Model Predictive Control using Feedback Linearization and Local Inner

Convex Constraint Approximations. In *Proceedings of the 2013 European Control Conference*, pages 2056–2061, 2013.

## 6.1 Introduction

As described in Section 4.3, when controlling nonlinear systems the resulting MPC optimization problem will be a nonconvex problem, which in general is quite difficult to solve online at high frequency. For certain nonlinear systems one way to handle this is to first create a linear response from (a virtual) input to the output of the system and then design the MPC controller for the system with the virtual input [Del Re et al., 1993]. This can be accomplished, if the system is feedback linearizable (or input to output linearizable), with inner feedback loop [Khalil, 2002] of the form

$$u = \gamma(x, \tilde{u}) \tag{6.1}$$

resulting in a closed loop system which is linear from $\tilde{u}$ to $y$

$$\dot{z} = Az + B\tilde{u}, \quad y = Cz \tag{6.2}$$

Note that the feedback linearization is in continuous time while MPC is in discrete time. Due to this discrete time implementation there will be a prediction error in the models, i.e., the state at next time step will not be exactly what the linearized model predicts. This issue has, to the authors knowledge, not been addressed so far in literature and is interesting for future research. However, in this chapter we have made the assumption that the controller is sampled fast enough such that any such sampling affects are neglectable. This kind of assumption is often valid in aircraft control.

We will outline the details of computing $\gamma(x, \tilde{u})$ in section 6.1.1, but we will first discuss some issues with the use of feedback linearization together with MPC.

Firstly, even if the original cost function (4.34a) is convex the resulting cost function expressed in $\tilde{u}$ can possibly be a nonconvex function. One could simply ignore this complication and formulate a new cost function, which is quadratic in $\tilde{u}$. The performance trade off is analyzed in Primbs and Nevistic [1997] with the conclusion that this approximation is justified only when the complete problem can be formulated as a QP.

Secondly, and in our view a much more problematic issue, is that even simple control signal constraints as

$$\underline{u} \leq u_{k+i} \leq \overline{u} \tag{6.3}$$

will transform into a nonlinear and state dependent constraints on $\tilde{u}$ using (6.1) according to

$$\pi(x_{k+i}, \underline{u}) \leq \tilde{u}_{k+i} \leq \pi(x_{k+i}, \overline{u}) \tag{6.4}$$

Several different methods to handle this have been presented in the literature.

In, e.g., Deng et al. [2009] the authors calculate the exact input constraints at time $k$ and use them as constraints on the whole prediction horizon, an ad-hoc procedure which clearly does not guarantee recursive feasibility. Other authors such as Kothare et al. [1995], Kurtz and Henson [2010], Margellos and Lygeros [2010], Nevistic and Del Re [1994] propose to use the solution sequence from the previous time step to construct an approximation of the nonlinear constraints. These methods can guarantee stability under some strict assumptions, e.g., recursive feasibility. Despite this they can be quite computationally expensive if they work in an iterative manner, e.g., in Margellos and Lygeros [2010] this approximation is done by iteratively solving the linear MPC problem and in each iteration use the previous solution sequence $\{x^*_{k+i}\}_{i=0}^{N}$ to calculate the input constraints using (6.4). The iterations are cancelled when the solution sequence $\{u^*_{k+i}\}_{i=0}^{N-1}$ has converged. The authors of Kurtz and Henson [2010] use a non-iterative approach to construct the approximation of the nonlinear constraints. In the paper the authors propose to use the optimal solution sequence at time $k-1$, $\{u^*_{k-1+i}\}_{i=0}^{N-1}$ to construct a feasible solution at time $k$ as $\{\hat{u}_{k-1+i}\}_{i=0}^{N} = \left\{\{u^*_{k-1+i}\}_{i=0}^{N-1}, 0\right\}$ which is used to predict the future trajectory $\{\hat{x}_{k+i}\}_{i=1}^{N}$ and from this reconstruct the nonlinear state dependent constraints.

In this chapter we will adopt a different approach to handle the nonlinear constraints based on using the exact constraints for the current time step and a set of inner polytope approximations for future time steps.

In the recent paper by Pant et al. [2016] this method has been extended to also address the case when the states of the nonlinear system are available only through a state estimate and when the original states are not preserved in the feedback linearization. In [Deori et al., 2015] the authors apply a similar type of algorithm to the air traffic management problem. This approach has also been considered as a possible strategy for heating, ventilation and air conditioning in energy efficient buildings in the survey paper by Ercan [2016].

### 6.1.1 Feedback linearization

Let us, before we outline the proposed controller, describe the feedback linearization scheme.

If we consider affine-in-control nonlinear systems of the form

$$\dot{x} = f(x) + g(x)u, \quad y = h(x) \tag{6.5}$$

and define the *Lie derivative* in the direction of $f$ as

$$L_f = \sum_{i=1}^{n} f_i(x) \frac{\partial}{\partial x_i}$$

then, if we repeatedly differentiate the output, we obtain $\dot{y} = L_f h$, $\ddot{y} = L_f(L_f h) = L_f^2 h$ etc.

If we assume that the system has $\dim y = \dim u = m$ and apply the Lie derivative then we obtain for the i:th output

$$\begin{aligned}\dot{y}_i &= L_{(f+gu)}h_i = L_{f+u_1g_1+u_2g_2+\ldots+u_mg_m)}h_i \\ &= L_f h_i + u_1 L_{g_1}h_i + u_2 L g_2 h_i + \ldots + u_m L_{g_m}h_i\end{aligned}$$

If all $L_{g_j}h_i = 0$ this means that $\dot{y}_i = L_f h_i$ and no control signal input affects the output derivative. Continuing the differentiation until one of the $L_{g_j}L_f^{\nu_i}h_i \neq 0$, then $u_j$ affects $y_i^{(\nu_i)}$ and we say that the system has a *relative degree* of $\nu_i$ in $x_0$ [Khalil, 2002].

This procedure can be summarized in a *decoupling matrix* $R(x)$ according to

$$R(x) = \begin{bmatrix} L_{g_1}L_f^{\nu_1-1}h_i & \ldots & L_{g_m}L_f^{\nu_1-1}h_i \\ \vdots & & \vdots \\ L_{g_1}L_f^{\nu_m-1}h_m & \ldots & L_{g_m}L_f^{\nu_m-1}h_m \end{bmatrix} \tag{6.6}$$

which gives

$$\begin{bmatrix} y_1^{(\nu_1)} \\ \vdots \\ y_m^{(\nu_m)} \end{bmatrix} = R(x)\begin{bmatrix} u_1 \\ \vdots \\ u_m \end{bmatrix} + \begin{bmatrix} L_f^{\nu_1}h_1 \\ \vdots \\ L_f^{\nu_m}h_m \end{bmatrix}$$

If $R(x)$ is nonsingular then the control signal can be chosen as

$$u = R^{-1}(x)\left(-\begin{bmatrix} L_f^{\nu_1}h_1 \\ \vdots \\ L_f^{\nu_m}h_m \end{bmatrix} + \tilde{u}\right) \tag{6.7}$$

and the resulting closed loop system will be linear and decoupled from $\tilde{u}$ to $y$ Khalil [2002].

This procedure works well when the zero dynamics is stable or when there is no zero dynamics, i.e., when the relative degree is equal to the state dimension. In these cases the system does not need to be transformed into a controllable canonical form [Khalil, 2002] and the original system states can be kept which is especially good if we have constraints on the states.

In the rest of this paper we restrict our discussion to these kinds of systems that allow us to keep our original states. The aircraft example in Section 6.3.2 motivates this assumption.

## 6.2 The proposed algorithm

First we make the following assumptions.

**Assumption 6.1.** The nonlinear system (6.5) is input-output feedback linearizable using the control (6.7) and the linearized system has a discrete-time state-space description

$$x_{k+1} = Ax_k + B\tilde{u}_k \tag{6.8}$$

with no unstable zero dynamics.

**Assumption 6.2.** The functions $\Psi(\cdot)$ and $\ell(\cdot)$, in (4.34a), are such that they satisfy the necessary conditions of Theorem 4.2, the sets $\mathcal{X}$ and $\mathcal{U}$, in (4.34c) and (4.34d), are convex polytopes and for simplicity we assume $\mathcal{U}$ to be simple bounds on the control signal.

Applying an inner loop feedback linearization as described in the previous section to a nonlinear system of the form (6.5) and then MPC as an outer loop controller, the resulting MPC problem setup is

$$\underset{\tilde{u}}{\text{minimize}} \;\; \Psi(x_{k+N}) + \sum_{i=0}^{N-1} \ell(x_{k+i}, \tilde{u}_{k+i}) \tag{6.9a}$$

subj. to

$$x_{k+i+1} = Ax_{k+i} + B\tilde{u}_{k+i} \tag{6.9b}$$

$$x_{k+i} \in \mathcal{X} \tag{6.9c}$$

$$x_{k+N} \in \mathcal{T} \tag{6.9d}$$

$$\tilde{u}_{k+i} \in \Pi \tag{6.9e}$$

where we have defined

$$\Pi = \left\{ \tilde{u}_{k+i} \mid \pi(x_{k+i}, \underline{u}) \leq \tilde{u}_{k+i} \leq \pi(x_{k+i}, \overline{u}) \right\}$$

where the functions $\pi(\cdot)$ are the nonlinear constraints that arise from the feedback linearization (6.7).

### 6.2.1 Nonlinear constraint approximations

To begin with, since the current state, $x_k$, is known, the exact nonlinear constraint on $\tilde{u}_k$ can be calculated as

$$\pi(x_k, \underline{u}) \leq \tilde{u}_k \leq \pi(x_k, \overline{u})$$

Obviously, since this is a linear constraint in $\tilde{u}_k$, our scheme should be able to use this exactly without resorting to any conservative approximation. It is thus our goal to derive an algorithm where this constraint is used exactly, and the constraints on future control signals are included in an as non-conservative fashion as possible, while guaranteeing stability and feasibility.

A first step to handle the future time steps of the nonlinear constraint (6.9e) is to simply replace it with a global inner convex polytopic approximation

$$\mathcal{G} = \{(x, \tilde{u}) \mid x \in \mathcal{X},\; g_l(\mathcal{X}) \leq \tilde{u} \leq g_u(\mathcal{X})\} \tag{6.10}$$

where $g_u(\mathcal{X})$ is a concave piecewise affine function such that $g_u(\mathcal{X}) \leq \pi(\mathcal{X}, \overline{u})$ and $g_l(\mathcal{X})$ is a convex piecewise affine function such that $g_l(\mathcal{X}) \geq \pi(\mathcal{X}, \underline{u})$. An example of an inner approximation, $\mathcal{G}$, is shown in Figure 6.1. Note that this approximation is not unique and the degree of suboptimality vary with the method of approximation.

If the nonlinear constraints (6.9e) form a highly nonconvex set, then it is fair to assume that $\mathcal{G}$ poorly approximates the true nonlinear constraints over the entire state space, i.e., it can only be close to the true constraints in some, possibly small, regions, cutting of control authority in other regions. An example of this is shown in Figure 6.2. This motivates us to not use a global approximation for all time steps in the control signal sequence.

If one makes use of the fact that the true constraints are known at time $k$ it is easy to calculate the bounded evolution of the system to time $k+1$ and therefore all possible states, $\mathcal{X}_{k+1}$. It is then obvious that for this limited subset of the state-space there might exist a better inner convex approximation of the nonlinear constraints than the global approximation $\mathcal{G}$. Hence, we would like to construct a convex polytope, $\mathcal{I}$, over the set $\mathcal{X}_{k+1}$ and constrain $(x_{k+1}, \tilde{u}_{k+1})$ to this local approximation.

This procedure can of course be repeated for time step $k+2, k+3, \ldots, k+N-1$, generating a new local polytope for each $(x_{k+i}, \tilde{u}_{k+i})$. A significant problem will however occur if one tries to prove recursive feasibility of this scheme. Since we always use the exact constraint for the first control input, this conflicts with our inner approximation that was used for future control input, when we shift the horizon in standard MPC stability and recursive feasibility proofs. If we use the full control authority in the next time instant, the state predictions arising from that set will move outside the predictions that were used in the previous time instant when predictions were based on an inner approximation of the control input at $k+1$. To account for this, a scheme based on both inner approximations of control inputs for actual control decisions, and outer approximations of control inputs to perform the propagation of state bounds, will be used.

The local constraint approximations are constructed as inner convex approximations of the nonlinear constraints based on reachable sets.

**Definition 6.3.** At time $k$, the outer approximation of the i:th step reachable set $\mathcal{X}_{k+i}$ is recursively defined as

$$\mathcal{X}_{k+i} = A\mathcal{X}_{k+i-1} + B\mathcal{C}_{k+i-1}$$

where

$$\mathcal{X}_k = \{x_k\}$$

The set $\mathcal{C}_{k+i}$ is an outer polytopic approximation of the nonlinear control

constraints in the reachable set $\mathcal{X}_{k+i}$, i.e.,

$$\mathcal{C}_{k+i} = \left\{\tilde{u}_{k+i} \mid \omega_l^{k+i}(\mathcal{X}_{k+i}) \leq \tilde{u}_{k+i} \leq \omega_u^{k+i}(\mathcal{X}_{k+i})\right\}$$

where $\omega_u^{k+i}(\cdot)$ is a concave piecewise affine function such that

$$\omega_u^{k+i}(\mathcal{X}_{k+1}) \geq \pi(\mathcal{X}_{k+1}, \overline{u})$$

and $\omega_l^{k+i}(\cdot)$ is a convex piecewise affine function such that

$$\pi(\mathcal{X}_{k+1}, \underline{u}) \geq \omega_l^{k+i}(\mathcal{X}_{k+1})$$

The initial outer approximation, $\mathcal{C}_k$, is the exact control constraints, i.e.,

$$\mathcal{C}_k = \left\{u_k \mid \pi(x_k, \underline{u}) \leq \tilde{u}_k \leq \pi(x_k, \overline{u})\right\}$$

It should be noted here that we do not specify how to construct the sets $\mathcal{C}_{k+i}$, just constraints on how they can be constructed. The user is free to chose any method he or she may find suitable and the stability and feasibility of the algorithm holds regardless of the chosen method.

From the i:th step reachable set the local convex approximation, $\mathcal{I}_i^k$, $i$ step ahead at time $k$ can now be constructed as the polytope defined from the constraints

$$\begin{gathered} h_l^{k+i}(\mathcal{X}_{k+i} \cap \mathcal{X}) \leq \tilde{u}_{k+i} \leq h_u^{k+i}(\mathcal{X}_{k+i} \cap \mathcal{X}) \\ x_{k+i} \in \mathcal{X}_{k+i} \cap \mathcal{X} \end{gathered}$$

where $h_u^{k+i}(\cdot)$ is a concave piecewise affine function such

$$g_u(\mathcal{X}_{k+1}) \leq h_u^{k+i}(\mathcal{X}_{k+1}) \leq \pi(\mathcal{X}_{k+1}, \overline{u})$$

and $h_l^{k+i}(\cdot)$ is a convex piecewise affine function such

$$\pi(\mathcal{X}_{k+1}, \underline{u}) \leq h_l^{k+i}(\mathcal{X}_{k+1}) \leq g_l(\mathcal{X}_{k+1})$$

In other words, the local polytope, $\mathcal{I}_i^k$, shall be an inner approximation to the nonlinear constraints and on the subset $\mathcal{X}_{k+i}$ it shall hold that $\mathcal{G} \subseteq \mathcal{I}_i^k$, which can always be achieved.

Figure 6.1 shows an example that illustrates the relationship between the local polytopes, the global polytope and the nonlinear constraints. Note that as for the global inner convex approximation this construction is non unique, in this thesis we have used a tangent plane for the concave surfaces and a piecewise affine approximation of the convex surfaces (described further in the examples).

### 6.2.2 MPC receding horizon setup

Now let us summarize the discussion above into our proposed MPC algorithm. At each sample time $k$ solve the NMPC problem given by Algorithm 2.

The state and control signals are constrained to the local polytopes up to

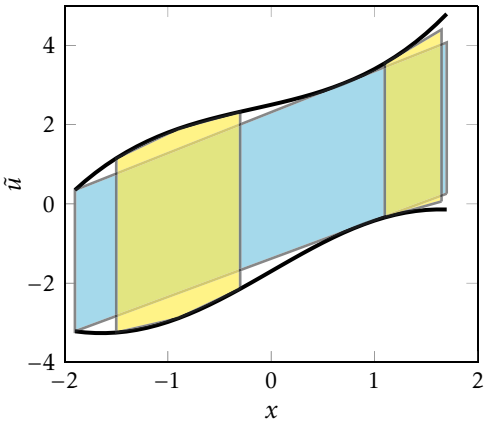


**Figure 6.1.** Example showing the nonlinear constraints on $\tilde{u}$ as upper and lower bound, the global approximation, $\mathcal{G}$, in cyan (dark shaded) and two local approximations, $\mathcal{I}_1^k$, for different $x_k$ in yellow.

horizon $N_l \leq N-1$ and constrained to the global polytope for $N_l < i < N$. The introduction of the horizon $N_l$ is to highlight that, depending on the problem, there might not be any performance gain in using the local polytopes for the entire horizon, $N-1$, so instead the fixed global inner approximation is used for the last part of the horizon. We will see an example of this in the next section. The final constraint set $\mathcal{T}$ is an invariant set as defined in Definition 4.1 and it is calculated using the global inner polytope approximation, $\mathcal{G}$, as the initial bounds on $x$ and $\tilde{u}$.

The practical advantage of this approach of shifting the local polytopes in time, as in (6.12), is that only one local approximation has to be calculated in each iteration. The theoretical advantage is that this also guarantees recursive feasibility.

We can now state the main stability result for this controller.

**Theorem 6.4.** *For any initially feasible state $x_0$, the MPC controller defined by Algorithm 2 remains feasible and stabilizes the system (6.5).*

**Proof:** To show recursive feasibility, let us denote the set of states where (6.11) is feasible with $\mathcal{F}$. Assume that $x_k \in \mathcal{F}$ and that (6.11) has the optimal solution sequence $\{\tilde{u}^*_{k+i}\}_{i=0}^{N-1}$.

We now claim that a feasible solution at time $k+1$ is to use the control sequence $\{\hat{u}^*_{k+i}\}_{i=1}^{N} = \left\{\{\tilde{u}^*_{k+i}\}_{i=1}^{N-1}, \kappa(x^*_N)\right\}$ where $\kappa(x)$ is a local controller as defined in Section 4.2.1.

To see that this is a feasible solution we first note that since $x_{k+N} \in \mathcal{T}$ we can select $\hat{u}_{k+N} = \kappa(x^*_N)$ since this will ensure that $x_{k+N+1} \in \mathcal{T}$ and all constraints

**Algorithm 2** Approximate NMPC algorithm

1: given measurement $x_k$, and a set of approximations, $\mathcal{I}_i^k$ and $\mathcal{G}$
2: Solve the QP

$$\underset{\tilde{u}}{\text{minimize}} \ \Psi(x_{k+N}) + \sum_{i=0}^{N-1} \ell(x_{k+i}, \tilde{u}_{k+i}) \tag{6.11a}$$

subj. to

$$x_{k+i+1} = Ax_{k+i} + B\tilde{u}_{k+i} \quad \forall\, i = 0, \ldots, N-1 \tag{6.11b}$$
$$\pi(x_k, \underline{u}) \leq \tilde{u}_k \leq \pi(x_k, \overline{u}) \tag{6.11c}$$
$$(x_{k+i}, \tilde{u}_{k+i}) \in \mathcal{I}_i^k \ \ \forall\, i = 1, \ldots, N_l \tag{6.11d}$$
$$(x_{k+i}, \tilde{u}_{k+i}) \in \mathcal{G} \ \ \forall\, i = N_l + 1, \ldots, N-1 \tag{6.11e}$$
$$x_{k+N} \in \mathcal{T} \tag{6.11f}$$

3: calculate the control signal as $u_k = \gamma(x, \tilde{u}(1))$ from (6.7)
4: update the local approximations $\mathcal{I}_i^{k+1}$ as

$$\mathcal{I}_i^{k+1} = \mathcal{I}_{i+1}^k \ \ \forall\, i = 1, \ldots, N_l - 1 \tag{6.12}$$

5: construct a new set $\mathcal{I}_{N_l}^{k+1}$ from the procedure in Section 6.2.1
6: repeat from 1.

are satisfied at $k + N + 1$. Also we note that since $\tilde{u}_{k+1}^* \in \mathcal{I}_1^k \subset \Pi$ this means that $\pi(x_{k+1}, \underline{u}) \leq \tilde{u}_{k+1}^* \leq \pi(x_{k+1}, \overline{u})$ is feasible at time $k + 1$.

Furthermore we have that all $\tilde{u}_{k+i}^*$, $i = 2, \ldots, N_l$ are feasible at time $k + 1$ since the local approximations are shifted one time step (6.12). The control $\tilde{u}_{k+N_l+1}^* \in \mathcal{G}$ at time $k$ are also feasible at time $k + 1$ since $\tilde{u}_{k+N_l+1}^* \in \mathcal{I}_{N_l}^{k+1} \supseteq \mathcal{G}$ in $\mathcal{X}_{k+N_l+1}$ at time $k + 1$ and $\mathcal{I}_{N_l}^{k+1}$ is always a nonempty set if the problem is initially feasible. All other $\tilde{u}_{k+N_l+i}^* \in \mathcal{G}$ are trivially feasible.

Convergence of the proposed algorithm is not affected by the local approximations and the standard proof from Section 4.2.1 hold without any change. This means that the controller defined through Algorithm 2 stabilizes the system (6.8) and by the Assumption 6.1 it also stabilizes the nonlinear system (6.5). □

## 6.3 Examples

In this section we present two examples to illustrate the properties of the proposed algorithm. In the first example we consider a fictitious nonlinear system whose purpose is to illustrate the generation and propagation of the local polytopes.

In the second example we consider the task of controlling a fighter aircraft that

has nonlinear unstable dynamics. The purpose of this example is to illustrate the degree of suboptimality for the proposed method.

The implementation and simulation has been performed in MATLAB with YALMIP [Löfberg, 2004] and MPT [Herceg et al., 2013].

### 6.3.1 Illustrative scalar example

Consider a nonlinear system given by

$$\dot{x} = 1.8x + \left(0.2x^4 + 0.875\right) u$$
$$y = x$$

with the constraints $-2 \leq x \leq 2$, and $-2 \leq u \leq 2$. Following the procedure in Section 6.1.1 we obtain the feedback linearization control law

$$u = \frac{1}{0.2x^4 + 0.875} \left(\tilde{u} - 1.8x\right)$$

and the resulting linear system is an integrator from $\tilde{u}$ to $y$.

The nonlinear feedback gives the following nonlinear control constraints on $\tilde{u}$

$$-0.4x^4 + 1.8x - 1.75 \leq \tilde{u} \leq 0.4x^4 + 1.8x + 1.75 \tag{6.13}$$

shown in Figure 6.2 together with the global approximation $\mathcal{G}$. Note the massive loss of control authority at, e.g., $x_k = \pm 2$, if only the global approximation $\mathcal{G}$ is used to constrain $\tilde{u}$.

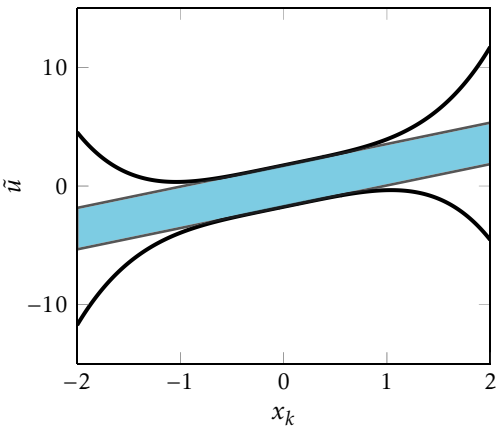


**Figure 6.2.** Nonlinear constraints on $\tilde{u}$ due to feedback linearization and a global inner approximation $\mathcal{G}$.

Algorithm 2 is applied to the discrete-time version of the integrator system with sample time 0.4s. Using $N = 5$ and $N_l = 4$, i.e., we use the global polytope $\mathcal{G}$ only to calculate the terminal constraint set, $\mathcal{T}$, and the objective

is to control the system to the origin. The local polytopes are calculated from the tangent line at the center point in the reachable set.

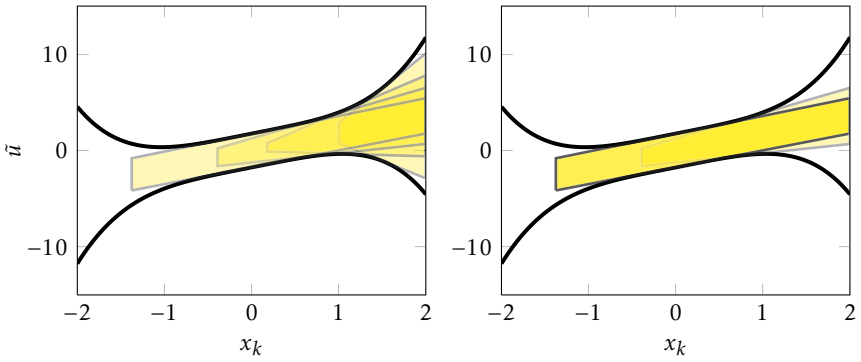


**Figure 6.3.** The left axis shows the nonlinear constraints on $\tilde{u}$ and the local polytopes $\mathcal{I}_i^0$ for $i = 1, \ldots, 4$ at time $k = 0$. The right axis shows the local polytopes $\mathcal{I}_i^2$ at time $k = 2$.

Starting in $x = 1.9$ the generated local polytopes at time $k = 0$, $\mathcal{I}_i^0$, are shown in the left part Figure 6.3. It clearly demonstrates the increased control signal ability compared to only using the global approximation as in Figure 6.2.

At the next time step the first local approximation, $\mathcal{I}_1^0$, is discarded. All other polytopes are shifted one step, i.e., $\mathcal{I}_1^1 = \mathcal{I}_2^0$, $\mathcal{I}_2^1 = \mathcal{I}_3^0$, etc. A new local polytope, $\mathcal{I}_4^1$, is generated at the end of the sequence. The right part of Figure 6.3 shows the local approximations at time $k = 2$. At this point the state has moved to $x_k \approx 0.6$ and an accurate approximation in the region around $x = 1.9$ is no longer of importance. In the figure one can see that the polytope $\mathcal{I}_3^0$ has been shifted and is now, at time $k = 2$, the polytope $\mathcal{I}_1^2$. The same holds for $\mathcal{I}_4^0 = \mathcal{I}_2^2$.

### 6.3.2 Nonlinear aircraft example

In this example we consider a continuous time nonlinear model of the same aircraft as in Section 5.3.1. The nonlinearity consists of an extra term in the moment equation (6.14b) which is proportional to the square of the angle of attack.

$$\dot{\alpha} = -1.8151\alpha + 0.9605q \tag{6.14a}$$

$$\dot{q} = 0.15\alpha^2 + 12.9708\alpha - 1.8988q + 19.8474\delta_e \tag{6.14b}$$

$$y = \alpha \tag{6.14c}$$

The states are the same as in Section 5.3.1 where $q$ is the angular rate in pitch and $\alpha$ the angle between the aircraft x-axis and the velocity vector.

The coefficients of the linear terms have been selected to correspond to the linearized dynamics of the ADMIRE model at Mach 0.6 and altitude 1000 m, for details see Forssell and Nilsson [2005]. The coefficient for the $\alpha^2$-term is selected to make the $\alpha$-contribution to the moment equation in the nonlinear model approximately 15% larger at $\alpha = 30°$ compared to the linearized model. The constraints on the system are basic control surface deflection limits $|\delta_e| \leq 25°$ and also limits on the angle of attack $-10° \leq \alpha \leq 30°$.

For the system (6.14) it is easy to see that by selecting the nonlinear feedback as

$$\delta_e = \tilde{u} - 0.0076\alpha^2 \tag{6.15}$$

the closed loop system from MPC control input, $\tilde{u}$, to the output, $\alpha$ will be linear.

$$\dot{\alpha} = -1.8151\alpha + 0.9605q \tag{6.16a}$$

$$\dot{q} = 12.9708\alpha - 1.8988q + 19.8474\tilde{u} \tag{6.16b}$$

We can now formulate an MPC problem for the system (6.16) on the form (6.11) where we chose to use a cost function that is quadratic in $\tilde{u}$ since the goal is to end up in a standard QP problem. In the cost function we used the tuning parameters

$$Q = \begin{bmatrix} 5 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 2$$

and the sample time is 1/60 second.

Note that the state constraints are still linear after the feedback linearization but the control constraints are now nonlinear and state dependent.

$$-25° + 0.0076\alpha_{k+i}^2 \leq \tilde{u}_{k+i} \leq 25° + 0.0076\alpha_{k+i}^2 \tag{6.17}$$

In this case both the lower and upper bound is convex and therefore form a nonconvex set. It is therefore difficult to make a good global inner convex polytope approximation, $\mathcal{G}$, of these constraints and control authority is lost around the angle of attack limits, see Figure 6.4.

The lower bound can be approximated arbitrarily well with increasing complexity of the polytope while at the upper bound we cut away control performance around the $\alpha$-limits with the global approximation. This opens up for the possibility that control performance could be gained using local approximations of the constraints when the current state is close to the $\alpha$-limits.

If we compare Algorithm 2, with $N_l = 1$, with a global nonlinear solver (the built-in branch and bound solver in YALMIP) for the problem (6.9), we obtain a measure of the performance loss, i.e., the suboptimality of our algorithm compared to using the exact nonlinear constraints. We have compared the open loop optimal cost of the two algorithms and as performance measure we use

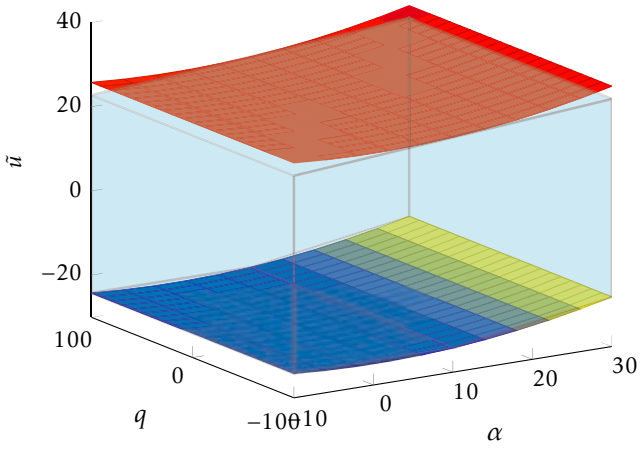


**Figure 6.4.** Nonlinear control signal constraints (6.17) and a global inner convex polytope approximation, $\mathcal{G}$

the relative error

$$\eta = \frac{\left|V^* - \hat{V}\right|}{|V^*|}$$

between the optimal cost of the proposed algorithm, $\hat{V}$, and the branch and bound solver, $V^*$. Figure 6.5 shows the variation of the relative error over the entire feasible state space.

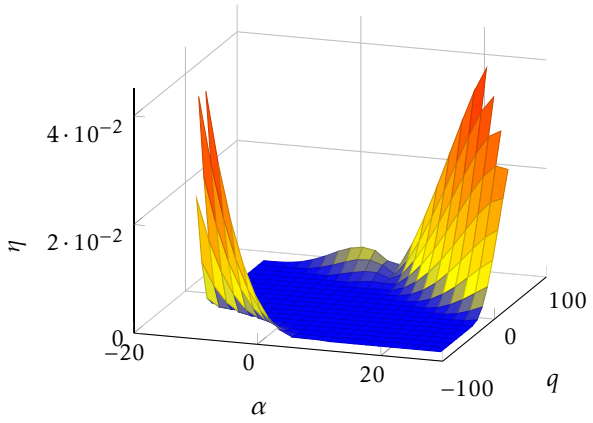


**Figure 6.5.** Variation of the relative error, $\eta$, between proposed algorithm and global solver over the state space.

From the figure we can conclude that for this example the maximum perfor-

mance loss is approximately 4% and that it occurs in areas of the state space where maximum control signals are likely to be used, i.e., large (absolute value) combinations of angles and angular rates.

In the example we use only one step local approximations and a fair question to ask is if any more performance could be gained by increasing the number of local approximations. In Table 6.1 we compare, for a subset of the state space, the relative error for both $N_l = 1$ and $N_l = 10$.

**Table 6.1.** A comparison of performance loss, $\eta$, when the number of local polytopes are increased from $N_l = 1$ to $N_l = 10$.

| $x_0$ | $\begin{pmatrix}-14.0\\-26.5\end{pmatrix}$ | $\begin{pmatrix}-5.0\\-9.5\end{pmatrix}$ | $\begin{pmatrix}15.0\\28.3\end{pmatrix}$ | $\begin{pmatrix}28.0\\52.9\end{pmatrix}$ |
|---|---|---|---|---|
| $N_l = 1$ | $4.1 \cdot 10^{-4}$ | $3.9 \cdot 10^{-14}$ | $7.7 \cdot 10^{-4}$ | $2.5 \cdot 10^{-2}$ |
| $N_l = 10$ | $3.5 \cdot 10^{-11}$ | $8.2 \cdot 10^{-16}$ | $3.5 \cdot 10^{-5}$ | $1.1 \cdot 10^{-2}$ |

The results in Table 6.1 indicate at least a 50% reduction in the relative error of the optimal cost when the number of local polytopes are increased. Although it is a significant decrease in the relative error the absolute values are in both cases relatively small and it is questionable if it gives any practical performance gain.

To evaluate the practical implications of the difference between the proposed controller and the branch and bound method we compare the time response of the closed loop system for the both controllers. We initiate the system at a steady state where it is expected to have a relatively large difference in optimal cost between the two controllers. However in Figure 6.6 we see that the time response of both closed loop systems are identical to the naked eye.

Since there appear to be no difference in the time response we also look at the error between the two time responses. Figure 6.7 reveal that there are very tiny differences in both state and control signals between the two closed loop systems. We can conclude that when there are only small nonlinearities, there is no practical gain in increasing the number of local polytopes, i.e., the important property is that we can use the full control authority for the current control input.

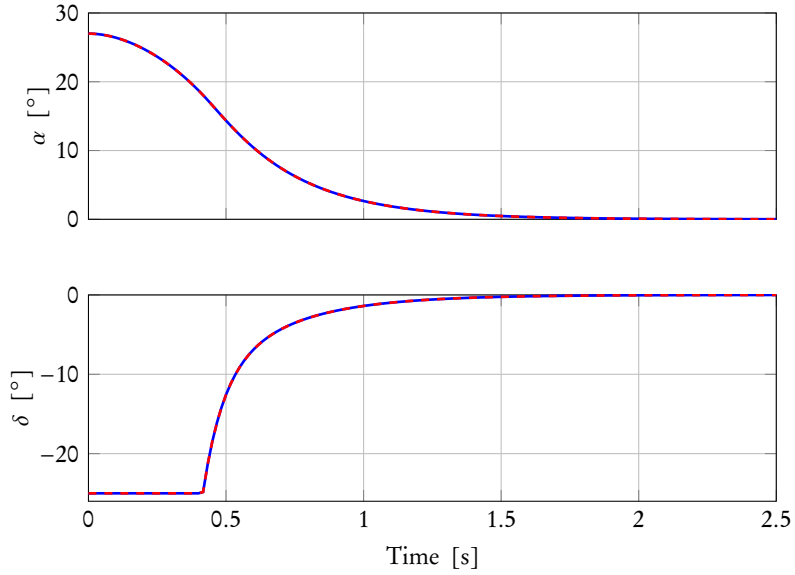


**Figure 6.6.** Time response of the proposed controller (blue solid line) and the branch and bound controller (red dashed line) for the initial condition $x_0 = [27 \quad 51]^T$. The two responses appear identical.

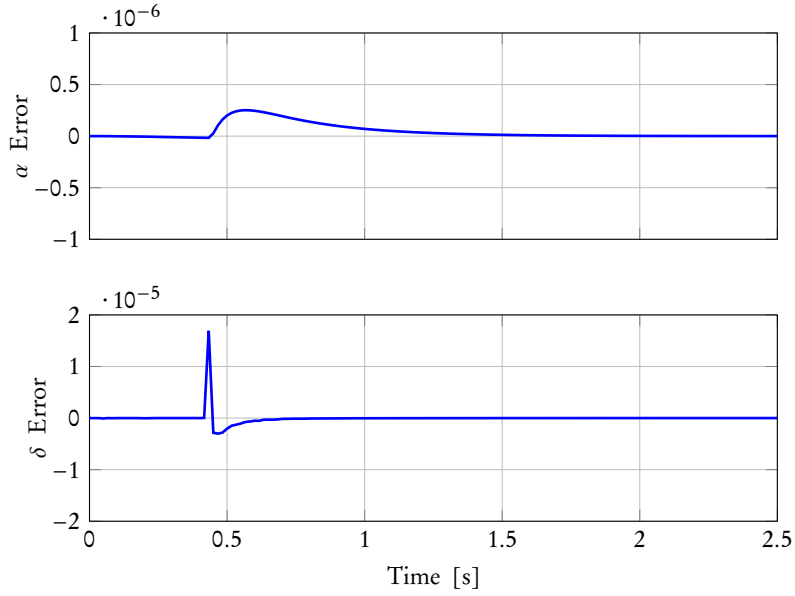


**Figure 6.7.** Time response of the difference in state and control between the two controllers.

# 7

# Testing stability and robustness of MPC controllers

In pervious chapters we have been concerned with the task of deriving model predictive control algorithms which have guaranteed stability properties. In this chapter we instead focus on the task of a posteriori verifying closed loop stability of model predictive controllers for which standard stability proofs can not be applied.

This chapter is based on the results previously published in

> Daniel Simon and Johan Löfberg. Stability analysis of Model Predictive Controllers using Mixed Integer Linear Programming. In *IEEE 55th Conference on Decision and Control*, pages 7270–7275, Las Vegas, 2016.

The section on robust stability analysis consists of previously unpublished material.

## 7.1 Introduction

It is a well known fact that finite time optimal controllers, such as MPC do not necessarily result in closed loop stable systems. It is, as we have discussed in previous chapters, common practice within the MPC community to add a final state constraint and/or a final state penalty in order to obtain guaranteed stability. However, for more advanced controller structures it can be difficult to show stability using these techniques. Additionally as we have seen in Chapter 5 the final state constraint set can consist of so many inequalities that the complexity of the MPC problem is too big for use in certain fast and time critical applications. This motivates us to instead focus on deriving

a tool for analysis of the closed loop stability for linear systems controlled with MPC controllers. We formulate an optimization problem that gives a sufficient condition for stability of the closed loop system and we show that the problem can be written as a *Mixed Integer Linear Programming Problem* (MILP).

We consider a linear model predictive control formulation of the form

$$V_k^* = \min_{x_{k+i}, u_{k+i}} \quad \sum_{i=0}^{N-1} x_{k+i}^T Q x_{k+i} + u_{k+i}^T R u_{k+i} + \Psi(x_{k+N}) \tag{7.1a}$$

$$\text{subj. to} \quad x_{k+i+1} = Ax_{k+i} + Bu_{k+i} \tag{7.1b}$$

$$Ex_{k+i} \leq f \quad i = 1, \ldots, N-1 \tag{7.1c}$$

$$Tx_{k+N} \leq t \tag{7.1d}$$

$$Gu_{k+i} \leq h \quad i = 1, \ldots, N-1 \tag{7.1e}$$

Stability of the MPC control law (7.1) is most often proven a priori by showing that the objective function (7.1a) is a valid Lyapunov function for the closed loop system by designing $\Psi(x_{k+N})$ and the terminal state constraint, $Tx_{k+N} \leq t$, such that

$$V_{k+1}^* - V_k^* \tag{7.2}$$

is guaranteed to be less that zero, see section 4.2.1 for details.

However there exist many MPC formulations, such as move blocking Cagienard et al. [2007] or soft constraints Zeilinger et al. [2014] (i.e., slack), for which it can be difficult to show stability using this standard framework as presented in Mayne et al. [2000]. Even if it allows for the possibility to guarantee stability, the addition of the terminal constraints can add a significant complexity to the original problem that might be unnecessary and limit the applicability of MPC within certain fields. In reference tracking MPC the terminal set can become very complex and some times not even finitely determined, see e.g., Chisci and Zappa [2003] or Limon et al. [2008]. Therefore it can often be beneficial to analyze and verify the stability of a certain design rather than building in the stability by adding extra constraints.

The problem of analyzing stability of optimization-based controllers is no new field. In Primbs and Nevistic [2000] the authors derive a stability test based on bounds of the cost function. The author of Primbs [2001] uses the KKT conditions for the MPC controller (7.1) to derive, using the S-procedure, an LMI that gives a sufficient condition for stability. Several papers have followed up on this idea such as Ahmad et al. [2014], Korda and Jones [2015], Lennox et al. [2008], Li et al. [2006], Løvaas et al. [2007]. These papers have extended the idea to hold for more general cases or improved the complexity of the resulting LMI. E.g., in Korda and Jones [2015] the authors extend the ideas in Primbs [2001] to hold for more general systems and show that given that the

system and its constraints are polynomial, the stability can be analyzed using sum-of-squares programming. In Lennox et al. [2008] the authors derive an LMI of much lower dimension than that of Primbs [2001], which improves on the complexity of the problem. However in order to reduce the size of the LMI the authors of Lennox et al. [2008] need to make the assumptions that the system is stable, i.e., $A$ is hurwitz, and that the constraints on control and states can be written as $LU_k + LNx_k \leq b$. These assumptions are fairly restrictive and not really suitable for our applications.

In this chapter we also exploit the KKT conditions of the MPC problem (7.1) to analyze the closed loop stability. In Section 7.2 we analyze the difference of the value function (7.2) as a proposed Lyapunov function candidate. But instead of deriving an LMI condition we formulate the problem as an indefinite quadratic bilevel optimization problem. We then show that this problem can be rewritten as a Mixed Integer Linear Program (MILP), which can have major computational advantages compared to using the LMI formulation. Later in Section 7.3 we will analyze robust stability and we will have to propose another Lyapunov function candidate and reformulate the stability problem such that we can show closed loop stability of systems subject to additive disturbances.

## 7.2 The MILP stability test

In this section we will derive the proposed stability test. We will formulate an optimization problem that uses the MPC problem's objective function as a candidate Lyapunov function and then minimizes the difference of the Lyapunov function between two consecutive time steps. Since the optimization problem only tests the validity of a certain Lyapunov function candidate the formulated problem can only verify stability, not prove instability. Therefore it is a sufficient but not necessary condition for stability. Nevertheless, in situations where the value function is non-decreasing at some point, it might be an indication of problems in the design and further investigations and simulations need to be made.

The resulting optimization problem is an indefinite quadratic bilevel optimization problem, which is very difficult to solve. We will finally show how this indefinite problem can be rewritten as a mixed integer linear programming problem.

If the objective function (7.1a) is a quadratic or LP-representable function in $x$ and $u$ we can rewrite the MPC problem (7.1), at time step $k$, in a more compact form as

$$\begin{aligned} \underset{U_k}{\text{minimize}} \quad & V_k \\ \text{subj. to} \quad & Ex_k + FU_k \leq b \end{aligned} \tag{7.3}$$

where

$$V_k = \frac{1}{2}U_k^T H U_k + U_k^T G x_k + \frac{1}{2}x_k^T \bar{Q} x_k$$

and where $U_k = [u_k^T, u_{k+1}^T, \ldots, u_{k+N-1}^T]^T$, $x_k$ is the current measured state and the matrices, $H$, $G$, $\bar{Q}$, $E$, $F$ and $b$ are suitably defined. The objective function, $V_{k+1}$, at time $k+1$ is analogously defined.

The stability test can then be formulated as the following optimization problem

$$\underset{U_k^*, U_{k+1}^*, x_k}{\text{minimize}} \quad V_k^* - V_{k+1}^* \tag{7.4a}$$

$$\text{subj. to} \quad x_{k+1} = Ax_k + Bu_k^* \tag{7.4b}$$

$$U_k^* = \arg\min \; V_k \tag{7.4c}$$

$$U_{k+1}^* = \arg\min \; V_{k+1} \tag{7.4d}$$

Hence we want to find the state $x_k$ of the system that results in the smallest possible difference in our candidate Lyapunov function when controlled with the MPC controller.

If this difference is nonnegative it means that $V_k$ is a valid Lyapunov function for the system and hence the closed loop is stable. On the other hand if this difference is negative this means that we have an increase in the Lyapunov function candidate for some point $x_k$ and hence it is not a valid Lyapunov function for the system.

Note however that when $V_k$ is a Lyapunov function then the difference (7.4a) is always zero if the origin is a stable equilibrium and negative if it is not, i.e., the minimum of the optimization problem will in fact never be positive. Therefore in order to show asymptotic stability or robustness the objective function (7.4a) needs to be modified to, e.g.,

$$V_k^* - V_{k+1}^* - \epsilon \left( x_k^T Q x_k + u_k^T R u_k \right)$$

At this point it should be pointed out that it is assumed, throughout the paper, that the MPC algorithms are recursively feasible. This must of course be tested before one can apply the proposed stability test. A method for performing the feasibility test is presented in Löfberg [2012].

Note also that in the case of the MPC problem having several non-unique solutions then the stability test derived here is a pessimistic bound on the stability, i.e., it selects the worst case combinations of optimal points, $U_k$, $U_{k+1}$ and $x_k$.

Since the MPC problem (7.3) is a convex QP we can replace it in (7.4c) and (7.4d) with the necessary and sufficient KKT conditions

$$HU_k + Gx_k + F^T \lambda_k = 0 \tag{7.5a}$$

$$Ex_k + FU_k - b \leq 0 \tag{7.5b}$$
$$\lambda_k \geq 0 \tag{7.5c}$$
$$\lambda_k^T(Ex_k + FU_k - b) = 0 \tag{7.5d}$$

and similar for time $k + 1$.

Note that equations (7.5d) are bilinear constraints but they can for each row be modeled using a *Big-M* reformulation as four linear constraints with a binary variable, $z_k^{(i)}$, and scalars, $m_1^{(i)}$, and, $m_2^{(i)}$, as

$$\lambda_k^{(i)}(e_i^T x_k + f_i^T U_k - b_i) = 0 \Leftrightarrow$$
$$0 \leq \lambda_k^{(i)} \leq m_1^{(i)} z_k^{(i)},$$
$$0 \leq b_i - e_i^T x_k - f_i^T U_k \leq m_2^{(i)}(1 - z_k^{(i)}) \tag{7.6}$$

We can see that the binary variable $z_k^{(i)}$ forces either the constraint, $b_i - e_i^T x_k - f_i^T U_k$, to be equal to zero (when $z_k^{(i)} = 1$) or the dual variable, $\lambda_k^{(i)}$, to be equal to zero (when $z_k^{(i)} = 0$), ensuring that the complementarity constraint holds. In other words, we can view this as, $z_k^{(i)}$ encodes whether the constraint is active or not, i.e., $z_k^{(i)} = 1 \Rightarrow e_i^T x_k + f_i^T U_k = b_i$. The scalars $m_1^(i)$ and $m_2^(i)$ needs to be selected large enough to not affect the value of $\lambda_k^{(i)}$ or $b_i - e_i^T x_k - f_i^T U_k$.

Using the KKT conditions (7.5) and the binary reformulation (7.6) we can write the problem (7.4) using the notation, $y = [U_k^T, x_k^T, \lambda_k^T, U_{k+1}^T, x_{k+1}^T, \lambda_{k+1}^T]^T$ and $\bar{z} = [z_k^T, z_{k+1}^T]^T$ as

$$\underset{y,\bar{z}}{\text{minimize}} \ \frac{1}{2} y^T \bar{H} y \tag{7.7a}$$
$$\text{subj. to} \ \bar{E} y = 0 \tag{7.7b}$$
$$\bar{A} y \leq \bar{b} + \bar{d}\bar{z} \tag{7.7c}$$

where

$$\bar{H} = \begin{bmatrix} H & G & 0 & 0 & 0 & 0 \\ G^T & \bar{Q} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -H & -G & 0 \\ 0 & 0 & 0 & -G^T & -\bar{Q} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\bar{E} = \begin{bmatrix} \bar{B} & A & 0 & 0 & -I & 0 \\ H & G & F^T & 0 & \dots & \\ & \dots & 0 & H & G & F^T \end{bmatrix}$$

$$\bar{A} = \begin{bmatrix} F & E & 0 & \dots & & \\ -F & -E & 0 & 0 & \dots & \\ \dots & 0 & -I & 0 & \dots & \\ \dots & 0 & I & 0 & \dots & \\ & \dots & 0 & F & E & 0 \\ & \dots & 0 & -F & -E & 0 \\ & \dots & 0 & 0 & 0 & -I \\ & \dots & 0 & 0 & 0 & I \end{bmatrix}$$

$$\bar{b} = \begin{bmatrix} b \\ M_2\mathbb{1} - b \\ 0 \\ 0 \\ b \\ M_2\mathbb{1} - b \\ 0 \\ 0 \end{bmatrix}, \quad \bar{d} = \begin{bmatrix} 0 & 0 \\ -M_2 & 0 \\ 0 & 0 \\ M_1 & 0 \\ 0 & 0 \\ 0 & -M_2 \\ 0 & 0 \\ 0 & M_1 \end{bmatrix}$$

and where we have defined $\bar{B}$ such that $\bar{B}U_k = Bu_k$. $M_1$ and $M_2$ are diagonal matrices with the Big-M scalars $m_1^{(i)}$ and $m_2^{(i)}$ as their diagonal elements.

Even though we have replaced the bilinear constraints with linear constraints with integer variables the problem is still an indefinite QP. This is very difficult for any integer programming method to solve and we have to reformulate the problem into an LP or convex QP program.

To make the final reformulation into a MILP we observe that

$$\min_{y,\bar{z}} \frac{1}{2} y^T \bar{H} y = \min_{\bar{z}} \left( \min_{y} \frac{1}{2} y^T \bar{H} y \right)$$

where $\bar{z}$ is a parameter in the optimization problem over $y$. We can then replace the optimization problem over $y$ with its KKT conditions in the minimization over $\bar{z}$ and thus we arrive at

$$\bar{H}y + \bar{A}^T\eta + \bar{E}^T\mu = 0 \tag{7.8a}$$

$$\bar{A}y - \bar{b} - \bar{d}\bar{z} \leq 0 \tag{7.8b}$$

$$\bar{E}y = 0 \tag{7.8c}$$

$$\eta^T(\bar{A}y - \bar{b} - \bar{d}\bar{z}) = 0 \tag{7.8d}$$

$$\eta \geq 0 \tag{7.8e}$$

The KKT conditions in (7.8) are necessary but not sufficient for optimality since $\bar{H}$ is indefinite. Thus it is necessary to test all points that fulfill the conditions in order to find the optimal point and this is what is done in the outer level optimization where we minimize over $\bar{z}$.

If we now multiply (7.8a) with $\frac{1}{2}y^T$ from the left we have

$$\frac{1}{2}y^T\bar{H}y = -\frac{1}{2}(y^T\bar{A}^T\eta + \underbrace{y^T\bar{E}^T}_{=0}\mu) = -\frac{1}{2}\left(\bar{b} + \bar{d}\bar{z}\right)^T\eta$$

We see that in the optimum the objective can be equivalently written as a linear term plus a bilinear term between a real variable and a binary variable. We can see that the elements of the bilinear term are either 0, if $\bar{z}^{(i)} = 0$, or equal to $\bar{d}_i^T\eta$ when $\bar{z}^{(i)} = 1$. Hence we can introduce a new variable, $w$, as

$$\bar{z}^T\bar{d}^T\eta = \mathbb{1}^T w$$

where the elements $w_i$ can be modeled, yet again using the Big-M formulation, with four linear constraints

$$-M_3(\mathbb{1} - \bar{z}) \leq w - \bar{d}^T\eta \leq M_3(\mathbb{1} - \bar{z}), \quad -M_3\bar{z} \leq w \leq M_3\bar{z}$$

We can now combine all the pieces together and formulate the stability problem (7.4) as the following MILP

$$\underset{\eta,\mu,y,w,\bar{z},q}{\text{minimize}} \quad -\frac{1}{2}\bar{b}^T\eta - \frac{1}{2}\mathbb{1}^T w \tag{7.9a}$$

subj. to

$$\bar{H}y + \bar{A}^T\eta + \bar{E}^T\mu = 0 \tag{7.9b}$$

$$\bar{E}y = 0 \tag{7.9c}$$

$$-M_3(\mathbb{1} - \bar{z}) \leq w - \bar{d}^T\eta \leq M_3(\mathbb{1} - \bar{z}) \tag{7.9d}$$

$$-M_3\bar{z} \leq w \leq M_3\bar{z} \tag{7.9e}$$

$$-M_4(\mathbb{1} - q) \leq (\bar{A}y - \bar{b} - \bar{d}\bar{z}) \leq 0 \tag{7.9f}$$

$$0 \leq \eta \leq M_5 q \tag{7.9g}$$

The task of selecting the Big-M constants, $M_i$, is by no means a simple task and is partly still an open problem. If the constants are selected too small the problem (7.9) will have a smaller feasible set meaning that we could obtain a positive optimal value of (7.9) even though the original problem (7.4) has a negative optimal value. This means that we obtain a false certificate of stability if we select the Big-M constants too small. On the other hand if we select the constants too big this will lead to numerical problems, i.e., they should be selected as small as possible without affecting the optimal solution.

Note that we are not really interested in finding the optimum of problem (7.9), but rather in finding if there exists any point, $x_k$, where the objective is less than zero. Hence we can reformulate the problem into a feasibility problem which often is much faster to solve

$$\text{find} \quad x_k$$

subj. to

$$-\frac{1}{2}\bar{b}^T\eta - \frac{1}{2}\mathbb{1}^T w < 0 \tag{7.10a}$$

$$\bar{H}y + \bar{A}^T\eta + \bar{E}^T\mu = 0 \tag{7.10b}$$

$$\bar{E}y = 0 \tag{7.10c}$$

$$-M_3(\mathbb{1} - \bar{z}) \leq w - \bar{d}^T\eta \leq M_3(\mathbb{1} - \bar{z}) \tag{7.10d}$$

$$-M_3\bar{z} \leq w \leq M_3\bar{z} \tag{7.10e}$$

$$-M_4(\mathbb{1} - q) \leq (\bar{A}y - \bar{b} - \bar{d}\bar{z}) \leq 0 \tag{7.10f}$$

$$0 \leq \eta \leq M_5 q \tag{7.10g}$$

Due to the special structure of the original problem there is a large amount of structure in the problem (7.10) that should be exploited to enhance the performance of the MILP representation.

### 7.2.1 Exploiting structure in the MILP

First, let us observe that in MPC problems there are often upper and lower bounds on the variables, e.g., $u_{min} \leq u_{k+i} \leq u_{max}$. These constraints can not be fulfilled with equality at the same time and hence the two corresponding binary variables, $z_k^{(i)}$ and $z_k^{(j)}$, in (7.6) can not be equal to one at the same time. So we can introduce the constraint

$$z_k^{(i)} + z_k^{(j)} \leq 1 \tag{7.11}$$

for the appropriate indices $i$ and $j$.

Consider now a single constraint $e_i^T x_k + f_i^T U_k \leq b_i$ in the original MPC problem. This single constraint generates through (7.6) the four constraints

$$e_i^T x_k + f_i^T U_k - b_i \leq 0 \tag{7.12a}$$

$$b_i - e_i^T x_k - f_i^T U_k \leq m_2^{(i)}(1 - z_k^{(i)}) \tag{7.12b}$$

$$-\lambda_k^{(i)} \leq 0 \tag{7.12c}$$

$$\lambda_k^{(i)} \leq m_1^{(i)} z_k^{(i)} \tag{7.12d}$$

in (7.7c).

By formulating the KKT conditions for (7.7), for each of these four constraints we have yet another binary variable, $q^{(\cdot)}$, in (7.9f) and (7.9g). Since the binary variables force a constraint to be active we can see that if $z_k^{(i)} = 0$ then (7.12a) and (7.12b) can not both be active at the same time. This means that the corresponding elements $q^{(i)}$ and $q^{(n_\lambda + i)}$ can not both be equal to one, i.e., we can constrain them as

$$q^{(i)} + q^{(n_\lambda + i)} \leq 1 + z_k^{(i)} \tag{7.13}$$

Furthermore if $z_k^{(i)} = 1$ we have from (7.12a) and (7.12b) that $e_i^T x_k + f_i^T U_k = b_i$

which gives in (7.9f) and (7.9g) that

$$m_4^{(i)}(1-q^{(i)}) \leq e_i^T x_k + f_i^T U_k - b_i = 0$$
$$m_4^{(n_\lambda+i)}(1-q^{(n_\lambda+i)}) \leq b_i - e_i^T x_k - f_i^T U_k = 0$$

and hence we can in this case without any loss of generality constrain both $q^{(i)}$ and $q^{(n_\lambda+i)}$ to be equal to one. This is done by adding the constraint

$$q^{(i)} + q^{(n_\lambda+i)} \geq 2z_k^{(i)} \tag{7.14}$$

The corresponding argumentation can be used for (7.12c) and (7.12d) to introduce the two additional constraints

$$q^{(3n_\lambda+i)} + q^{(4n_\lambda+i)} \leq 1 + (1-z_k^{(i)}) \tag{7.15}$$
$$q^{(3n_\lambda+i)} + q^{(4n_\lambda+i)} \geq 2(1-z_k^{(i)}) \tag{7.16}$$

The constraints (7.13) - (7.16) concern relations in one MPC constraint, $i$, but as stated in (7.11) there exist relations also between different MPC constraints. This gives additional relationships between the different binary variables which we encode as the following constraints

$$q^{(n_\lambda+i)} \leq 1 - z_k^{(j)} \tag{7.17}$$
$$q^{(2n_\lambda+i)} \leq 1 - z_k^{(j)} \tag{7.18}$$
$$q^{(3n_\lambda+i)} + q^{(4n_\lambda+i)} \geq 2z_k^{(j)} \tag{7.19}$$
$$q^{(n_\lambda+j)} \leq 1 - z_k^{(i)} \tag{7.20}$$
$$q^{(2n_\lambda+j)} \leq 1 - z_k^{(i)} \tag{7.21}$$
$$q^{(3n_\lambda+j)} + q^{(4n_\lambda+j)} \geq 2z_k^{(i)} \tag{7.22}$$

The stability analysis optimization problem to be solved thus consists of the problem (7.10) with the additional constraints (7.11) and (7.13) - (7.22). Note that the added binary constraints are redundant in the original problem and do not affect the optimal solution. They are only added to cut off binary combinations in order to possibly increase the performance of the solver.

Note that nowhere in the derivation of the algorithm do we use the fact that it is the MPC controller objective function that we have as candidate Lyapunov function, $V_k$. Hence we can generalize the algorithm to the use of any other positive definite $V_k$ by appropriately modifying the matrix $\bar{H}$.

In the next sections we will look at three examples and try to illustrate some properties and performance of the proposed algorithm.

All examples have been implemented in MATLAB using YALMIP, Löfberg [2004]. The MILP problems have been solved using the solver Gurobi 5.6.2, Inc. [2014] and the LMI problems have been solved using MOSEK 7.1, ApS [2015].

### 7.2.2 A sufficient but not necessary condition

The first example is taken from Löfberg [2003] where we consider the following unstable, non minimum phase system

$$x_{k+1} = \begin{bmatrix} 1.216 & -0.055 \\ 0.221 & 0.9947 \end{bmatrix} x_k + \begin{bmatrix} 0.02763 \\ 0.002673 \end{bmatrix} u_k$$

which we control using an unconstrained finite time MPC controller with the objective function

$$V_k = \underset{u_{k+i}}{\text{minimize}} \sum_{i=0}^{N-1} x_{k+i}^T Q x_{k+i} + R u_{k+i}^2$$

and $Q = 10I$ and $R = 1$. Since the controller is unconstrained we can easily calculate for which prediction horizon length, $N$, the closed loop system is stable by looking at the eigenvalues of the closed loop system matrix, $A - BL$. In the range $N = 1, \ldots, 50$ we calculate the eigenvalues of the closed loop system and can conclude that it is stable for $N \geq 8$. When we use the algorithm derived in Section 7.2 to test for stability we obtain the result that the system is stable for $N \geq 21$. Running the LMI algorithm from Primbs [2001] we obtain the same results.

Here we clearly see that the test is only a sufficient, but not necessary, condition for stability since apparently there exists a set of $8 \leq N \leq 20$ where the system is stable but the objective function does not constitute a valid Lyapunov function for the closed loop system.

### 7.2.3 Computational complexity of the stability test

Let us now consider a more realistic example taken from the aircraft industry.

We consider the stabilization of the short period dynamics of the ADMIRE aircraft [Forssell and Nilsson, 2005] and we have linearized the system at Mach 0.6 and altitude 4km. This results in the system

$$x_{k+1} = \begin{bmatrix} 0.9798 & 0.0158 \\ 0.1449 & 0.9787 \end{bmatrix} x_k + \begin{bmatrix} 0.0106 \\ 0.4878 \end{bmatrix} u_k \tag{7.23}$$

This system is controlled with an input constrained MPC controller with objective function

$$\sum_{i=0}^{N-1} x_{k+i}^T Q x_{k+i} + R u_{k+i}^2 + x_{k+N}^T P x_{k+N}$$

with $Q = I$, $R = 10$ and $P = \begin{bmatrix} 10 & 1 \\ 1 & 2 \end{bmatrix}$ and the constraints $-5 \leq u_{k+i} \leq 5$.

To investigate the computational complexity of the algorithm we have compared our developed MILP algorithm to the LMI algorithm developed in Primbs

[2001]. Note that the system is open loop unstable and hence is the method developed by Lennox et al. [2008] not applicable.

Both algorithms verify that the closed loop system is stable for all tested prediction horizon lengths but the computational time differs significantly between the two algorithms.

The following table shows the solvers computational time in seconds for the two methods as a function of prediction horizon.

| N | 2 | 4 | 6 | 8 | 10 |
|---|---|---|---|---|---|
| LMI | 0.03 | 0.52 | 5.36 | 22.42 | 80.06 |
| MILP | 0.06 | 0.17 | 0.28 | 0.42 | 0.61 |

As can be seen from the table the computation time grows very rapidly with increasing prediction horizon for the LMI algorithm while it remains relatively small for the MILP approach.

### 7.2.4 Move blocking and no stabilizing constraints

Let us again consider the ADMIRE aircraft model (7.23).

This time we design a stabilizing MPC controller of the form (7.1) where the objective is a quadratic function

$$\sum_{i=0}^{N-1} x_{k+i}^T Q x_{k+i} + R u_{k+i}^2 + x_{k+N}^T P x_{k+N}$$

with $Q = \begin{bmatrix} 2 & 0 \\ 0 & 0.1 \end{bmatrix}$, $R = 10$ and $P$ is the associated LQ cost. The constraints are upper and lower bounds on the states and control

$$\begin{bmatrix} -10 \\ -50 \end{bmatrix} \leq x_k \leq \begin{bmatrix} 10 \\ 50 \end{bmatrix}, \quad -20 \leq u_k \leq 20$$

Since the system is unstable and constrained it is "generally necessary" to have a final state constraint, Mayne et al. [2000]. We select the final state constraint set to the invariant set of the associated LQ controller, see e.g., Rawlings and Mayne [2009] for more details.

Let us first consider the statement that it is necessary to have a terminal state constraint. The MPC controller is by construction stabilizing and when we test the stability of the closed loop system with the algorithm derived in the previous section we see that it indeed is stable for all $N$ in the tested range ($N = 2, \ldots, 10$). The question is, is it still stable if we would remove the terminal state constraint set?

Running the algorithm again, now without the terminal constraint set in the controller, the test indicates that the controller still is stable for all tested prediction horizons. To verify this result we select 1000 random initial

conditions and simulate the closed loop system with prediction horizon, $N = 5$. As shown in Figure 7.1 all initial conditions that are within the initially feasible set are stable.

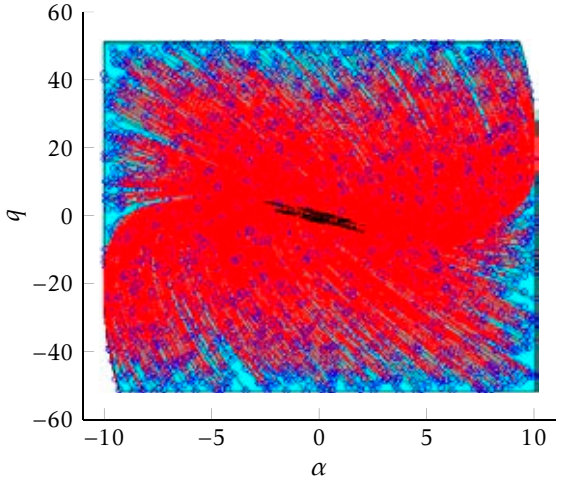


**Figure 7.1.** A simulation of the closed loop system from 1000 different initial conditions for the MPC controller without terminal constraint. The blue circles are the initial conditions, red lines are the trajectories and the black squares are the final states. The cyan polytope is the set of initially feasible states.

Now, simplify the MPC controller even more by introducing move blocking and analyze the stability of the resulting closed loop system. We select the prediction horizon $N = 4$ and introduce the move blocking structure

$$U_k = T\hat{U}_k, \quad T = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

where $\hat{U}_k$ is the new reduced set of input signals. This blocking structure gives a controller where the sampling time of the controller output is half the internal sampling time used in the predictions.

As pointed out in Cagienard et al. [2007] the standard feasibility and stability arguments can not be used to prove stability for this move blocking strategy. Instead we aim to prove stability by using our MILP test.

When applying the MILP algorithm it returns a sufficient condition certificate that the move blocking MPC design is stabilizing. To verify this result we again simulate a set of 1000 random initial conditions, now with horizon $N = 4$, throughout the state constraint set, see Figure 7.2. It is clear from this figure that for all initially feasible points the move blocking MPC controller remains feasible and stabilizes the system, as proven by the MILP test.

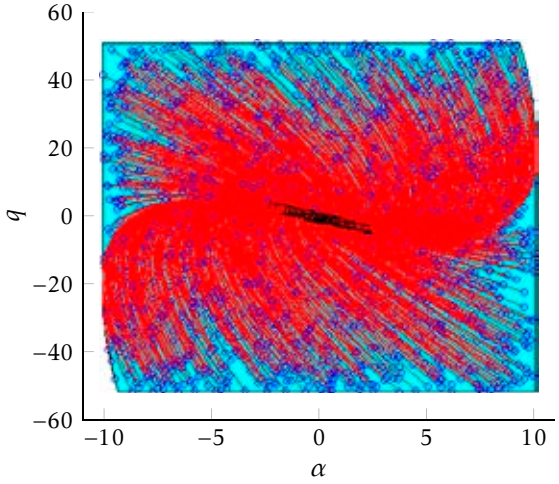


**Figure 7.2.** A simulation of the closed loop system from 1000 different initial conditions for the move blocking MPC controller.

## 7.3 Testing for robust stability

In the previous sections we have considered only nominal stability of the closed loop system. However in any real cases there will be disturbances and model errors acting on the system. In this section we will make a rather straightforward extension of the nominal stability test to a robust stability test for systems subject to additive disturbances.

### 7.3.1 Minimal robust invariant set

Consider the case when the linear system is subject to an additive disturbance, $w_k$, i.e.,

$$x_{k+1} = Ax_k + Bu_k + w_k, \quad w_k \in \mathcal{W}$$

The disturbance $w_k$ is unknown but bounded to some known set $\mathcal{W}$. It is obvious that the state in this case will not converge to the origin since close enough to the origin a disturbance can push the state away from the origin. But we know that given a stabilizing control law there exist a positively invariant set around the origin such that if the state enters this set it will remain within this set for all bounded disturbances [Kolmanovsky and Gilbert, 1998]. This set is known as the *minimal robust positively invariant set*, $\mathcal{R}_\infty^{min}$. In order to formally define this set we first need to define a robust positively invariant set.

**Definition 7.1.** (RPI set) [Rakovic et al., 2005] The set $\mathcal{R}_\infty \subseteq \mathbb{R}^n$ is a *robust positively invariant set* (RPI) of the stable system $x_{k+1} = Ax_k + w_k$ if $Ax_k + w_k \in \mathcal{R}_\infty$ for all $x_k \in \mathcal{R}_\infty$ and all $w_k \in \mathcal{W}$.

We can now define the minimal robust positively invariant set as follows.

**Definition 7.2.** (minimal RPI set) [Rakovic et al., 2005] The *minimal* RPI set, $\mathcal{R}_\infty^{min}$, of the system $x_{k+1} = Ax_k + w_k$ is the robust positively invariant set in $\mathbb{R}^n$ that is contained in every closed robust positively invariant set of the system.

For general linear stable systems there is no known way to calculate an exact representation of the minimal RPI set but in Rakovic et al. [2005] the authors develop an algorithm for calculating an invariant outer approximation of the minimal RPI set. Furthermore the algorithm gives a measure of the approximation error.

**Example 7.3**

Let us illustrate the RPI set calculations with a simple example. Consider the same aircraft model as in the example from section 7.2.4 and let the system be subject to a bounded disturbance, $-0.43 \leq w_k \leq 0.43$, on both states. If we calculate the approximation of the minimal RPI set for the system using the unconstrained MPC solution (i.e., the LQ feedback) with the penalty matrices.

$$Q = \begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1, \quad P = \begin{bmatrix} 658.765 & 9.912 \\ 9.912 & 2.6816 \end{bmatrix}$$

In Figure 7.3 we plot the outer RPI approximation (cyan) together with an inner approximation. Note that the inner approximation is not an invariant approximation.

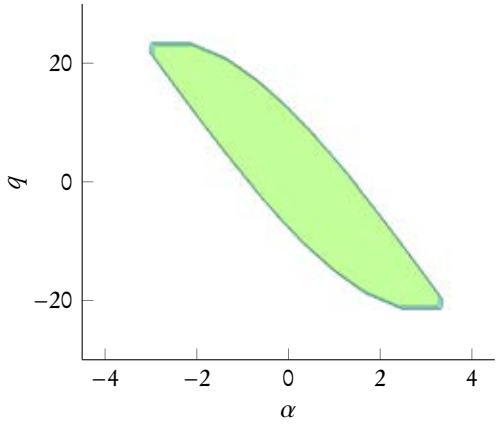


**Figure 7.3.** An example of the inner (yellow) and outer (cyan) minimal RPI set approximation calculated with an approximation error of $\epsilon = 0.5$.

### 7.3.2 The robust stability condition

Since the Lyapunov function candidate from the previous section will not have a negative time difference within the minimal RPI set we need to formulate the robust stability test as a convergence test to the minimal RPI set instead. Therefore let us define $d_k^*$ to be the distance from a given $x_k$ to the minimal RPI set, i.e., $d_k^* = \min_{a_k} \|x_k - a_k\|_2$ subj. to $a_k \in \mathcal{R}_\infty^{min}$.

In order for us to prove that the system is stable and that the states asymptotically converges to the minimal RPI we want the distance to the minimal RPI set to monotonically decrease over time, so we require that

$$d_k^* - d_{k+1}^* \geq \epsilon d_k^* \quad \forall\, x_k \in \mathcal{X} \;\; w_k \in \mathcal{W}$$

Since $d_{k+i}^* \geq 0 \; \forall x_k$ we can just as well show that

$$(1-\epsilon) d_k^{*2} - d_{k+1}^{*2} \geq 0$$

This means that

$$\min_{x_k \in \mathcal{X}} (1-\epsilon) d_k^{*2} - d_{k+1}^{*2} \geq 0$$

and we can in the same way as in previous section formulate the following indefinite bilevel optimization problem.

$$\underset{x_k, x_{k+1}, a_k, a_{k+1}, U_k, w_k}{\text{minimize}} \quad \frac{1}{2}\left((1-\epsilon)\|x_k - a_k\|_2^2 - \left\|x_{k+1} - a_{k+1}^*\right\|_2^2\right) \tag{7.24a}$$

$$\text{subj. to} \tag{7.24b}$$

$$x_{k+1} = Ax_k + Bu_k^* + Dw_k \tag{7.24c}$$

$$U_k^* = \arg\min_{U_k} \; U_k^T H U_k + U_k^T G x + k \tag{7.24d}$$

$$\text{subj. to} \quad Ex_k + FU_k \leq b$$

$$a_{k+1}^* = \arg\min_{a_{k+1}} \; \frac{1}{2}\|x_{k+1} - a_{k+1}\|_2^2 \tag{7.24e}$$

$$\text{subj. to} \quad a_{k+1} \in \tilde{\mathcal{R}}_\infty^{min}$$

$$w_k \in \mathcal{W} \tag{7.24f}$$

$$a_k \in \tilde{\mathcal{R}}_\infty^{min} \tag{7.24g}$$

Note that in this case, our Lyapunov function candidate is simply $d_k^{*2}$ but nothing prevents the user from proposing other, perhaps more suitable, Lyapunov functions.

At this point we need to make some remarks about the above formulation. First we can note that comparing (7.4) with the formulation (7.24) only the MPC formulation at time $k$ remain, this is because now the MPC problem at time $k+1$ does not affect the solution ($U_{k+1}$ does not appear in the objective function). Further, the closest point in $\mathcal{R}_\infty^{min}$ at time $k$, $a_k$, does not need to be explicitly stated as the optimal minimum distance point with a separate

subproblem, since the objective function guarantees that the minimum distance point at time $k$ is selected automatically (which is not the case for the point, $a_{k+1}$). The disturbance, $w_k$, will also be optimized such that $x_{k+1}$ is as far away as possible from $\mathcal{R}_\infty^{min}$, i.e., the worst case scenario, since this is what minimizes the objective function. Finally we should also note that, as we discussed in the previous section, we can not calculate the exact representation of $\mathcal{R}_\infty^{min}$ and hence we use the outer approximation, which we will denote $\tilde{\mathcal{R}}_\infty^{min}$, from Rakovic et al. [2005].

### 7.3.3 Reformulation into a MILP

Define $y = [U_k, x_k, a_k, \lambda, w_k, x_{k+1}, a_{k+1}, \gamma]$ where $\lambda$ is the dual variable related to (7.24d) and $\gamma$ the dual variable related to (7.24e). We can then, by replacing (7.24d) and (7.24e) with their KKT conditions, formulate the robust stability test in the same form as the nominal stability test (7.7).

$$\begin{aligned} \underset{y,\bar{z}}{\text{minimize}} \quad & \frac{1}{2} y^T \bar{H} y \\ \text{subj. to} \quad & \bar{E} y = 0 \\ & \bar{A} y \leq \bar{b} + \bar{d}\bar{z} \end{aligned}$$

where

$$\bar{H} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_\epsilon & -I_\epsilon & 0 & 0 & 0 & 0 & 0 \\ 0 & -I_\epsilon & I_\epsilon & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -I & I & 0 \\ 0 & 0 & 0 & 0 & 0 & I & -I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and we have used $I_\epsilon$ to denote $(1-\epsilon)I$. Furthermore

$$\bar{E} = \begin{bmatrix} \bar{B} & A & 0 & 0 & D & -I & 0 & 0 \\ H & G & 0 & F^T & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -I & I & F_{\mathcal{R}}^T \end{bmatrix}$$

$$\bar{A} = \begin{bmatrix} F & E & 0 & 0 & 0 & 0 & 0 & 0 \\ -F & -E & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & F_{\mathcal{R}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & F_{\mathcal{W}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & F_{\mathcal{R}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -F_{\mathcal{R}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -I \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I \end{bmatrix}$$

$$\bar{b} = \begin{bmatrix} b \\ M_1\mathbb{1} - b \\ b_{\mathcal{R}} \\ 0 \\ 0 \\ b_{\mathcal{W}} \\ b_{\mathcal{R}} \\ M_3\mathbb{1} - b_{\mathcal{R}} \\ 0 \\ 0 \end{bmatrix}, \quad \bar{d} = \begin{bmatrix} 0 & 0 \\ -M_1 & 0 \\ 0 & 0 \\ 0 & 0 \\ M_2 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & -M_3 \\ 0 & 0 \\ 0 & M_4 \end{bmatrix}$$

This is equivalent to the formulation (7.7) and from this point on we can apply the derivation from previous section and end up with in the feasibility test (7.10) that now test for robust stability. The details are omitted for brevity.

### 7.3.4 Robust stability of an agile fighter aircraft

Let us now apply the robust stability test on the aircraft model in Section 7.2.4. The dynamics are

$$x_{k+1} = \begin{bmatrix} 0.9798 & 0.0158 \\ 0.1449 & 0.9787 \end{bmatrix} x_k + \begin{bmatrix} 0.0106 \\ 0.4878 \end{bmatrix} u_k + w_k \tag{7.25}$$

This system is controlled with the MPC controller

$$\begin{aligned} \underset{x_{k+i}, u_{k+i}}{\text{minimize}} \quad & \sum_{i=1}^{N-1} x_{k+i}^T Q x_{k+i} + u_{k+i}^T R u_{k+i} + x_{k+N}^T P x_{k+N} \\ \text{subj. to} \quad & \\ & x_{k+1+i} = A x_{k+i} + B u_{k+i} \\ & \begin{bmatrix} -10 \\ -50 \end{bmatrix} \leq x_{k+i} \leq \begin{bmatrix} 10 \\ 50 \end{bmatrix} \\ & -20 \leq u_{k+i} \leq 20 \end{aligned}$$

where $P$, $Q$ and $R$ are defined as in Example 7.3 and $N = 5$.

We utilize the test for recursive feasibility in Löfberg [2012] to calculate the maximal disturbance for which the system still is recursively feasible. The test results in that the maximum disturbance is given by

$$\|w_k\|_\infty \leq 0.4308$$

We now formulate the robust stability test as defined above for convergence of the state into the outer approximation of the minimal RPI set. This is the set from Example 7.3 but now we use the approximation error bound $\epsilon = 5 \cdot 10^{-5}$.

The test show that the system is robustly stable in the sense that it converges into the outer approximation of the minimal RPI set when controlled with the

MPC controller. If we instead would have used the standard stability test from Section 7.2 it would have failed to prove stability and pointed out a state, $x_k$, within the minimal RPI set as a point where the difference in the Lyapunov function increases.

To verify the result we generate a set of 500 random feasible initial conditions on the state and simulate the closed loop system for 20 time steps. The results from the simulations are shown in Figure 7.4 and it is possible to verify that all the trajectories converge into the minimal RPI set, as predicted by our test.

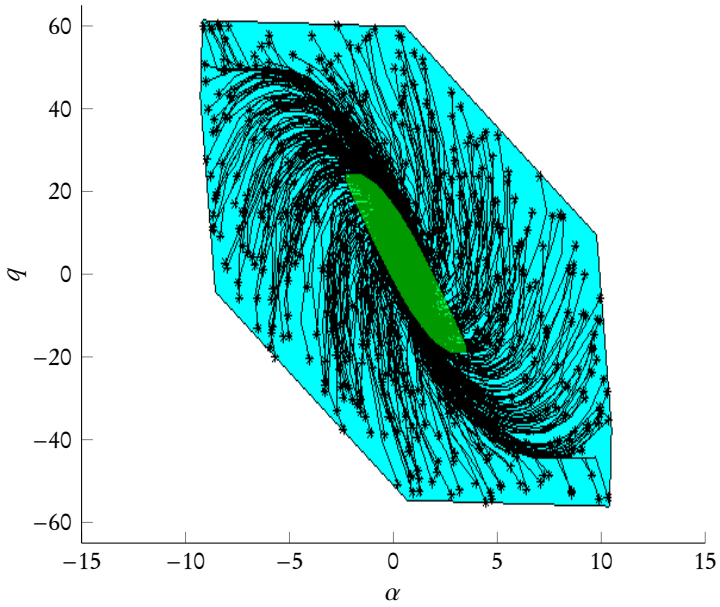


**Figure 7.4.** The state trajectory of the closed loop system from 500 randomly chosen feasible initial conditions. The cyan polytope is the initially feasible set, the yellow polytope is the minimal RPI set.

Part II

# Application

# 8

# Industrial implementation of an MPC controller for a fighter aircraft

## 8.1 Introduction

In previous chapters we have mainly focused on theoretical problems that have a great relevance for the aeronautical industry. However it is equally important to review the practical implementation aspects and tuning strategies of real world flight control design problems in order to evaluate the applicability of MPC within the aeronautical industry and to identify new research directions.

We will therefore focus on the practical aspects of model predictive controllers for flight control design in this chapter. We will describe the design, tuning and implementation of a model predictive controller as the main flight control system for an agile fighter aircraft. The controller has been implemented in Saab's main simulation environment, ARES, and tested both using desktop simulations with virtual pilots and simulator testing using real human pilots. The main objective of this chapter is to give a proof of concept of MPC as a design technique for advanced flight control systems for agile fighter aircraft. The secondary objective is to describe a design and tuning methodology of MPC controllers suitable for flight control design.

As we discussed in the introduction there exist much research relating model predictive control to aeronautical applications and there exists a wide variety of applications and theory. However, research that is within the scope of this thesis, maneuver limiting of fighter aircraft, is more scarce. Great examples of the topic of maneuver limiting can be found in papers such as Falkena et al. [2011], Gros et al. [2012] and Hartley [2015]. In Falkena et al. [2011] they compare MPC with other control strategies to achieve envelope protection for small personal transport air vehicles. The authors compare MPC with a PID based approach, a command limiting approach and a virtual control limiting

approach. The MPC controller is applied in combination with a nonlinear dynamic inversion controller. Hartley [2015] develop a stall protection system for undetected ice build up based on an MPC controller and an adaptive prediction model that is measured online while Gros et al. [2012] applies nonlinear MPC to control a generic aircraft model during extreme maneuvering without breaking any flight envelop limitations.

In this chapter we implement a gain scheduled linear MPC controller structure with an internal sample interval that is different from the closed loop implementation sampling. We propose a tuning methodology based on classical LQ tuning methodology that has been used for decades at Saab for flight control design. Finally we demonstrate the performance and validity of the proposed design with the implementation in the main flight mechanical simulator at Saab Aeronautics, which is also used for the JAS 39 Gripen fighter aircraft development.

## 8.2 The MPC controller structure

The MPC controller we have designed is an inner loop controller for the unstable pitch dynamics of the ARES aircraft model (see Section 3.4). The controller for the lateral dynamics is the baseline gain scheduled LQ controller of the ARES model.

We have selected to use an MPC controller with a linear prediction model scheduled over Mach number and altitude. As pointed out in Keviczky and Balas [2006a] a gain scheduled controller often performs sufficiently well compared to a full nonlinear MPC for these type of applications. As we will see later in this chapter the nonlinearities in the dynamics over the range of angle of attack can be handled with great performance using linear models and the integral action scheme explained in Section 4.2.3. The nonlinear pitch dynamics has been linearized around level trimmed flight at different Mach and altitude points in the subsonic region.

The MPC controller structure that has been chosen is the basic linear MPC structure (4.4) but without any terminal state constraints and with slack added.

$$\underset{x_{k+i},u_{k+i}}{\text{minimize}} \quad \sum_{i=0}^{N-1}(x_{k+i}-x_r)^T Q(x_{k+i}-x_r) + (u_{k+i}-u_r)^T R(u_{k+i}-u_r) + \gamma\varepsilon_i^T\varepsilon_i + (x_{k+N}-x_r)^T P(x_{k+N}-x_r) \tag{8.1a}$$

subj. to

$$x_{k+i+1} = Ax_{k+i} + Bu_{k+i} + D\bar{x} \tag{8.1b}$$

$$x_{min} - \varepsilon_i \le x_{k+i} \le x_{max} + \varepsilon_i \quad \forall\, i = 0, \ldots, N \tag{8.1c}$$

$$u_{min} \le u_{k+i} \le u_{max} \quad \forall\, i = 0, \ldots, N-1 \tag{8.1d}$$

$$x_k = x(t_k) \tag{8.1e}$$

The state of the system is chosen to be $x = [\delta_c \ \delta_e \ \alpha \ q]^T$, where $\delta_c$ is the canard control surface angle, $\delta_e$ is the elevon control surface angle[1], $\alpha$ is the angle of attack and finally $q$ is the pitch rate. In other words the model used here is the short period dynamics (3.8) augmented with the control surface actuator dynamics. One reason to augment the short period dynamics with the actuator dynamics is that it reduces the phase lag in the closed loop system compared to just ignoring the fast actuator dynamics and hence increases the performance. The control inputs $u$ are the commanded canard control surface angle and the commanded elevon control surface angle.

Since the variables in a linearized model is deviations from the linearization point we have rewritten the prediction model (8.1b) to be expressed in the actual values, which then includes the trim state, $\bar{x}$. The matrix $D$ in (8.1b) can easily be derived from linear model expression as

$$\begin{aligned}(x_{k+1} - \bar{x}) &= A(x_k - \bar{x}) + B(u_k - \bar{u}) \\ x_{k+1} &= Ax_k + Bu_k + (I - A)\bar{x} + B\bar{u} \\ &= Ax_k + Bu_k + (I - A)\bar{x} + [B \ \ 0]\bar{x} \\ &= Ax_k + Bu_k + (\underbrace{(I - A) + [B \ \ 0]}_{D})\bar{x}\end{aligned}$$

In the third equality we have used the assumption that the control surface command in trim, $\bar{u}$, is equal to the control surface angles, i.e., the states $\delta_c$ and $\delta_e$.

The main frequency of the flight control system is 120 Hz and hence the MPC problem (8.1) solved in a receding horizon fashion at 120 Hz. However since we want the prediction horizon to be long enough to cover at least the most part of the step response of the closed loop system, which is approximately 2 seconds, this would require a long prediction horizon and hence a very large QP to solve. Therefore we instead discretize the prediction model and the MPC controller in 10 Hz yielding a prediction horizon of $N = 20$.

Note that this approach, with defining the MPC controller in one frequency and running it in another, higher, frequency, can be viewed as a type of move blocking. The MPC controller assumes the control signal is updated only every 0.1 seconds which can be viewed as the control is kept constant for 12 samples (in the 120 Hz implementation) in a row while in reality it updates every 1/120 th second. One important implication of this is that as for any move blocking approach the standard recursive feasibility and stability proofs from Section 4.2.1 do not apply.

There are some important subtle implications when we have one internal frequency in the MPC controller and another in which the MPC controller is run and we will discuss these further in the next sections on tuning of the MPC controller.

[1]see Section 3.4 for explanation of the canard and elevon control surfaces.

In the objective (8.1a) we have penalized the deviations from the target steady state, $x_r$, and control, $u_r$. These are calculated as in Section 4.2.2 as the solution to

$$\begin{bmatrix} A - I & B \\ C & 0 \end{bmatrix} \begin{bmatrix} x_r \\ u_r \end{bmatrix} = \begin{bmatrix} -D\bar{x} - E\hat{d} \\ r \end{bmatrix} \tag{8.2}$$

This equation does not have a unique solution unless there are as many rows in $C$ as there are columns in $B$, i.e., the matrix on the left hand side is invertible. Using a (weighted) pseudo inverse is equivalent to use the (weighted) minimum norm solution on the control signal $u_r$. We tried this approach but could not find a suitable tuning such that the controller has good performance both at trimmed level flight as well as for various pilot commands. Instead we chose to use two different versions of (8.2), one when the pilot command is equal to zero and one when the pilot command is not equal to zero. The pilot commands an angle of attack deviation, $\Delta\alpha_{cmd}$, from the angle of attack for trimmed flight, $\bar{\alpha}$. When the pilot's command is zero, $\Delta\alpha_{cmd} = 0$, the reference is set to

$$r = \begin{bmatrix} \bar{\alpha} \\ 0 \end{bmatrix}$$

and the matrix

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

but when $\Delta\alpha_{cmd} \neq 0$

$$r = \bar{\alpha} + \Delta\alpha_{cmd}, \quad C = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}$$

The variable, $\hat{d} \in \mathbb{R}^2$ in (8.2), which is the integral term as described in Section 4.2.3, is estimated using a Kalman filter and the model

$$\begin{bmatrix} x_{k+1} \\ d_{k+1} \end{bmatrix} = \begin{bmatrix} A_{KF} & E_{KF} \\ 0 & I \end{bmatrix} \begin{bmatrix} x_k \\ d_k \end{bmatrix} + \begin{bmatrix} B_{KF} \\ 0 \end{bmatrix} u_k + Gw_k \tag{8.3}$$

$$y_k = x_k + v_k \tag{8.4}$$

Even though the matrices $A_{KF}$, $B_{KF}$ and $E_{KF}$ represent the same continuous time system as the matrices $A$, $B$ and $E$ in the MPC controller we denote them differently since the Kalman filter is discretized in 120 Hz and the MPC controller in 10 Hz. The $E_c$-matrix in the continuous time system, from which the $E$ and $E_{KF}$ matrices are calculated, has the form

$$E_c = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The matrix $G$ in the Kalman filter has been chosen to

$$G = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This means that we assume to have a perfect model of the actuator dynamics and that there are no disturbances and errors entering that dynamic.

The states and controls are constrained by upper and lower limits on each variable. The state constraints are softened using slack variables, $\varepsilon_i$, and our experience is that one slack variable per state and time step is required to obtain good performance. We have chosen to use a quadratic penalty on the slack variables even though this causes the active constraints to be violated to some small extent also in cases when it is not necessary [Maciejowski, 2002]. The reason for this is that in earlier projects we have experienced difficulties in achieving excellent tuning and smooth control signals using linear penalty functions.

The MPC controller is rewritten into a quadratic program of the form

$$\begin{aligned} \underset{z}{\text{minimize}} \;\; & z^T H z + z^T G x_k \\ & \text{subj. to} \\ & b_{min} \leq Az \leq b_{max} \\ & z_{min} \leq z \leq z_{max} \end{aligned}$$

where $z^T = [u_k, u_{k+1}, u_{k+2}, \ldots, u_{k+N-1}, \varepsilon_0, \varepsilon_1, \ldots, \varepsilon_N]$ and solved online using the solver qpOASES [Ferreau et al., 2014].

At this point it is probably good to highlight that in the MPC formulation (8.1) we have not used any terminal state constraint, which we know from Section 4.2.1 is in general necessary to guarantee stability when the system is unstable and constrained. There are several reasons for this. The first one is that the multi-frequency approach we have taken with the MPC controller derived in one frequency and implemented in another makes it impossible to prove recursive feasibility and hence also stability using standard approaches. Secondly, even if we could prove stability for every linearized prediction model the controller is gain scheduled over the flight envelope and to the authors knowledge there does not exist any systematic ways to guarantee stability for this kind of setup.

## 8.3 Tuning of the MPC controller

In this section we will concentrate on the methodology we have used to tune the MPC controller (8.1). The parameters to tune are the penalty matrices,

$P$, $Q$ and $R$, of the objective function (8.1a), the prediction horizon, $N$, and prediction model discretization time step as well as the penalty matrices, $Q_{KF}$ and $R_{KF}$, in the Kalman filter.

The constraints in the MPC controller are given by external requirements as

$$-10^\circ \leq \alpha \leq 18^\circ, \quad -75^\circ/s \leq q \leq 75^\circ/s, \quad -28^\circ \leq \delta_e \leq 28^\circ, \quad -50^\circ \leq \delta_c \leq 25^\circ$$

The standard tool for flight control law design at Saab has for a long time been LQ control and an extensive and proven methodology exists for tuning LQ controllers. Additionally many of the design requirements for flight control systems are formulated either in continuous time or in frequency domain. Therefore the approach taken here is to perform an LQ design in continuous time for each of the design points in the flight envelope and then when we have satisfactory linear properties translate the design into MPC equivalent penalty matrices.

We have chosen to perform the LQ design in 18 envelope points in the subsonic region up to an altitude of 6 kilometers. The speeds have been chosen to Mach 0.2, 0.3, 0.4, 0.6, 0.8 and 0.9 and the altitudes 1 km 3 km and 6 km.

In each envelope point a continuous time LQ controller is tuned that meets the basic performance requirements for category IV aircraft in MIL-F-8785C [1980]. If the discretized equivalent penalty matrices $Q_d$, $R_d$ and the Ricatti solution, $P_d$, were to be implemented in a standard MPC controller we would regain the performance of the LQ controller when no constraints on the system are active. However since we use a multi frequency approach and have designed the MPC controller in a lower frequency (10 Hz) than it is implemented in (120 Hz) this will no longer hold. An example of this is shown in Figure 8.1. In the figure we compare the continuous time response of an LQ controller with different discrete implementations and we can see that if we implement the continuous LQ feedback in 120 Hz (the magenta lines) the response is basically the same as the continuous time response. We also obtain the same response if we design a discrete LQ controller with the same penalty matrices for a sample time of 0.1 s and implement it at 10 Hz (cyan lines). Note though that the control signal is different because it is now piecewise constant in 10 Hz.

If we now take the discrete LQ controller designed for a sample time of 0.1 s and implement it at 120 Hz we will obtain the response corresponding to the green lines in figure 8.1. We clearly see that the response is slower than the original continuous time design and the explanation is straightforward. It is because the magnitude of the control signal is smaller in the discrete time case and it needs to be active for the longer sampling time for it to have the intended effect on the system. When implemented in a much higher frequency the effect of each control action is much smaller and hence the response becomes slower.

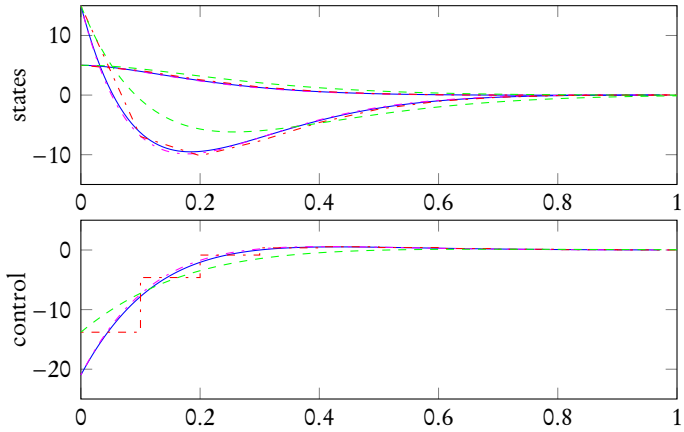


**Figure 8.1.** An example of the response of a continuous time LQ controller (blue), its discrete implementation at 120 Hz (magenta), a discrete time LQ controller at 10 Hz (red) and the 10 Hz discrete LQ implemented at 120 Hz (green).

If we want to be able to use the LQ design methodology together with the multi frequency approach in the MPC implementation we need to compute new penalty matrices such that the unconstrained closed loop response of the MPC controller running at 120 Hz but using a discrete time model based on 10 Hz sampling is equal to that of the continuous time LQ design.

What we would like is to find if there exist a set of penalty matrices $P$, $Q$ and $R$ such that the 10 Hz discrete time LQ feedback gain, $K_d$, is equal to the continuous time feedback gain, $K$, calculated using the penalty matrices $Q_{LQ}$ and $R_{LQ}$. If the feedback gains are equal then the discrete time controller will result in the same closed loop response as the continuous time controller when implemented in 120 Hz. As it turns out we can in fact formulate this problem as an optimization problem, which is linear in the decision variables $P$, $Q$ and $R$.

To see this we first note that the optimal discrete time LQ feedback is given by [Åström and Wittenmark, 1996]

$$K_d = (B^T P B + R)^{-1} B^T P A \tag{8.5}$$

where $P$ is the solution to the discrete time Riccati equation

$$P = A^T P A + A^T P B (B^T P B + R)^{-1} B^T P A + Q \tag{8.6}$$

Now, if we want the feedback gain, $K_d$, to be equal to a previously designed continuous time feedback, $K$, we have

$$\begin{aligned} K - K_d = 0 \Leftrightarrow \\ K - (B^T P B + R)^{-1} B^T P A = 0 \Leftrightarrow \\ (B^T P B + R) K - B^T P A = 0 \end{aligned}$$

Furthermore we can rewrite the Riccati equation (8.6) using (8.5) as

$$\begin{aligned} P &= A^T PA + A^T PB \underbrace{(B^T PB + R)^{-1} B^T PA}_{K_d} + Q \\ &= A^T PA + A^T PBK_d + Q \\ &= A^T PA + K_d^T (R + B^T PB) K_d + Q \end{aligned}$$

where we in the last equality have used that

$$A^T PB = A^T PB(R + B^T PB)^{-1}(R + B^T PB) = K_d^T (R + B^T PB)$$

since $(R + B^T PB)$ is symmetric. If we now have $K_d = K$ we finally obtain

$$P = A^T PA + K^T (R + B^T PB) K + Q$$

To find the suitable penalty matrices we thus need to solve the following system of equations

$$\left(B^T PB + R\right) K - B^T PA = 0 \tag{8.7}$$

$$A^T PA + K^T \left(R + B^T PB\right) K + Q = P \tag{8.8}$$

$$Q \geq 0 \tag{8.9}$$

$$P \geq 0 \tag{8.10}$$

$$R \geq 0 \tag{8.11}$$

However, there might not exist a solution to these equations and hence we make try to find the best possible choice by minimizing the two-norm of (8.7) subject to the constraints (8.8) - (8.11).

$$\underset{P,Q}{\text{minimize}} \quad \left\| \left(R + B^T PB\right) K - B^T PA \right\|_2 \tag{8.12a}$$

subj. to

$$P = Q + A^T PA - K^T \left(R + B^T PB\right) K \tag{8.12b}$$

$$Q \geq 0 \tag{8.12c}$$

$$P \geq 0 \tag{8.12d}$$

$$R = R_{LQ} \tag{8.12e}$$

The first thing we can note about the problem (8.12) is that it is linear in the unknown matrices, $P$, $Q$ and $R$. Secondly, in order to set the scaling of the solution we need to fix one of the elements in one of the variables and since we penalize the both control commands equally we can thus introduce the constraint (8.12e).

To summarize the above discussion we have tuned the MPC controller by first tuning a continuous time LQ controller in each envelope point. Then found suitable penalty matrices for the MPC controller by solving the optimization problem (8.12) to ensure the MPC controller acts as the LQ controller when no constraints are active, despite being designed in 10 Hz but running in

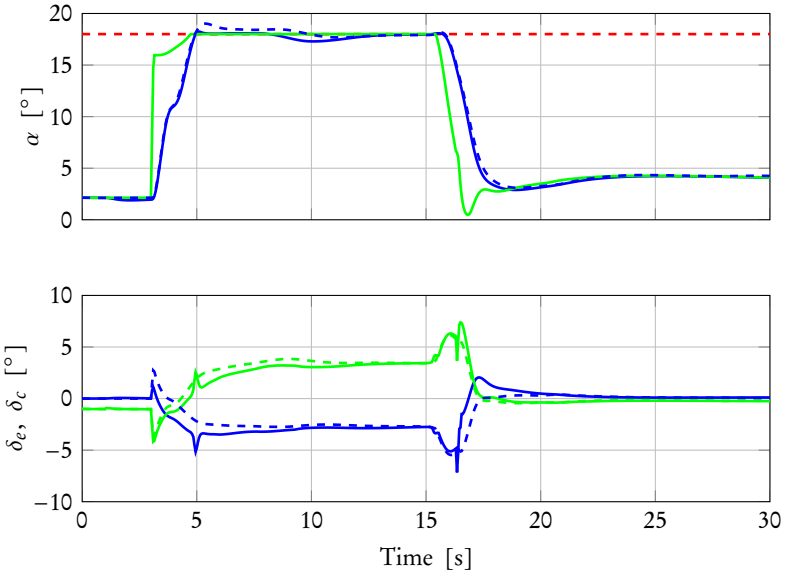


**Figure 8.2.** Comparison of angle of attack response and control surface commands of the proposed MPC controller and the baseline LQ controller at Mach 0.6 and altitude 2 km.

120 Hz. The matrices are then scheduled over the entire flight envelope together with the system dynamics $A$, $B$ and $E$.

The penalty on the slack variables has been chosen to be constant over the entire flight envelope and to be significantly larger than the other penalty matrices. We have without any closer analysis manually tuned the penalty to $\gamma = 10^5$ in order to have good constraint satisfaction but not an overly aggressive controller.

The tuning of the Kalman filter has been a bit more ad hoc. We have simply fixed the penalty matrix for the measurement noise, $R_{KF} = I$, and tuned the penalty matrix for the process noise, $Q_{KF}$, until the filter was fast enough and the controller achieved integral action. The penalty matrices were also chosen constant over the flight envelope.

## 8.4 Simulator testing

The MPC controller derived in the previous sections has been implemented and tested in Saabs in-house simulation environment ARES, both in desktop simulations with a virtual pilot and in a real simulator with human pilots.

Let us start by comparing the response of the MPC controller derived in the previous sections with the ARES baseline LQ controller (see Section 3.4.1)

with the same tuning using the virtual pilot. We evaluated the performance by looking at a maximum performance turns performed by a virtual pilot in different parts of the flight envelope. The angle of attack response for a turn from Mach 0.6 and altitude 2 km is shown in Figure 8.2.

The upper part of the figure shows the angle of attack response of the MPC controller (blue solid line) and the LQ controller (blue dashed line), the angle of attack limit (red dashed line) and pilot angle of attack command (green). In the lower part of the figure the control surface commands for the canards (blue) and the elevons (green) are shown. In the figure we can clearly see that even though the LQ controller is tuned to not give any overshoot of the angle of attack limit it can not manage to keep the angle of attack within the prescribed limit. The main reason for this is that Mach 0.6 and altitude 2 km is in between two envelope points where the LQ controller has been tuned and the nonlinearities and model errors that are present in the true aircraft dynamics makes the performance in between the design points worse. The overshoot of the angle of attack limit is almost one degree for the LQ controller while the MPC controller on the other hand manages to keep the limit very well.

If we look at the control surface commands (the lower part of Figure 8.2) at time five seconds, we can see that the MPC controller react with spikes in the two control commands. This is resulting from that the angle of attack prediction surpasses the limit and the slack penalty kicks in.

The above results show the performance of the MPC controller in one envelope point. To evaluate a more dynamical scenario we perform the same maneuver starting from a high-speed point, see Figure 8.3. In this case the speed drops from the initial Mach 0.9 almost down to Mach 0.45 which means that the MPC controller changes throughout almost its entire scheduling envelope in speed. In the first part of the maneuver the command is not at the maximum angle of attack because here at higher speeds the command is set to a maximum load factor command (which corresponds to a lower angle of attack). But as the speed reduces the angle of attack increases to reach its maximum value at corner speed, i.e., when the angle of attack and load factor both are at their respective maximum values. We can see that even for this more dynamical case the MPC controller works well and manages to limit the angle of attack more effectively than the standard LQ controller.

Considering also the same maneuver in a low speed and low altitude case we obtain the results in figure 8.4.

In the above cases the maneuvers are either very dynamical in the sense that the maneuver ranges over a large range of speeds or they are initiated in envelope points that are not tuning points of the controller. This means that the prediction model is less accurate and the performance of the controller is not optimized. If we instead look at the performance close to a design point we can see that the performance of the MPC controller and LQ controller is

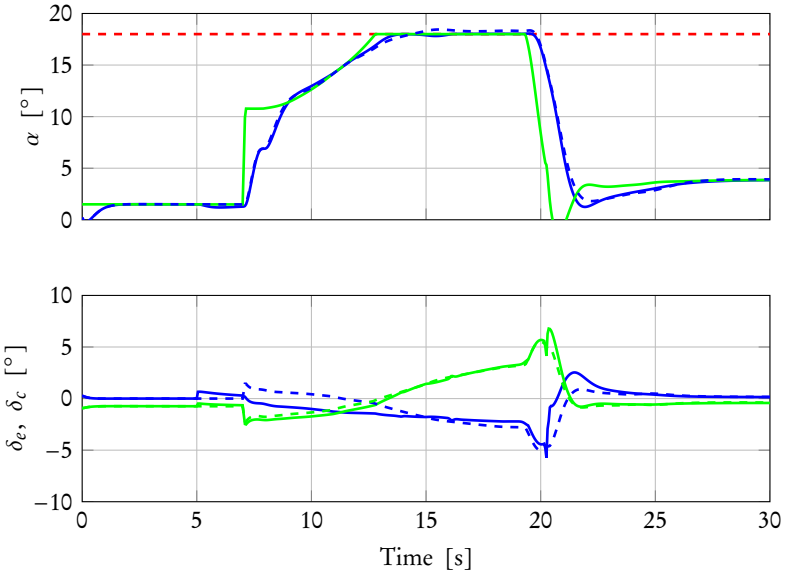


**Figure 8.3.** The corresponding simulation as in Figure 8.2 but now starting at Mach 0.9 and altitude 6 km.

as expected very similar, see Figure 8.5. From this simulation we can see that when the aircraft response is not affected by the constraints and the response of the MPC controller is identical to that of the LQ controller. Hence our tuning strategy of calculating equivalent MPC penalty matrices makes the closed loop system behave just like the linear design in the unconstrained cases.

To obtain a real proof of concept for the performance of the proposed controller we have implemented the MPC controller also in Saab's main flight mechanical simulator. This simulator is a realtime software simulator that uses software models to simulate all aircraft sensors, actuators and computer systems. The simulator has a real cockpit with control stick, throttle lever and rudder pedals in a dome like projector environment.

The controller is executed on a Linux operating system and the maximum execution time has been measured to approximately 5 ms using the qpOASES solver. For a final implementation off-the-shelf software for solving the QP should not be used but instead tailor made software for fast solution of MPC problems such as, e.g., Wang and Boyd [2010] should be utilized. One could also consider approaches such as those found in Gibbens and Medagoda [2011], Richards et al. [2009], Zavala and Biegler [2009] or Hartley et al. [2012] to improve the realtime implementation.

To evaluate the performance of the MPC controller a series of high angle of attack maneuvers have been performed. One such maneuver is the so called bleed off turn (BOT) which is a decelerating turn at maximum available angle

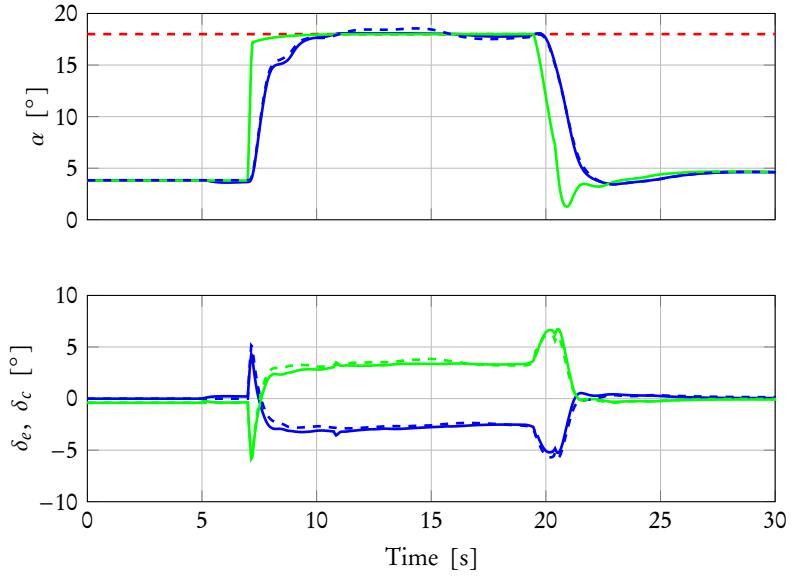


**Figure 8.4.** The corresponding simulation as in Figure 8.2 but now starting at Mach 0.4 and altitude 1 km.

of attack. Figure 8.6 shows a BOT starting from Mach 0.6 at altitude 5 km.

Figure 8.7 shows another BOT maneuver now starting from Mach 0.5 and at an altitude of 1 km.

Both these figures verifies the MPC controllers ability to maintain the angle of attack limit throughout the whole BOT maneuver. If we compare the performance achieved in the simulator with the desktop simulations we can see that it is very similar. There are no overshoot of the angle of attack limit in any of the performed maneuvers. In the maneuvers in Figure 8.6 and 8.7 we can see on the control surface deflections that the MPC controller reacts when the angle of attack is approaching the limit as two spikes in the control surface deflections around 2.5 seconds.

Note that the plotted pilot angle of attack command is not the internal command in the controller since this was not available to measure but an offline reconstruction of the command from the control stick input.

Figure 8.8 shows the angle of attack response and control surface deflections to a pitch input command. Also in this maneuver the MPC controller exhibits very good performance and it limits the angle of attack very well. In this maneuver we can discern small chattering in the control surface deflections. This chatter comes from the MPC controller and it comes from the fact that in the simulator implementation of the MPC controller we updated the prediction model and penalty matrices only every tenth iteration of the controller. This

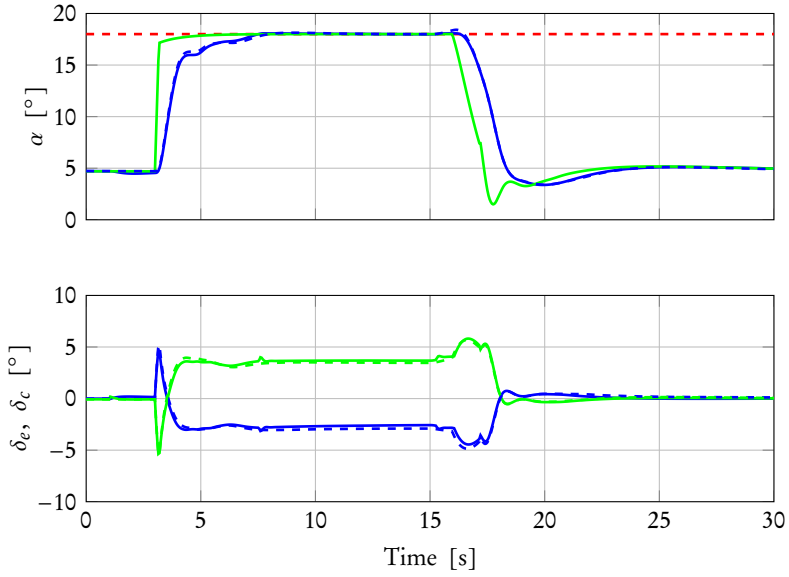


**Figure 8.5.** The corresponding simulation as in Figure 8.2 but now starting at Mach 0.4 and altitude 3 km.

lead to a small jump in the control signal, which becomes larger the more seldom the prediction model is updated. Vibrations of this kind is of course not desirable and needs to be handled if one wants to implement a lower update rate of the prediction model.

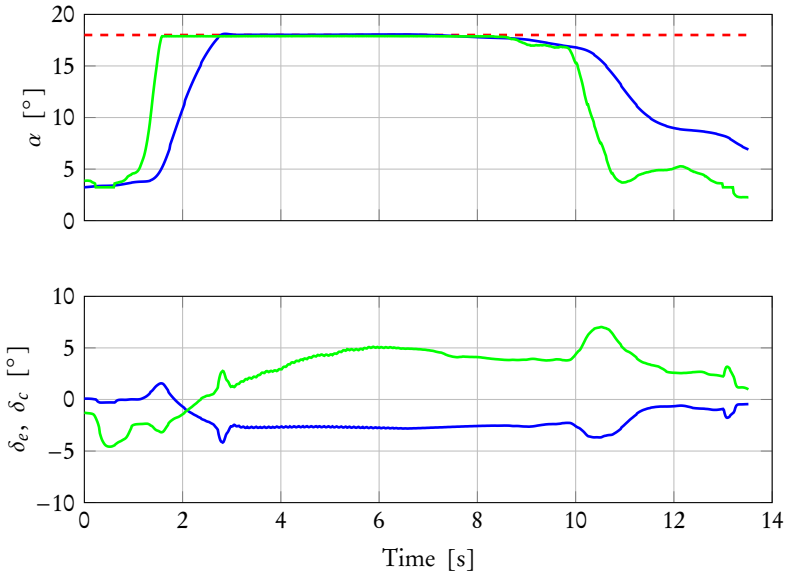


**Figure 8.6.** A BOT starting at Mach 0.6 and altitude 5 km performed in the simulator. Upper figure shows the angle of attack response (blue), pilot command (green) and the angle of attack limit (red). Lower figure shows the canard (blue) and elevon (green) control surface deflections.

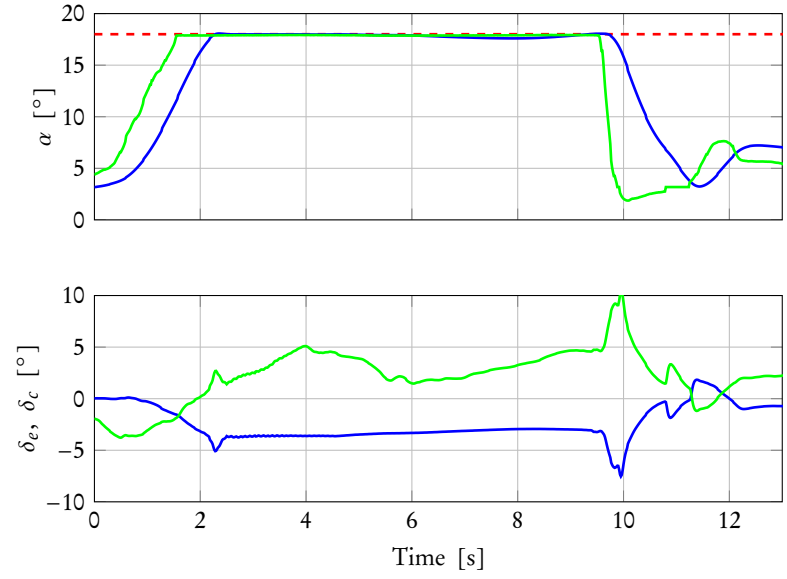


**Figure 8.7.** The corresponding maneuver as in Figure 8.6, now starting at Mach 0.5 and altitude 1 km.

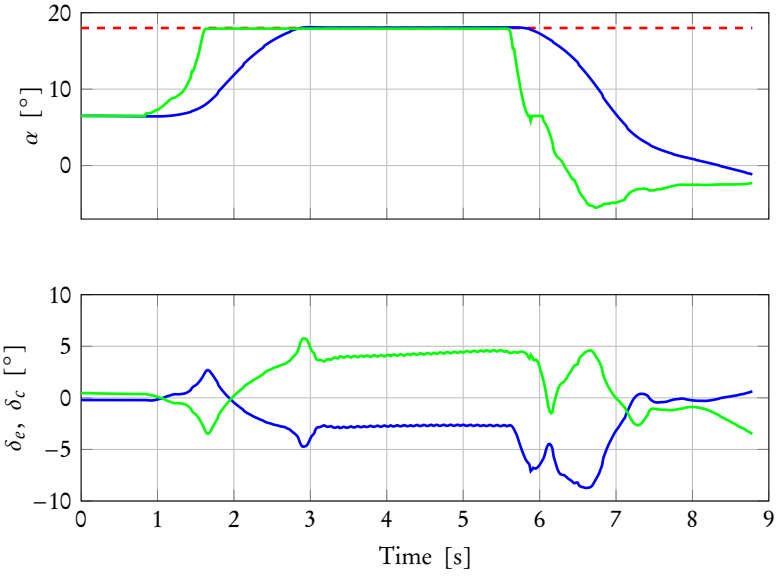


**Figure 8.8.** A pitch up command from Mach 0.3 and altitude 1 km. Upper figure shows the angle of attack response (blue), pilot command (green) and the angle of attack limit (red). Lower figure shows the canard (blue) and elevon (green) control surface deflections.

# 9

# Aircraft maneuver limiting using command governors

In this final chapter we will take a slightly different approach to the concept of maneuver limiting and envelope protection. As discussed in the introduction there exist several different ways of handling constraints on the system and model predictive control is only one of those. In this chapter we will instead investigate a closely related method called reference governor, or command governor, as a method for maneuver limiting in a realistic fighter aircraft simulation environment. We will apply the command governor design to the flight dynamical simulation environment, ARES, and we will augment the ARES baseline LQ controller, as described in Section 3.4.1, with a command governor to limit any angle of attack and load factor limit overshoots.

This chapter is based on the journal paper

> Daniel Simon, Ola Härkegård, and Johan Löfberg. Command Governor Approach to Maneuver Limiting in Fighter Aircraft. *Journal of Guidance, Control, and Dynamics*, 40(6):1514–1527, 2017b.

which also has been published as the conference paper

> Daniel Simon, Ola Härkegård, and Johan Löfberg. Angle of Attack and Load Factor Limiting in Fighter Aircraft using Command Governors. In *AIAA Guidance, Navigation, and Control Conference, AIAA SciTech Forum*, 2017a.

## 9.1 Introduction

Due to different uncertainties, model errors and disturbances on the real aircraft, the true closed loop response is never the ideal response that we

designed the controller for (see Section 3.4.1). Throughout the flight envelope the actual closed loop response can have an overshoot of the design limits. This overshoot is highly undesirable since it can over stress the aircraft or put it into an unsafe state. Therefore we will add a command governor to the pilot commanded $\Delta n_{z,cmd}$ which alters the command to a new reference command that, as far as possible, will ensure that the design limits on the angle of attack and normal load factor are not exceeded.

There can be several benefits from using reference and command governors for the maneuver limiting task compared to using model predictive controllers. First, the governors can be used as add-ons to existing legacy controllers so there is no need to redo the complete design. Furthermore the nominal inner loop controller can be tuned to achieve good performance in the nominal case, e.g., use nonlinear feedbacks to linearize the closed loop system, and the governor can focus on the maneuver limiting task. It also gives a good modularity such that one can replace parts of the control system without the need to redo all of the design. Last but not least from a flight safety perspective it might be easier to certify optimization algorithms running in an outer loop, which can be turned off in case of failures without affecting stability.

While model predictive controllers have been extensively investigated for flight control applications, see Chapter 8 and the references in it, very little research has been performed on applying reference and command governors to flight control design and maneuver limiting, see e.g., Famularo et al. [2008], Kolmanovsky and Kahveci [2009], Martino [2008], Petersen et al. [2013], Ye et al. [2015], Zinnecker et al. [2009]. Most of these papers consider simplified conditions with simpler linear or nonlinear system and no complex, full scale, nonlinear and time varying simulation environments.

In the papers by Petersen et al. [2013] and Zinnecker et al. [2009] the authors apply reference governors to the control of hypersonic vehicles. In the paper by Zinnecker the focus is mainly on input constraints. Kolmanovsky and Kahveci [2009] uses a reference governor to handle control actuator limitations of a UAV glider and compare this to an adaptive anti-windup scheme, and in the paper by Martino [2008] the author investigates command governors for handling amplitude and rate constraints on a small commercial aircraft. Ye et al. [2015] investigate reference governors for maneuver limiting in high angle of attack maneuvers. They investigate and compare static and dynamic reference governors with a reference governor structure based on a step response model of the closed loop system. The different reference governors are evaluated on a linear aircraft model and the conclusion is that the dynamic reference governor performs much better than the static reference governor but it has a complex maximal output admissible set. The governor structure with a step response model has comparable performance with the dynamic reference governor but the complex admissible set is replaced with a finite horizon approximation which is much simpler to implement. Famularo et al. [2008] thoroughly

investigate a robust command governor approach for constrained control of aircraft. They apply the robust command governor to one fighter aircraft and one small commercial aircraft with both input and output constraints. The results are promising but the simulations are only done in one envelope point and thus lack the added complexity of changing dynamics over the envelope.

The drawback with robust command governor designs is that there must be a margin to the constraint at all times to account for disturbances. In our case, when we want to use the command governor to achieve maneuver limiting, this is not an acceptable solution. Instead we need to consider soft constraints, achieved using slack variables. An example of a command governor structure utilizing soft constraints via slack variables are given in Kalabic et al. [2013]. Here the authors utilize the slack variables and their penalties to prioritize between various important constraints.

An interesting approach to command governor structure is proposed in Li et al. [2014]. The authors consider only second-order systems with input time delays and show that for these systems it is sufficient to consider only four distinct time points per output in the prediction horizon to guarantee constraint satisfaction. This reduces the computational complexity of the online solution of the resulting quadratic program, however it requires that the reference is kept constant over the prediction horizon.

In this chapter we extend the previous research and go beyond what has been done before. We implement and analyze command governors for maneuver limiting in ARES (see section 3.4), the most complex aircraft models available at Saab Aeronautics, surpassed only by real flight testing. Due to the changing dynamics of the aircraft over the flight envelope, we can not use the classical structure of the command governor. Instead we have to adopt command governor structure much like the reference governor based on a step response model in Ye et al. [2015].

We will start by giving a brief introduction to reference and command governors in Section 9.2 and then in Section 9.3 we discuss the different architectural design choices we have investigated. The final design and some examples from the simulations are presented and discussed in Section 9.4.

## 9.2 Reference and command governors

A command governor, or reference governor, is a device that takes the reference input and alters it based on the current estimate of the system state, $\hat{x}(t)$, such that the output from the system remains within certain limits. The general structure of the closed loop system, from Figure 3.4, and command governor is shown in Figure 9.1.

The simplest form of a reference governor is the static reference governor Gilbert et al. [1994], Gilbert and Tan [1991], Gilbert et al. [1995] which optimizes a scalar gain, $\gamma$, such that when $\bar{r}(k) = \gamma r(k)$ is applied to the

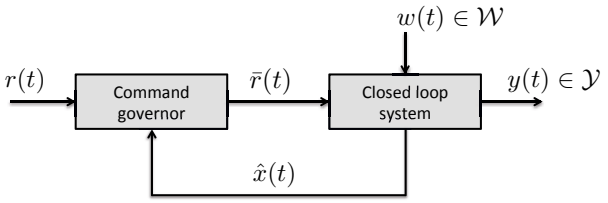


**Figure 9.1.** The reference governor general structure.

system the output constraints, $y(k) \in \mathcal{Y}$, are satisfied. This is done by solving the optimization problem Gilbert et al. [1995]

$$\underset{\gamma \in [0,1]}{\text{maximize}} \ \gamma \ \text{ subj. to } \ Ax(k) + B\gamma r(k) \in \mathcal{O}_\infty, \quad Cx(k) + D\gamma r(k) \in \mathcal{Y} \tag{9.1}$$

where $\mathcal{O}_\infty$ is the *maximal output admissible set* Kolmanovsky and Gilbert [1998] and where $x(k) = \hat{x}(t)$.The maximal output admissible set for a system $x(k+1) = Ax(k) + Bw(k)$, $y(k) = Cx(k) + Dw(k)$ is formally defined as

$$\mathcal{O}_\infty = \{x(k) \in \mathbb{R}^n \mid y(k+i) \in \mathcal{Y} \ \ \forall \, w(k+i) \in \mathcal{W} \ i \in Z^+\} \tag{9.2}$$

This can easily be generalized to a definition where the combination of $x(k)$ and a constant reference, $\bar{r}$, is such that $y(k) \in \mathcal{Y}$ for all future time steps.

Since the static reference governor can suffer from oscillations a dynamic reference governor was developed Gilbert et al. [1995] in which the reference was parameterized as

$$\bar{r}(k+1) = \bar{r}(k) + \gamma(r(k) - \bar{r}(k)) \tag{9.3}$$

and the optimization maximizes $\gamma \in [0, 1]$ such that the successor state is in the admissible set.

The dynamic reference governor is closely related to the more flexible command governor Bemporad et al. [1997] where the reference $\bar{r}$ instead is parameterized as

$$\bar{r}(k+i) = \gamma^i \mu(k) + \nu(k) \tag{9.4}$$

where $\gamma$ is a fixed constant $\gamma \in [0, 1)$ and the optimization variables are $\mu(t)$ and $\nu(t)$. The optimization problem is now formulated as

$$(\mu^*, \nu^*) = \underset{\mu,\nu}{\arg\min} \ \ ||\mu||_Q^2 + ||\nu - r(k)||_R^2 + \sum_{i=0}^{\infty} ||y(k+i) - \nu||_P^2$$
$$\text{subj. to} \quad y(k+i) \in \mathcal{Y} \ \forall \ i \in Z^+$$

where $\mu \in \mathbb{R}^p$ and $\nu$ shall be selected from the set of constant signals such that the output in equilibrium is $y \in \mathcal{Y}$.

A generalization of the command governor is the extended command governor Gilbert and Ong [2011], Kalabic et al. [2011] in which the decaying se-

quence $\gamma^i \mu(k)$ in the parameterization of $\bar{r}(k)$ is replaced by a fictitious system output, $\bar{C}\bar{x}(k)$, where the fictitious state evolve according to $\bar{x}(k+1) = \bar{A}\bar{x}(k)$. The matrix $\bar{A}$ should be stable but otherwise the matrices $\bar{A}$ and $\bar{C}$ can be chosen arbitrarily and it is easy to see that with a certain choice of $\bar{A}$ and $\bar{C}$ we recover the original command governor formulation. Gilbert and Ong Gilbert and Ong [2011] propose a shift sequence for the fictitious system, i.e.,

$$\bar{A} = \begin{bmatrix} 0 & I & 0 & \dots & \\ 0 & 0 & I & 0 & \dots \\ & & 0 & \ddots & \\ & & & 0 & I \\ & & & 0 & 0 \end{bmatrix}, \quad \bar{C} = \begin{bmatrix} I & 0 & 0 & \dots \end{bmatrix}$$

while Kalabic et al. Kalabic et al. [2011] propose to use Laguerre sequences.

These basic principles discussed above have been extended to cover, e.g., nonlinear systems, robust reference governors and other types of structures. An excellent survey of the different types of reference and command governors is given in the paper by Kolmanovsky et al. [2014].

## 9.3 Command governor design

Let us start by discussing the different architectural choices that we have to make in order to find a suitable command governor design for the maneuver limiting application.

### 9.3.1 N-step prediction approach

The use of an output admissible set, $\mathcal{O}_\infty$, in our implementation of the command governor is not suitable since the dynamics of the aircraft varies over the altitude and speed envelope. The output admissible set, which is calculated based on the dynamics has to be either calculated offline in advance based on a set of design points and then switched between online, or recalculated in every iteration of the command governor. The approach with online recalculation is to complex since it would require to much computational power in the iterative procedure of determining the polytopic shape of the set. In fact, as discussed earlier, the number of inequalities describing the polytopic set might not even be finitely determined [Kolmanovsky and Gilbert, 1998]. The first approach would not require the extensive online calculations but instead the sets must be *robust output admissible* with respect to the model errors that come from the changing dynamics in between the design points and this defeats the purpose of our implementation. Furthermore no theoretical stability guarantees can be made when changing, or recalculating, the admissible set second order between two iterations of the controller.

An alternative approach is similar to the one in reference Ye et al. [2015]. Simply constrain the N-step forward predictions of the model output, i.e.,

$$\underline{y} - \varepsilon(k+i) \leq y(k+i) \leq \bar{y} + \varepsilon(k+i), \quad \forall\, i = 1, \ldots, N \tag{9.5}$$

where $\underline{y}$ and $\bar{y}$ are the upper and lower limits for the output and $\varepsilon(k+i) \geq 0$ is a slack variable added to soften the constraints. This of course does not give any guarantees that the constraints can be fulfilled for all future time steps, but it serves our purpose since we want to use it as a soft maneuver limit. The approach discussed in Li et al. [2014] would also be an alternative that is more theoretically motivated but, as discussed later in this section, it requires a constant input reference which we decided not to adopt.

### 9.3.2 Selection of discretization technique

The use of the N-step prediction as constraints requires that the prediction model is explicitly implemented in the command governor. A very simple choice is to use a discrete time implementation of the desired closed loop response of the aircraft with the nominal controller.

$$G_m(s) = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2}$$

From the nominal controller we obtain, in each sample time, the current desired damping, $\zeta$, and frequency, $\omega_0$. The discretization can be done in several different ways, e.g., using Euler forward, $s = \frac{1}{T_s}(q-1)$, Tustin's approximation, $s = \frac{2}{T_s}\frac{(q-1)}{(q+1)}$, or zero-order-hold. Euler forward gives very simple analytical expressions for how the coefficients, $a$, $b$ and $c$ in the discrete time step response model, $y(k+1) = ay(k) + by(k-1) + cu(k-1)$, depend on the current damping and frequency, which are frequently updated.

$$a = 2(1 - T_s\zeta\omega_0), \quad b = 2\zeta\omega_0 T_s - 1 - w_0^2 T_s^2, \quad c = w_0^2 T_s^2$$

However the Euler discretization requires a sufficiently high sampling rate to be accurate enough, see Figure 9.2. A high sampling rate requires a large number of prediction steps, $N$, to achieve a sufficiently long prediction time.

The objective of having a long prediction time is that we want the command governor to react earlier when a constraint violation is predicted.

If we instead use Tustin's approximation or zero-order-hold discretization we can implement a lower sampling rate in the prediction model and thus reduce the number of prediction steps required to achieve the same prediction time. Although the zero-order-hold is the most accurate approximation, it is exact in the sampling instants, it doesn't have the simple analytical expressions for the dependence of damping and frequency in the model coefficients.

Implementing both the Euler forward discretization and Tustin's approximation reveals that the longer prediction time that can be achieved with Tustin (with

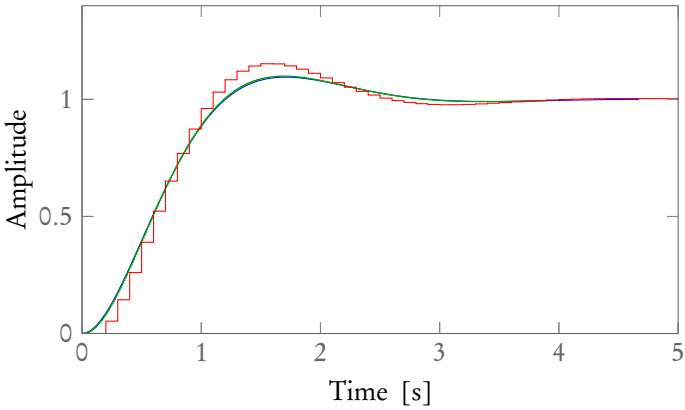


**Figure 9.2.** The step response for a second order continuous time system (blue line) with $\zeta = 0.6$ and $\omega_0 = 2.3$ together with the corresponding discrete time system using Euler forward discretization with sampling time $T_s = 0.01$ s (green line) and $T_s = 0.1$ s (red line).

the same number of prediction steps as with Euler forward) does not give any improvement in the performance of the command governor compared to the Euler approximation, see Figure 9.3. In this figure we can see that the load factor responses are almost identical for the two discretization methods. The reason for this is probably that the desired closed loop response model, $G_m(q)$, that we use as prediction model, does not accurately enough model the true closed loop system to benefit from the longer prediction time that we obtain with Tustin's approximation.

### 9.3.3 Model error correction term

An attempt to solve the above problem can be to add a model correction term, $\hat{d}$, to the prediction model

$$y_m(k) = G_m(q)\bar{r}(k) + \hat{d}(k)$$

and then estimate the correction term online. A simple way to do this has been suggested in literature to handle nonlinear systems Kolmanovsky et al. [2014]. This technique uses the output from the linear prediction model of the system

$$y_{linear}(k) = G_m(q)\bar{r}(k)$$

and compare that to the true closed loop system output, $y(k)$, to estimate a constant model correction term

$$\hat{d}(k+i) = y(k) - y_{linear}(k) \quad \forall\, i = 0, \ldots, N$$

and then add this to the prediction model output.

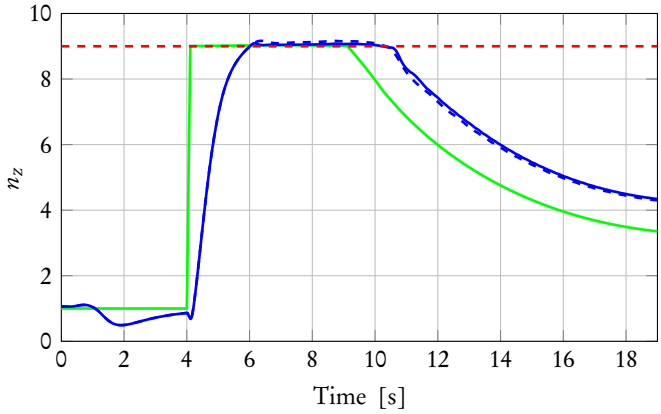


**Figure 9.3.** Load factor response for the command governor with Euler discretization of the response model (solid blue line) and with the Tustin discretization (dashed blue line). Green line is pilot commanded load factor and the red dashed line is the load factor limit.

However when implementing this model error estimation in the simulation environment only very little performance is gained, see Figure 9.4.

For our transfer function model the correction factor, $\hat{d}$, adds a constant offset correction to the predictions and does not capture dynamical model errors such as e.g., errors in the damping. Since the nominal closed loop system has integral action the steady state error is minimal and the true model error probably comes from dynamical properties.

### 9.3.4 Selection of objective function and parameterization of the reference

As discussed in the beginning of this section, there exist several different possibilities to parameterize the command governor output $\bar{r}(k)$ and to formulate the objective function. An intuitive and straight forward formulation of the objective function is to simply penalize the difference between the pilot commanded reference, $r(k)$, and the applied reference, $\bar{r}(k)$, as

$$\sum_{i=0}^{N} (\bar{r}(k+i) - r(k))^2 \tag{9.6}$$

This objective function can be used with both static and dynamic reference governor formulations. However for our purposes the standard formulation of the dynamic reference governor (9.3) is not suitable since it requires that the governor be robustly designed with respect to constraint violations. If, e.g., a disturbance enters the system at time $k$ the parameterization of $\bar{r}$ does not allow it to be reduced from the value at time $k$, since $\gamma \geq 0$, and hence

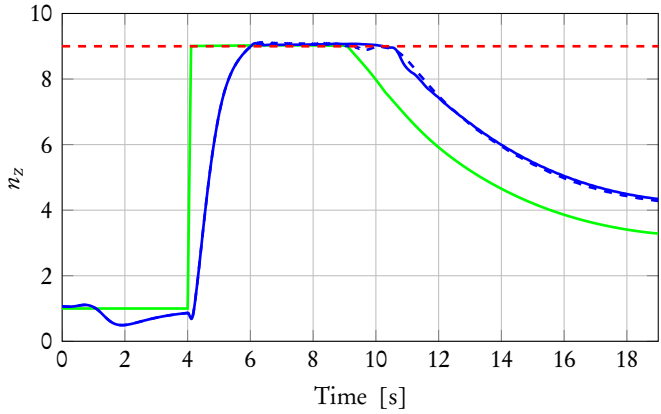


**Figure 9.4.** Load factor response for the command governor with model error estimator (dashed blue line) and without the model error estimator (solid blue line). Green line is pilot commanded load factor and the red dashed line is the load factor limit.

cannot counteract the disturbance. In our application of maneuver limiting we want the reference governor to allow the output $y(k)$ to reach the limit, but as far as possible not overshoot it. Therefore do we want to allow the applied reference $\bar{r}$ to be reduced if disturbances or model errors predict an overshoot.

If we instead choose to use a static reference governor, then $\bar{r}(k) \in \mathcal{Y}$ with $\bar{r}(k+i) = \bar{r}(k)\ \forall\ i = 1, \ldots, N$. This is a very attractive choice since the optimization problem then only has one free variable to optimize over and it becomes almost trivial to solve. However the static reference governor might suffer from oscillations in the closed loop response Gilbert et al. [1995]. Even though our simulations do not indicate that oscillations could arise this study is not enough to make conclusive statements about that.

An alternative is to let $\bar{r}(k+i)$ be a sequence of free optimization variables in an MPC like fashion. This gives the algorithm maximum flexibility in the choice of future applied reference inputs but with the cost of increasing the dimension of the optimization variable from 1 to $N$. The simulations of the command governor do not indicate a large enough performance gain to motivate such an increase in complexity. In fact for some simulations there is no performance increase by using a sequence of reference signals in comparison to using a constant reference, see Figure 9.5.

A flexible but yet simple alternative is also the command governor Bemporad et al. [1997] or the extended command governor Gilbert and Ong [2011] parameterization of $\bar{r}$. The command governor gives the algorithm good flexibility but with limited complexity increase. For the command governor approach with $\bar{r}(k+i) = \gamma^i \mu(k) + \nu(k)$ we can adopt a similar formulation of

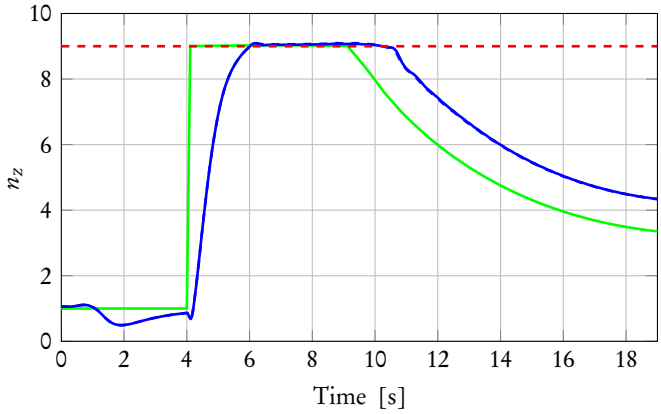


**Figure 9.5.** A comparison between a command governor with a constant reference signal and one with a sequence of reference signals in the prediction model. Blue dashed line is the command governor with a reference sequence and the solid blue line is the command governor with a constant reference over the prediction horizon. Green line is pilot commanded load factor and the red dashed line is the load factor limit.

the objective as in Bemporad Bemporad et al. [1997] and formulate it such as

$$\underset{\mu,\nu}{\text{minimize}} \;\; \beta_1 \mu(k)^2 + \beta_2 (\nu(k) - r(k))^2 \tag{9.7}$$

where the first term can be viewed as a penalty on the changes in the sequence of $\bar{r}$ and the second term is penalty on the stationary deviation from desired reference.

Finally the objective function must also include some penalty on the slack variables, $s(k+i)$, used to soften the constraints (9.5). The most common choices are to use a quadratic or a linear penalty. The linear penalty has the advantage that if it is high enough it will force the slack to be zero if there exist such a feasible solution and only non-zero otherwise. On the other hand with a linear penalty the control signal has a tendency to have more abrupt changes while it is fairly smooth with quadratic penalty on the slack.

## 9.4 Simulation results

In this section we will discuss the achieved simulation results when implementing an outer loop command governor in the ARES simulation environment at Saab Aeronautics for maneuver limiting. The implemented command governor is formulated as the minimization problem

$$\underset{\mu,\nu,\varepsilon_\alpha,\varepsilon_{n_z}}{\text{minimize}} \;\; \beta_1 \mu(k)^2 + \beta_2 (\nu(k) - \Delta n_{z,cmd}(k))^2 + \sum_{i=1}^{N} (\varepsilon_\alpha (k+i)^2 + \varepsilon_{n_z} (k+i)^2) \tag{9.8}$$

subject to the following constraints

$$\alpha(k+i) = G_m(q)\bar{\alpha}_{cmd}(k+i) \tag{9.9a}$$
$$n_z(k+i) = G_m(q)\bar{n}_{z,cmd}(k+i) \tag{9.9b}$$
$$\bar{\alpha}_{cmd}(k+i) = K_{\alpha/n_z}(\gamma^i\mu(k) + \nu(k)) + \alpha_{trim} \tag{9.9c}$$
$$\bar{n}_{z,cmd}(k+i) = (\gamma^i\mu(k) + \nu(k)) + \cos\theta \tag{9.9d}$$
$$\alpha_{min} - \varepsilon_\alpha(k+i) \le \alpha(k+i) + \hat{d}_\alpha \le \alpha_{max} + \varepsilon_\alpha(k+i) \tag{9.9e}$$
$$n_{z,min} - \varepsilon_{n_z}(k+i) \le n_z(k+i) + \hat{d}_{n_z} \le n_{z,max} + \varepsilon_{n_z}(k+i) \tag{9.9f}$$
$$\varepsilon_\alpha(k+i) \ge 0 \tag{9.9g}$$
$$\varepsilon_{n_z}(k+i) \ge 0 \tag{9.9h}$$

where the constraints are for $i = 1, \ldots, N$.

We have used the Euler forward discretization for the response model, $G_m(q)$, which gives the following relation between the time steps of the predicted outputs

$$\alpha(k+1) = a\alpha(k) + b\alpha(k-1) + c\bar{\alpha}_{cmd}(k-1)$$

with $a = 2(1 - T_s\zeta\omega_0)$, $b = 2\zeta\omega_0 T_s - 1 - w_0^2 T_s^2$, and $c = w_0^2 T_s^2$, and similar for the load factor model. To simplify the implementation we select the same sampling time for the command governor as for the nominal controller, $T_s = 1/60$ s. The input to the command governor is the measured angle of attack and load factor at the current time step and the previous time step as well as the current delta load factor command.

The selection of prediction horizon is a tradeoff between performance of the command governor and size of optimization problem to be solved. The rise time for the step response of the nominal controller, in well tuned design points, is approximately 2 seconds and it would be reasonable to include at least half of that, i.e., $N = 50$ samples, in the prediction horizon. However, from simulations we experienced no noticeable increase in performance of the command governor for prediction horizons above approximately 40 samples and hence that was chosen as prediction horizon.

Several different objective function penalties were tested. The $\beta$-penalties were changed with a factor of ten from small penalties up to penalties larger than the slack penalty. In general we also wanted a larger penalty on the deviation from the pilots reference input, i.e., $\beta_2$, than on the transient behavior of the reference, i.e., $\beta_1$. In Figure 9.6 the load factor step response of the closed loop system is shown for several different $\beta$-tunings. From this we can conclude that too small penalties will cause a very bad response. This is due to the shortcomings in the prediction model combined with an excessive penalty on the constraint violations. If the penalties are selected larger than one, i.e., larger than the slack penalty, the command governor will not have any measurable effect on the closed loop system. The ideal penalties from the

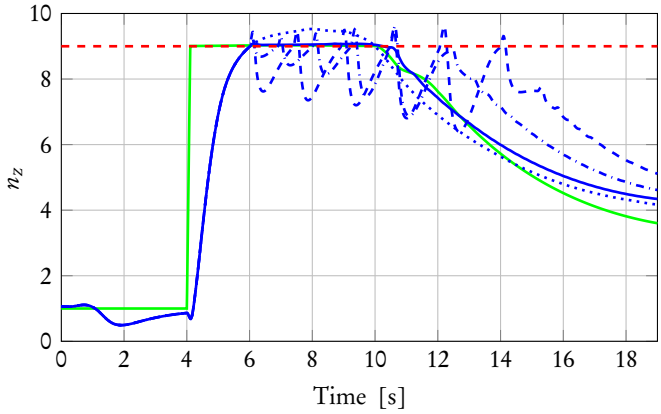


**Figure 9.6.** The normal load factor response of the closed loop system for different penalty combinations in the command governor objective function. The solid blue line is the penalties $\beta_1 = 0.01$, $\beta_2 = 0.1$. The dashed and dash-dotted lines are $\beta_1 = 0.001$, $\beta_2 = 0.01$ and $\beta_1 = 0.01$, $\beta_2 = 0.001$ respectively and the dotted line are the tuning with $\beta_1 = 10$, $\beta_2 = 100$.

simulations have been selected as

$$\beta_1 = 0.01, \quad \beta_2 = 0.1$$

The formulation (9.8) and (9.9) is a standard quadratic program and it has been implemented using the QP solver qpOASES Ferreau et al. [2014]. Extensive research has been performed to develop fast, real-time solvers for quadratic programs Jerez et al. [2014], Jones et al. [2012], Richter et al. [2012] enabling real-time implementation in aircraft computer systems. Additionally if we take a closer look at the problem (9.8) and (9.9) we can see that it in fact only has the two free variables, $\mu$ and $\nu$, to optimize. These two implicitly give all other variables. This mean that we can perform a simple bisection search in the two variables to solve the problem in micro seconds or solve the problem parametrically off-line and make an explicit implementation of the command governor.

To illustrate the achieved performance of the command governor implementation in ARES we have made a series of bleed off turns at different Mach and altitude points. If the maneuver is initiated at speeds above corner speed then the load factor limit will be the most limiting constraint and as the speed reduces the angle of attack limit will become the active constraint.

In Figure 9.7 we have plotted the angle of attack and load factor responses from a bleed off turn initiated at Mach 0.75 and altitude 1000 m.

The upper figure shows the load factor response and the load factor command with and without the command governor. The lower figure shows the angle of attack response and the angle of attack commands. The green lines are

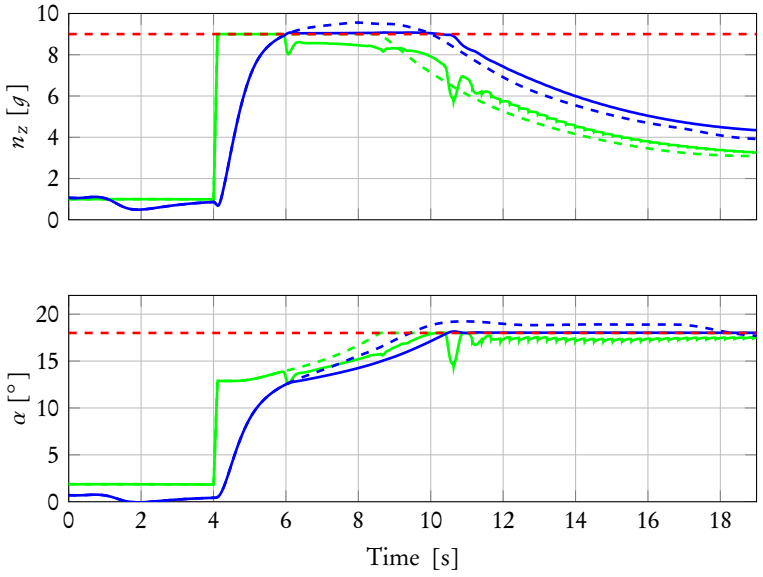


**Figure 9.7.** An ARES simulation of a bleed off turn from Mach 0.75 and altitude 1000 m. Green lines are commanded angle of attack and load factor, blue lines are the aircraft response and the red lines are the corresponding limits. Dashed lines are response without command governor and solid lines are the response with the command governor.

the commanded angle of attack and load factor, the blue lines are the actual aircraft response and the red dashed lines are the load factor and angle of attack limits. The dashed lines are the command and response without the command governor and the solid lines are the responses with the command governor. In this simulation the command governor achieves a distinct reduction in the maneuver limit overshoot, from about 0.6 g to 0.1 g overshoot in load factor and 1.2 degrees to 0.2 degrees in angle of attack.

We can also see that at the beginning of the maneuver the command governor does not interfere with the pilots command, but as soon as the command governor predicts an overshoot of the maneuver limit it modifies the commanded load factor change, $\Delta n_{z,cmd}$, which is used to calculate the commands, $n_{z,cmd}$ and $\alpha_{cmd}$. By more closely examining the figures one can see that the command governor reacts fairly late, approximately 0.1 second before the overshoot, compared to the prediction horizon, which is 0.67 second long. We have tried several different ways to have the command governor react earlier but without success. Our conclusion is, as discussed in the previous section, that this is a result of imperfect model knowledge of the closed loop system.

One other drawback is that when the angle of attack is at the limit value, i.e., from 11 seconds and onwards, there are small oscillations in the calculated command. Since this is not present at other envelope points such as in

Figure 9.8 and 9.9, this is most likely an affect of improper tuning.

At lower speeds, when the nominal maneuver limit overshoot is even bigger, the short reaction time of the command governor result in quite aggressive reference adjustment, see Figure 9.8. This aggressive adjustment causes a small oscillation in the angle of attack response. This is most likely a result of the tuning of the command governor, which is constant throughout the flight envelope. In a final implementation it is suggested to schedule the objective function penalties, $\beta_i$, as a function of speed and altitude, just as is done in the design of the nominal controller.

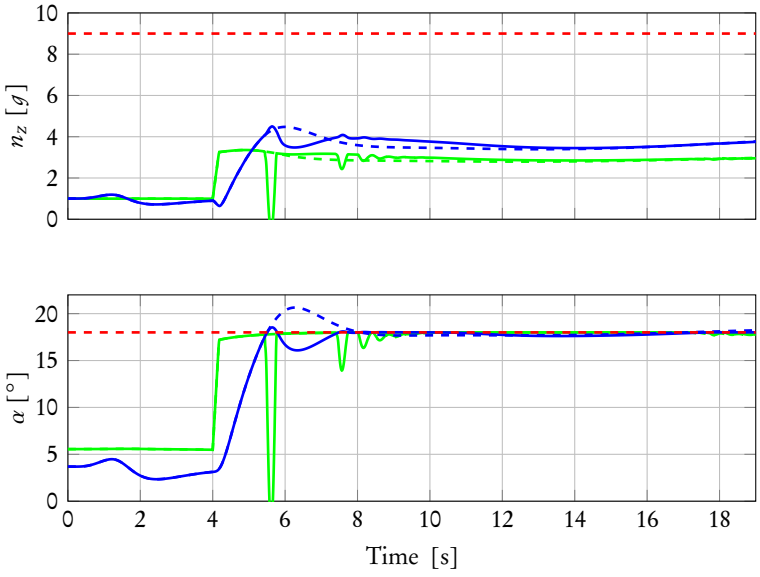


**Figure 9.8.** An ARES simulation of a bleed off turn from Mach 0.4 and altitude 1000 m. Green lines are commanded angle of attack and load factor, blue lines are the aircraft response and the red lines are the corresponding limits. Dashed lines are response without command governor and solid lines are the response with the command governor.

It should be noted here that the two illustrated maneuvers in figures 9.7 and 9.8 are initiated at speeds and altitude points that are not design points of the nominal controller. This means that the true closed loop response might not be as close to the desired closed loop response as it is if the maneuvers were initiated closer to the design points. Also the speed retardation throughout the maneuver will affect the model error. This means that the angle of attack and load factor limit overshoot will be bigger here and at the same time the command governor will have more difficulties predicting it, i.e., we are studying the hard cases here.

If we instead investigate the response starting from on of the envelope points

where the nominal controller has been tuned we can see that the command governor makes only minimal adjustments to the reference command and there is in principle no difference between the responses, see Figure 9.9. In this case the dynamics probably does not change that much during the transient of the step response and hence the closed loop dynamics are more similar to the desired dynamics.

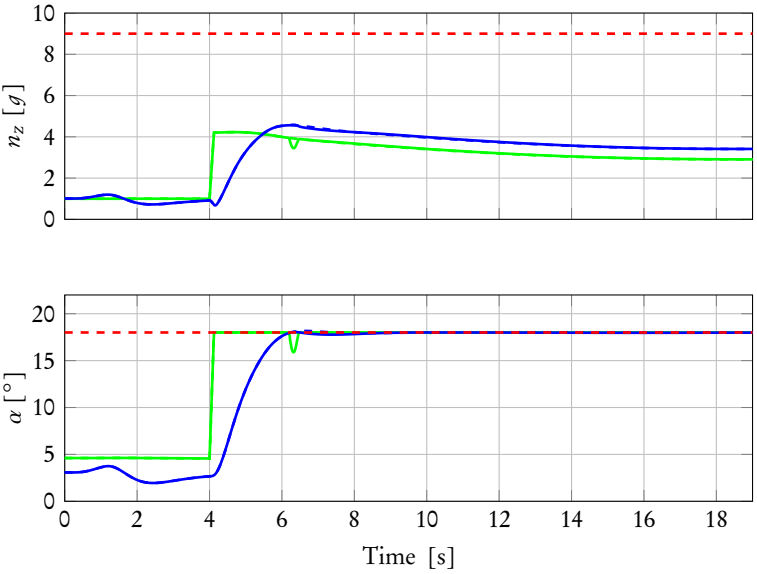


**Figure 9.9.** An ARES simulation of a bleed off turn from Mach 0.6 and altitude 6000 m. Green lines are commanded angle of attack and load factor, blue lines are the aircraft response and the red lines are the corresponding limits. Dashed lines are response without command governor and solid lines are the response with the command governor.

In conclusion we can say that the feature that the command governor can be used as add-on functionality to existing control system architecture makes it an attractive choice for future industrial applications. However, in spite of the attractive add-on feature of the command governor, the simple structure and tuning that are presented in literature is not as straightforward in a real application. In fact if we compare the design and tuning of the command governor to the MPC controller developed in Chapter 8 it is the authors opinion that the MPC controller in fact is easier to tune.

The implemented command governor gives a significant reduction in the angle of attack and load factor limit overshoot in most cases. It improves the design in envelope points where the nominal controller due to model errors or tuning does not satisfy the output constraints. However in these areas the used prediction model is not as accurate as desired. Methods to overcome the model errors in the prediction model showed to be insufficient. A more

advanced scheme for estimating the model error that has a better potential of capturing the relevant dynamics should be developed before putting the command governor into production use.

# 10

# Conclusions and future work

The overall objective of this thesis has been to investigate Model Predictive Control and its applicability to the aeronautical industry and especially to the task of achieving carefree maneuvering in fighter aircraft. We have investigated both theoretical aspects as guaranteed stability for reference tracking in linear MPC and for nonlinear MPC and practical aspects as controller structure and tuning.

In Chapter 5 we studied the topic of reference tracking. Although reference tracking is a relatively mature topic in MPC, very little has been published regarding stability properties when the terminal constraint set depends on the current reference, especially stability when the reference approaches the border of the feasible set. The successful research that has been performed within this area is based on lifting the problem of calculating the terminal constraint set into a higher dimension in which the terminal constraint set is constant for an arbitrary reference. The main drawback with these algorithms is increased complexity of the resulting optimization problem.

In this thesis we instead made a simple extension to the standard dual mode MPC algorithm to allow for tracking arbitrary setpoints that approaches the boundary of the feasible set. This allows for, e.g., maneuver load limiting in fighter aircraft design. We proved that by translating and scaling the terminal state constraint set with a positive scalar $\lambda$ such that it is a subset of the feasible set for arbitrary references, both stability and recursive feasibility can be guaranteed. The performance and robustness of the proposed algorithm is comparable to the existing methods while the complexity of the proposed controller can be significantly reduced. The complexity reduction partly comes from the fact that we utilize duality theory to rewrite a complex set of

constraints into a much simpler form and partly from the fact that the terminal constraint set is in a lower dimension.

In Chapter 6 we addressed the problem of nonlinear aircraft dynamics while trying to avoid complicated nonconvex optimization. We developed a new method of model predictive control for nonlinear systems based on feedback linearization and local convex approximations of the control constraints. We have shown that the proposed method can guarantee recursive feasibility and convergence to the origin. An example from the aircraft industry shows that good performance can be attained and that the loss of optimality can be small with reasonably simple computational efforts.

The methods described in Chapter 5 and 6 both utilize the standard approach to prove recursive feasibility and stability as described in Chapter 4.2.1. However there are many MPC formulations where this standard approach does not work. In Chapter 7 we stated a problem formulation giving a sufficient condition for stability and developed an algorithm for testing stability for linear MPC formulations. The approach is based on formulating an indefinite bilevel programming problem in which the MPC optimization problem is a constraint. We then use the KKT conditions of the MPC optimization problem and big-M reformulation to rewrite the indefinite bilevel problem into a mixed integer linear program. The main benefits of this approach is that we can verify stability of MPC formulations for which the standard approach of proving stability fails and doing so at a reasonable computational time. It is also fairly straightforward to extend the stability test to also test for robust stability, which we also showed in Section 7.3 for additive set bounded disturbances.

The main drawback of the approach is that it relies on the big-M formulation and it can be difficult to correctly select the big M parameters. If they are selected poorly the method will not perform well and can even fail.

The second part of the thesis is devoted to more application-oriented work. Here we have implemented an MPC controller and a command governor in the main flight mechanical simulation environment (ARES) at Saab Aeronautics.

In Chapter 8 we implemented a basic gain scheduled linear MPC controller and investigated the performance that can be achieved for an advanced fighter aircraft. The objective was to implement maneuver limitation of the angle of attack for a fighter aircraft using an MPC controller and to investigate a suitable structure and tuning methodology.

The proposed MPC controller used a lower internal prediction time step than the actual online closed loop update frequency. This can be viewed as an engineering approach to move blocking. We have used Saab's tuning methodology for LQ controllers and applied that for the tuning of the MPC controller. In order to achieve the same linear performance of the MPC controller and the LQ tuning when the MPC controller has a different internal prediction frequency we needed to recalculate equivalent penalty matrices for the MPC problem from the LQ penalty matrices and showed that this can be

done using convex optimization.

The achieved performance of the MPC controller is very good, we recover the performance of the LQ design when no constraints are active and the MPC controller manages to limit the maximum angle of attack within specified maneuver limits. The realtime simulations in Saab's flight mechanical simulator show promising results for future development of MPC controllers for advanced fighter aircraft.

Since much of the flight control design methodology works in an iterative manner we also wanted to investigate the possibility of an add-on feature for the maneuver limiting task. For this purpose we investigated in Chapter 9 command governors as an alternative for maneuver limiting. We implemented an MPC inspired command governor in the ARES simulation environment and investigated different structures and tuning aspects. Although the command governor is a simpler form of controller and appears to be simpler to tune than an MPC controller it is our conclusion that this was not the case, in fact adopting the LQ tuning methodology for the MPC controller makes that much simpler to tune and the performance of the MPC controller is superior to the command governor. The main difficulties for the command governor was to accurately enough predict the response of the aircraft to have a comparable performance to the more advanced MPC structures.

There are still numerous of things that are open topics for research that are important for the use of MPC in aeronautical applications. These concerns areas such as robustness to model errors, stability of gain scheduled MPC controllers and realtime implementations.

One research direction that we find interesting is to continue on the stability test outlined in Chapter 7. Interesting topics to investigate are, e.g., adapting the test for robustness against parametric model errors and testing for stability of gain scheduled MPC controllers. Also if one can reduce the sensitiveness of the approach to the selection of big-M parameters this would be a big step forward. It can also be interesting to look at runtime assurance algorithms for MPC controllers as a complement to the stability test.

In this thesis we have used off-the-shelf QP solvers for the implementation of the MPC controllers. We therefore think it is important to investigate implementation aspects such as custom-made QP solvers and realtime requirements. For example, how will performance and stability be affected if the solver is forced to terminate before the optimum is found?

Looking at the combined control of the pitch and lateral dynamics and the interconnection between them is also an interesting future research topic. How should the structure of the controller be and how should it be tuned to achieve the desired characteristics? Also, investigating further how to adapt and combine the MPC controller methodology with industry know-how and design features as described in Section 3.3.

# Bibliography


Nur Syazreen Ahmad, Joaquin Carrasco, and William Heath. A Less Conservative LMI Condition for Stability of Discrete-Time Systems with Slope-Restricted Nonlinearities. *IEEE Transactions on Automatic Control*, PP(99):1–1, 2014. Cited on page 112.

Johan Åkesson and Per Hagander. Integral Action - a Disturbance Observer Approach. In *Proceedings of the 2003 European Control Conference*, 2003. Cited on page 56.

MOSEK ApS. The MOSEK optimization toolbox for MATLAB manual. Version 7.1 (Revision 28)., 2015. Cited on page 119.

Karl Johan; Åström and Björn Wittenmark. *Computer Control System Theory and Design*. Number 3. Prentice Hall, third edition, 1996. ISBN 978-0133148992. Cited on page 137.

David Avis and Komei Fukuda. A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. *Discrete & Computational Geometry*, 8(1):295–313, 1992. Cited on page 24.

Gary J Balas. Flight Control Law Design: An Industry Perspective. *European Journal of Control*, 2003(9):207–209, 2003. Cited on page 37.

Mato Baotic. *Optimal Control of Piecewise Affine Systems – a Multi-parametric Approach* –. Dissertation, Swiss Federal Institute of Technology, 2005. Cited on pages 23 and 64.

Alberto Bemporad, Alessandro Casavola, and Edoardo Mosca. Nonlinear control of constrained linear systems via predictive reference management. *Automatic Control, IEEE Transactions on*, 42(3):340–349, 1997. Cited on pages 150, 155, and 156.

Alberto Bemporad, Francesco Borrelli, and Manfred Morari. Model predictive control based on linear programming - the explicit solution. *IEEE Transactions on Automatic Control*, 47(12):1974–1985, dec 2002a. Cited on page 61.

Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, jan 2002b. Cited on pages 61, 63, and 64.

Aharon Ben-Tal and Arkadi Nemirovski. Robust optimization methodology and applications. *Mathematical Programming*, 92(3):453–480, may 2002. Cited on page 76.

Raktim Bhattacharya, Gary J Balas, M. Alpay Kaya, and Andy Packard. Nonlinear Receding Horizon Control of an F-16 Aircraft. *Journal of Guidance, Control, and Dynamics*, 25(5):924–931, sep 2002. Cited on page 3.

F. Blanchini. Set invariance in control. *Automatica*, 35:1747 – 1767, 1999. Cited on pages 50 and 79.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 1 edition, 2004. ISBN 9780521833783 Cited on pages 11, 13, 14, 18, 26, 67, and 76.

R. Cagienard, P. Grieder, E.C. Kerrigan, and M. Morari. Move blocking strategies in receding horizon control. *Journal of Process Control*, 17(6): 563–570, 2007. Cited on pages 112 and 122.

Mark Cannon. Efficient nonlinear model predictive control algorithms. *Annual Reviews in Control*, 28(2):229–237, jan 2004. Cited on page 67.

H. Chen and F. Allgöwer. A Quasi-Infinite Horizon Nonlinear Model Predictive Control Scheme with Guaranteed Stability. *Automatica*, 34(10):1205–1217, 1998. Cited on page 66.

Luigi Chisci and Giovanni Zappa. Dual mode predictive tracking of piecewise constant references for constrained linear systems. *International Journal of Control*, 76(1):61–72, jan 2003. Cited on pages 71, 72, and 112.

G Cornuéjols. Valid inequalities for mixed integer linear programs. *Mathematical Programming*, 112(1):3–44, 2008. Cited on page 22.

Fabio Andrade de Almeida and D. Leissling. Fault-Tolerant Flight Control System Using Model Predictive Control. In *2009 Brazilian Symposium on Aerospace Eng. & Applications*, 2009. Cited on page 3.

Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry, Algorithms and Applications*. Springer London, third edit edition, 2008. ISBN 9783540779735. Cited on page 23.

N M C de Oliveira and L T Biegler. Constaint handling and stability properties of model predictive control. *AIChE Journal*, 40(7):1138–1155, 1994. Cited on page 60.

Simone L. De Oliveira Kothare and Manfred Morari. Contractive Model Predictive Control for Constrained Nonlinear Systems. *IEEE Transactions on Automatic Control*, 45(6):1053–1071, 2000. Cited on page 66.

Luigi Del Re, Jacques Chapuis, and Vesna Nevistic. Predictive Control with Embedded Feedback Linearization for Bilinear Plants with Input Constraints. In *Proceedings of the 32th IEEE Conference on Decision and Control (CDC)*, pages 2984–2989, 1993. Cited on page 96.

Jiamei Deng, Victor M. Becerra, and Richard Stobart. Input Constraints Handling in an MPC/Feedback Linearization Scheme. *International Journal of Applied Mathematics and Computer Science*, 19(2):219–232, jun 2009. Cited on page 97.

Luca Deori, Simone Garatti, and Maria Prandini. A model predictive control approach to aircraft motion control. In *American Control Conference*, pages 2299–2304, 2015. Cited on page 97.

Vivek Dua and Efstratios N Pistikopoulos. An Algorithm for the Solution of Multiparametric Mixed Integer Linear Programming Problems. *Annals of Operations Research*, 99(i):123–139, 2001. Cited on page 64.

Derek W Ebdon. *Model Predictive Control of Aerospace Systems*. PhD thesis, Air force institute of technology, 1996. Cited on page 3.

Atam Ercan. New Paths Toward Energy-Efficient Buildings. *IEEE Industrial Electronics Magazine*, 10(4):50–66, 2016. Cited on page 97.

Utku Eren, Anna Prach, Başaran Bahadır Koçer, Saša V. Raković, Erdal Kayacan, and Behçet Açıkmeşe. Model Predictive Control in Aerospace Systems: Current State and Opportunities. *Journal of Guidance, Control, and Dynamics*, pages 1–25, 2017. Cited on page 3.

Wouter Falkena, Clark Borst, Q.P. Chu, and J.A. Mulder. Investigation of Practical Flight Envelope Protection Systems for Small Aircraft. *Journal of Guidance, Control, and Dynamics*, 34(4):976–988, jul 2011. Cited on pages 3 and 131.

Domenico Famularo, Davide Martino, and Massimiliano Mattei. Constrained Control Strategies to Improve Safety and Comfort on Aircraft. *Journal of Guidance, Control, and Dynamics*, 31(6):1782–1792, 2008. Cited on page 148.

A. Ferramosca, Daniel Limon, I. Alvarado, T. Alamo, and E.F. Camacho. MPC for tracking with optimal closed-loop performance. *Automatica*, 45(8): 1975–1978, aug 2009. Cited on pages 71, 79, and 84.

A. Ferramosca, Daniel Limon, I. Alvarado, T. Alamo, F. Castaño, and E.F. Camacho. Optimal MPC for tracking of constrained linear systems. *International Journal of Systems Science*, 42(8):1265–1276, aug 2011. Cited on pages 83 and 84.

Hans Joachim Ferreau, Christian Kirches, Andreas Potschka, Hans Georg Bock, and Moritz Diehl. qpOASES: a parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363, dec 2014. Cited on pages 135 and 158.

R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, 2nd edition, 1987. ISBN 9780471494638 Cited on page 60.

Fernando A.C.C. Fontes. A general framework to design stabilizing nonlinear model predictive controllers. *Systems & Control Letters*, 42(2):127–143, feb 2001. Cited on page 67.

Lars Forssell and Ulrik Nilsson. ADMIRE The Aero-Data Model In a Research Environment Version 4.0, Model Description. Technical Report December, Swedish Defence Research Agency, 2005. Cited on pages 4, 40, 106, and 120.

Hiroaki Fukushima, Ryosuke Saito, Fumitoshi Matsuno, Yasushi Hada, Kuniaki Kawabata, and Hajime Asama. Model Predictive Control of an Autonomous Blimp with Input and Output Constraints. In *2006 IEEE International Conference on Control Applications*, pages 2184–2189. IEEE, oct 2006. Cited on page 3.

Tomas Gal. A " Historiogramme " of Parametric Programming. *The Journal of the Operational Research Society*, 31(5):449–451, 1980. Cited on page 61.

B. A. G. Genuit, Liang Lu, and W. P. M. H. Heemels. Approximation of PWA Control Laws Using Regular Partitions : An ISS Approach. In *Proceedings of the 18th IFAC World Congress*, pages 4540–4545, 2011. Cited on page 65.

Peter W Gibbens and Eran D B Medagoda. Efficient Model Predictive Control Algorithm for Aircraft. *Journal of Guidance, Control, and Dynamics*, 34(6): 1909–1915, 2011. doi: 10.2514/1.52162. Cited on page 141.

J C Gibson. *Development of a Methodology for Excellence in Handling Qualities Design for Fly by Wire Aircraft*. Delft University press, 1999. ISBN 9040718415. Cited on page 38.

H.-G. Giesseler, M Kopf, P Varutti, T Faulwasser, and Rolf Findeisen. Model Predictive Control for Gust Load Alleviation. *IFAC Proceedings Volumes*, 45 (17):27–32, 2012. Cited on page 3.

E.G. Gilbert, I. Kolmanovsky, and K.T. Tan. Nonlinear control of discrete-time linear systems with state and control constraints: a reference governor with global convergence properties. *Proceedings of 1994 33rd IEEE Conference on Decision and Control*, 1:144–149, 1994. Cited on page 149.

Elmer G. Gilbert and Chong Jin Ong. Constrained linear systems with hard constraints and disturbances: An extended command governor with large domain of attraction. *Automatica*, 47(2):334–340, 2011. Cited on pages 150, 151, and 155.

Elmer G. Gilbert and Kok Tin Tan. Maximal output admissible sets and the nonlinear control of linear discrete-time systems with state and control constraints. *IEEE Transactions on Automatic Control*, 36(9):1008–1020, 1991. Cited on page 149.

Elmer G Gilbert, Ilya Kolmanovsky, and K.T. Tan. Discrete-time reference governors and the nonlinear control of systems with state and control constraints. *International Journal of Robust and Nonlinear Control*, 5(1995): 487–504, 1995. Cited on pages 149, 150, and 155.

Sebastien Gros, Rien Quirynen, and Moritz Diehl. Aircraft control based on fast non-linear MPC & multiple-shooting. In *51st IEEE Conference on Decision and Control (CDC)*, pages 1142–1147. IEEE, dec 2012. Cited on pages 3, 131, and 132.

Branko Grünbaum. *Convex Polytopes*. Springer-Verlag, 1 edition, 2003. ISBN 0-387-00424-6. Cited on pages 23 and 75.

Kristoffer Gryte and Thor I Fossen. Non-linear Model Predictive Control for Longitudinal and Lateral Guidance of a Small Fixed-Wing UAV in Precision Deep Stall Landing. In *AIAA Guidance, Navigation, and Control Conference*. AIAA Paper 2016-0512, 2016. Cited on page 3.

O Härkegård. *Backstepping and Control Allocation with Applications to Flight Control*. Dissertation, Linköping University, 2003. Cited on page 31.

E N Hartley. Predictive Control with Parameter Adaptation to Achieve $\alpha$ -Protection in the RECONFIGURE Benchmark in the Presence of Icing. *IFAC-PapersOnLine*, 48(314):172–177, 2015. Cited on pages 3, 131, and 132.

E N Hartley and J M Maciejowski. A longitudinal flight control law based on robust MPC and H2 methods to accommodate sensor loss in the RECONFIGURE benchmark. In *9th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, volume 48(21), pages 1000–1005, 2015. Cited on page 3.

E N Hartley, J M Maciejowski, and K-V Ling. Performance evaluation of multiplexed model predictive control for a large airliner in nominal and contingency scenarios. In *2012 American Control Conference (ACC)*, pages 1199–1204. IEEE, jun 2012. Cited on page 141.

Michael A. Henson. Nonlinear model predictive control: current status and future directions. *Computers & Chemical Engineering*, 23(2):187–202, dec 1998. Cited on page 67.

M. Herceg, Michal Kvasnica, Colin N. Jones, and Manfred Morari. Multi-Parametric Toolbox 3.0. In *Proceedings of the 2013 European Control Conference*, pages 502–510, 2013. Cited on pages 84 and 104.

Gurobi Optimization Inc. Gurobi Optimizer reference manual, 2014. URL `www.gurobi.com`. Cited on page 119.

A. Jadbabaie, J. Yu, and J. Hauser. Unconstrained receding-horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 46(5):776–783, may 2001. Cited on page 66.

Juan L Jerez, Paul J Goulart, Stefan Richter, George A Constantinides, Eric C Kerrigan, and Manfred Morari. Embedded Online Optimization for Model Predictive Control at Megahertz Rates. *IEEE Transactions on Automatic Control*, 59(12):3238–3251, 2014. Cited on page 158.

Colin N. Jones, P. Grieder, and S.V. Raković. A logarithmic-time solution to the point location problem for parametric linear programming. *Automatica*, 42(12):2215–2218, dec 2006. Cited on page 64.

Colin N Jones, Alexander Domahidi, Manfred Morari, Stefan Richter, Fabian Ullmann, and Melanie Zeilinger. Fast Predictive Control : Real-time Computation and Certification. In *4th IFAC Nonlinear model predictive control conference*. IFAC, 2012. Cited on page 158.

Uros Kalabic, Ilya Kolmanovsky, Julia Buckland, and Elmer Gilbert. Reference and extended command governors for control of turbocharged gasoline engines based on linear models. In *Proceedings of the IEEE International Conference on Control Applications*, pages 319–325, 2011. Cited on pages 150 and 151.

Uros Kalabic, Yash Chitalia, Julia Buckland, and Ilya Kolmanovsky. Prioritization Schemes for Reference and Command Governors. In *European Control Conference*, pages 2734–2739, 2013. Cited on page 149.

M.M. Kale and A.J. Chipperfield. Stabilized MPC formulations for robust reconfigurable flight control. *Control Engineering Practice*, 13(6):771–788, jun 2005. Cited on pages 3 and 4.

R E Kalman. Contributions to the Theory of Optimal Control. *Boletin de la Sociedad Matematica Mexicana*, 5:102–119, 1960. Cited on page 2.

S. S. Keerthi and Elmer G. Gilbert. Optimal Infinite-Horizon Feedback Laws for a General Class of Constrained Discrete-Time Systems : Stability and Moving-Horizon Approximations. *Journal of Optimization Theory and Applications*, 57(2):265–293, 1988. Cited on page 49.

Eric C. Kerrigan and Jan M Maciejowski. Soft Constraints and Exact Penalty Functions in Model Predictive Control. In *Proceedings of the UKACC International Conference*, 2000. Cited on page 60.

Tamás Keviczky and Gary J Balas. Flight test of a receding horizon controller for autonomous UAV guidance. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 3518–3523. IEEE, 2005. Cited on page 3.

Tamás Keviczky and Gary J Balas. Receding horizon control of an F-16 aircraft: A comparative study. *Control Engineering Practice*, 14(9):1023–1033, sep 2006a. Cited on pages 3, 4, and 132.

Tamas Keviczky and Gary J Balas. Software-Enabled Receding Horizon Control for Autonomous Unmanned Aerial Vehicle Guidance. *Journal of Guidance, Control, and Dynamics*, 29(3):680–694, 2006b. Cited on page 3.

Hassan K. Khalil. *Nonlinear Systems third edition*. Prentice Hall, 2002. ISBN 0-13-067389-7. Cited on pages 48, 96, and 98.

Ilya Kolmanovsky and Elmer G. Gilbert. Theory and computation of disturbance invariant sets for discrete-time linear systems. *Mathematical Problems in Engineering*, 4(4):317–367, 1998. Cited on pages 123, 150, and 151.

Ilya Kolmanovsky and Nazli E Kahveci. Constrained Control of UAVs Using Adaptive Anti-windup Compensation and Reference Governors. In *Proceedings of SAE AeroTech Congress and Exhibition*, 2009. Cited on page 148.

Ilya Kolmanovsky, Emanuele Garone, and Stefano Di Cairano. Reference and command governors: A tutorial on their theory and automotive applications. *2014 American Control Conference*, pages 226–241, jun 2014. Cited on pages 151 and 153.

Milan Korda and Colin N. Jones. Stability and Performance Verification of Optimization-based Controllers. 2015. URL `http://arxiv.org/abs/1501.03919`. Cited on page 112.

Mayuresh V. Kothare, Vesna Nevistic, and Manfred Morari. Robust Constrained Model Predictive Control for nonlinear Systems: A Comparative Study. In *Proceedings of the 34th IEEE Conference on Decision and Control (CDC)*, number December, pages 2884–2885, 1995. Cited on page 97.

D K Kufoalor and T A Johansen. Reconfigurable fault tolerant flight control based on Nonlinear Model Predictive Control. In *2013 American Control Conference*, pages 5128–5133. IEEE, jun 2013. Cited on pages 3 and 4.

Michael J Kurtz and Michael A Henson. Feedback linearizing control of discrete-time nonlinear systems with input constraints. *International Journal of Control*, 70:4:603–616, 2010. Cited on page 97.

Michal Kvasnica. *Real-Time Model Predictive Control via Multi-Parametric Programming: Theory and Tools*. VDM Verlag, 2009. ISBN 9783639206449. Cited on page 64.

Jean B Lasserre. Global Optimization with Polynomials and the Problem of Moments. *SIAM Journal on Optimization*, 11:796–817, 2001. Cited on page 20.

Jay H Lee and Brian Cooley. Recent advances in model predictive control and other related areas. In *5th International conference on chemical process control*, pages 201–216, 1997. Cited on page 54.

Barry Lennox, W.P. Heath, and Guang Li. Concise stability conditions for systems with static nonlinear feedback expressed by a quadratic program. *IET Control Theory & Applications*, 2(7):554–563, jul 2008. Cited on pages 112, 113, and 121.

Guang Li, William P. Heath, and Barry Lennox. The stability analysis of systems with nonlinear feedback expressed by a quadratic program. *Proceedings*

*of the 45th IEEE Conference on Decision and Control*, 2(3):4247–4252, 2006. Cited on page 112.

Qinghua Li, Uros V. Kalabic, and Ilya V. Kolmanovsky. Fast reference governors for second-order linear systems with constraints and an input time-delay. *Automatica*, 50(2):641–645, 2014. Cited on pages 149 and 152.

Daniel Limon, I. Alvarado, T. Alamo, and E.F. Camacho. MPC for tracking piecewise constant references for constrained linear systems. *Automatica*, 44 (9):2382–2387, sep 2008. Cited on pages 71, 72, 84, 86, 91, and 112.

Johan Löfberg. *Minimax approaches to robust model predictive control*. Dissertation, Linköping University, 2003. Cited on page 120.

Johan Löfberg. YALMIP: A Toolbox for Modeling and Optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. Cited on pages 84, 104, and 119.

Johan Löfberg. Oops! I cannot do it again: Testing for recursive feasibility in MPC. *Automatica*, 48(3):550–555, 2012. Cited on pages 114 and 127.

Christian Løvaas, María M. Seron, and Graham C. Goodwin. A dissipativity approach to robustness in constrained model predictive control. *Proceedings of the IEEE Conference on Decision and Control*, 2:1180–1185, 2007. Cited on page 112.

Jan M. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, 1 edition, 2002. ISBN 0-201-39823-0. Cited on pages 2, 60, and 135.

Jan M Maciejowski and Colin N Jones. MPC Fault-tolerant Flight Control Case Study: Flight 1862. In *IFAC Safeprocess Conference*, 2003. Cited on page 3.

Urban Maeder and Manfred Morari. Offset-free reference tracking with model predictive control. *Automatica*, 46(9):1469–1476, sep 2010. Cited on page 55.

L Magni, G De Nicolao, L Magnani, and R Scattolini. A stabilizing model-based predictive control algorithm for nonlinear systems. *Automatica*, 37: 1351–1362, 2001. Cited on page 66.

Hugues Marchand, Alexander Martin, Robert Weismantel, and Laurence Wolsey. Cutting planes in integer and mixed integer programming. *Discrete Applied Mathematics*, 123(1-3):397–446, 2002. Cited on page 22.

Kostas Margellos and John Lygeros. A simulation based MPC technique for feedback linearizable systems with input constraints. In *49th IEEE Conference on Decision and Control (CDC)*, pages 7539–7544. Ieee, dec 2010. Cited on page 97.

Davide Martino. Flight control with amplitude and rate constraints: A command governor approach. In *American Control Conference*, pages 1788–1793, 2008. Cited on page 148.

David Mayne. Nonlinear Model Predictive Control: Challenges and Opportunities. In *Nonlinear model predictive control*, pages 23–44. Birkhäuser Verlag Basel, 2000. Cited on page 67.

D.Q. Mayne and H. Michalska. Receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 35(7):814–824, jul 1990. Cited on page 67.

D.Q. Mayne, James B. Rawlings, Christopher V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, jun 2000. Cited on pages 49, 50, 54, 61, 79, 112, and 121.

Edward S. Meadows and Thomas Badgwell. Feedback through steady-state target optimization for nonlinear model predictive control. *Journal of Vibration and Control*, 4:61–74, 1998. Cited on pages 53, 54, and 55.

B Mettler. *Identification Modeling and Characteristics of Miniature Rotorcraft*. Springer, 2002. ISBN 9781402072284. Cited on page 90.

H. Michalska. A New Formulation of Receding Horizon Stabilising Control without Terminal Constraint on the State. *European Journal of Control*, 3: 2–14, 1997. Cited on page 67.

H. Michalska and D.Q. Mayne. Robust receding horizon control of constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 38(11):1623–1633, 1993. Cited on page 66.

MIL-F-8785C. Flying Qualities of Piloted Airplanes, 1980. Cited on pages 38, 43, and 136.

Manfred Morari and Jay H Lee. Model predictive control: past , present and future. *Computers & Chemical Engineering*, 23:667–682, 1999. Cited on page 67.

Kenneth R. Muske and Thomas Badgwell. Disturbance modeling for offset-free linear model predictive control. *Journal of Process Control*, 12(5):617–632, aug 2002. Cited on page 55.

Kenneth R. Muske and James B. Rawlings. Model predictive control with linear models. *AIChE Journal*, 39(2):262–287, feb 1993. Cited on pages 49, 53, 54, and 55.

Nato. Flight Control Design - Best Practices. Technical report, Laws, Task Group SCI-026 on Flight Control, 2000. Cited on pages 1 and 40.

Robert C. Nelson. *Flight Stability and Automatic Control*. McGraw-Hill, 2 edition, 1998. ISBN 0-07-115838-3. Cited on pages 29 and 34.

Yurii Nesterov and Arkadii Nemirovskii. *Interior-Point polynomial Algorithms in Convex Programming*. SIAM, 1 edition, 1994. Cited on page 76.

Vesna Nevistic and L. Del Re. Feasible Suboptimal Model Predictive Control for Linear Plants with State Dependent Constraints. In *Proceedings of the 1994*

*American Control Conference*, volume 2862, pages 2862–2866, 1994. Cited on page 97.

G De Nicolao, L Magni, and R Scattolini. Stabilizing Nonlinear Receding Horizon Control via a Nonquadratic Terminal State Penalty. In *Proceedings of the symposium on control, optimization and supervision CESA'96*, pages 185–187, 1996. Cited on page 66.

G De Nicolao, L Magni, and R Scattolini. Stabilizing Receding-Horizon Control of Nonlinear Time-Varying Systems. *IEEE Transactions on Automatic Control*, 43(7):1030–1036, 1998. Cited on page 66.

G De Nicolao, L Magni, and R. Scattolini. Stability and robustness of nonlinear receding horizon control. In *Nonlinear model predictive control*, pages 3–22. Birkhäuser Basel, 2000. Cited on page 66.

NLR. Aircraft Accident Report 92-11, El Al Flight 1862. Technical report, Nederlands Aviation Safety Board, 1992. Cited on page 3.

Gabriele Pannocchia and Eric C. Kerrigan. Offset-free control of constrained linear discrete-time systems subject to persistent unmeasured disturbances. In *Proceedings of the 42th IEEE Conference on Decision and Control*, number December, pages 1–6, 2003. Cited on page 55.

Gabriele Pannocchia and James B. Rawlings. Disturbance models for offset-free model-predictive control. *AIChE Journal*, 49(2):426–437, feb 2003. Cited on page 55.

Yash Vardhan Pant, Houssam Abbas, and Rahul Mangharam. Robust Model Predictive Control for Non-Linear Systems with Input and State Constraints Via Feedback Linearization Tech Report : Robust Model Predictive Control for Non-Linear Systems. In *IEEE 55th Conference on Decision and Control*, number December, pages 5694–5699, 2016. Cited on page 97.

Christopher Petersen, Morgan Baldwin, and Ilya Kolmanovsky. Model Predictive Control Guidance with Extended Command Governor Inner-Loop Flight Control for Hypersonic Vehicles. In *AIAA Guidance, Navigation, and Control (GNC) Conference*. AIAA Paper 2013-5028, 2013. Cited on pages 3 and 148.

James A Primbs. The analysis of optimization based controllers. *Automatica*, 37(June):3297–3301, 2001. Cited on pages 112, 113, and 120.

James A Primbs and Vesna Nevistic. MPC Extensions to Feedback Linearizable Systems. In *Proceedings of the 1997 American Control Conference*, number June, pages 2073–2077, 1997. Cited on page 96.

James A Primbs and Vesna Nevistic. A new approach to stability analysis for constrained finite receding horizon control without end constraints. *Ieee Transactions on Automatic Control*, 45(8):1507–1512, 2000. Cited on page 112.

James A Primbs, Vesna Nevistic, and John C Doyle. Nonlinear optimal control:

A control Lyapunov function and receding horizon perspective. *Asian Journal of Control*, 1(1):14–24, 1999. Cited on page 66.

A. I. Propoi. Use of linear programming methods for synthesizing sampled-data automatic systems. *Automation and Remote Control*, 24(7):837–844, 1963. Cited on page 45.

D M Raimondo, O Huber, M Schulze Darup, M Mönningmann, and Manfred Morari. Constrained time-optimal control for nonlinear systems : a fast explicit approximation. In *Proceedings of the 4th IFAC Nonlinear Model Predictive Control Conference*, pages 113–118, 2012. Cited on page 65.

S V Rakovic, E C Kerrigan, K I Kouramas, and D Q Mayne. Invariant Approximation of the Minimal Robust Positively Invariant Set. *IEEE Transactions on Automatic Control*, 50(3):406–410, 2005. Cited on pages 123, 124, and 126.

Christopher V. Rao and James B. Rawlings. Steady states and constraints in model predictive control. *AIChE Journal*, 45(6):1266–1278, jun 1999. Cited on pages 54, 70, and 72.

James B. Rawlings. Tutorial Overview of Model Predictive Control. *IEEE Control Systems Magazine*, 20(3):38–52, 2000. Cited on page 54.

James B. Rawlings and David Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, first edition, 2009. ISBN 978-0-9759377-0-9. Cited on pages 59 and 121.

James B. Rawlings and Kenneth R. Muske. The Stability of Constrained Receding Horizon Control. *IEEE Transactions on Automatic Control*, 38(10): 1512–1516, 1993. Cited on page 49.

James B. Rawlings, Edward S. Meadows, and Kenneth R. Muske. Nonlinear Model Predictive Control: A Tutorial and Survey. In *IFAC Advanced Control of Chemical Processes*, pages 185–197, 1994. Cited on page 54.

Arthur Richards, William Stewart, and Alex Wilkinson. Auto-coding Implementation of Model Predictive Control with Application to Flight Control. *European Control Conference*, pages 150–155, 2009. Cited on page 141.

Stefan Richter, Colin Neil Jones, and Manfred Morari. Computational Complexity Certification for Real-Time MPC With Input Constraints Based on the Fast Gradient Method. *IEEE Transactions on Automatic Control*, 57(6): 1391–1403, 2012. Cited on page 158.

J. A. Rossiter. A Global Approach to Feasibility in Linear MPC. In *UKACC ICC*, 2006. Cited on page 54.

Matteo Rubagotti, Davide Barcelli, and Alberto Bemporad. Approximate Explicit MPC on Simplicial Partitions with Guaranteed Stability for Constrained Linear Systems. In *Proceedings of the 4th IFAC Nonlinear Model Predictive Control Conference*, pages 119–125, 2012. Cited on page 65.

Lars Rundqwist and Karin Ståhl-Gunnarsson. Phase compensation of rate limiters in unstable aircraft. In *IEEE International Conference on Control Applications*, pages 19–24, 1996. Cited on page 39.

P.O.M. Scokaert, D.Q. Mayne, and J.B. Rawlings. Suboptimal model predictive control (feasibility implies stability). *IEEE Transactions on Automatic Control*, 44(3):648–654, mar 1999. Cited on page 67.

C. Shearer and S. Heise. Constrained model predictive control of a nonlinear aerospace system. In *Guidance, Navigation, and Control Conference and Exhibit*. AIAA Paper 98-4235, 1998. Cited on pages 3 and 4.

Bilal Siddiqui. *Reconfigurable flight control for fighter aircraft using MPC*. VDM Verlag, 2010. ISBN 978-3-639-26225-4. Cited on page 3.

Daniel Simon and Johan Löfberg. Stability analysis of Model Predictive Controllers using Mixed Integer Linear Programming. In *IEEE 55th Conference on Decision and Control*, pages 7270–7275, Las Vegas, 2016. Not cited.

Daniel Simon, Johan Löfberg, and Torkel Glad. Reference tracking MPC using terminal set scaling. In *51st IEEE Conference on Decision and Control (CDC)*, pages 4543–4548, dec 2012. Not cited.

Daniel Simon, Johan Löfberg, and Torkel Glad. Nonlinear Model Predictive Control using Feedback Linearization and Local Inner Convex Constraint Approximations. In *Proceedings of the 2013 European Control Conference*, pages 2056–2061, 2013. Not cited.

Daniel Simon, Johan Löfberg, and Torkel Glad. Reference Tracking MPC using Dynamic Terminal Set Transformation. *IEEE Transactions on Automatic Control*, 59(10):2790–2795, 2014. Not cited.

Daniel Simon, Ola Härkegård, and Johan Löfberg. Angle of Attack and Load Factor Limiting in Fighter Aircraft using Command Governors. In *AIAA Guidance, Navigation, and Control Conference, AIAA SciTech Forum*, 2017a. Not cited.

Daniel Simon, Ola Härkegård, and Johan Löfberg. Command Governor Approach to Maneuver Limiting in Fighter Aircraft. *Journal of Guidance, Control, and Dynamics*, 40(6):1514–1527, 2017b. Not cited.

Sigurd Skogestad and Ian Postlethwaite. *Multivariable Feedback Control Analysis and Design*. John Wiley & Sons, 2 edition, 2005. ISBN 978-0-470-01168-3. Cited on page 47.

Marc Steinberg. A comparison of intelligent, adaptive, and nonlinear flight control laws. In *Guidance, Navigation, and Control Conference and Exhibit*, page 11. AIAA Paper 99-4044, 1999. Cited on page 3.

Brian L. Stevens and Frank L. Lewis. *Aircraft Control and Simulation*. John Wiley & Sons, 2 edition, 2003. ISBN 0-471-37145-9. Cited on pages 29, 35, and 37.

Mario Sznaier and Mark J Damborg. Suboptimal control of linear systems With state and control inequality constraints. In *Proceedings of the 26th IEEE Conference on Decision and Control (CDC)*, pages 761–762, 1987. Cited on pages 51 and 83.

Yang Wang and Stephen Boyd. Fast Model Predictive Control Using Online Optimization. *Control Systems Technology, IEEE Transactions on*, 18(2):267–278, 2010. Cited on page 141.

Yinan Wang, Andrew Wynn, and Rafael Palacios. Model-Predictive Control of Flexible Aircraft Dynamics using Nonlinear Reduced-Order Models. In *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, number January, pages 1–11, 2016. Cited on page 3.

Laurence A. Wolsey. *Integer Programming*. Wiley-interscience, 1998. ISBN 0-471-28366-5. Cited on pages 20 and 22.

Zhi-Jun Yang, Xiao-Hui Qi, and Gan-Lin Shan. Simulation of Flight Control Laws Design Using Model Predictive Controllers. In *IEEE International Conference on Mechatronics and Automation*, pages 4213–4218, 2009. Cited on page 3.

Hui Ye, Mou Chen, and Qingxian Wu. Flight Envelope Protection Control Based on Reference Governor Method in High Angle of Attack Maneuver. *Mathematical Problems in Engineering*, 2015:1–15, 2015. Cited on pages 148, 149, and 152.

Victor M. Zavala and Lorenz T. Biegler. The advanced-step NMPC controller: Optimality, stability and robustness. *Automatica*, 45(1):86–93, 2009. Cited on page 141.

Melanie N Zeilinger, Manfred Morari, and Colin N Jones. Soft Constrained Model Predictive Control With Robust Stability Guarantees. *IEEE Transactions on Automatic Control*, 59(5):1190–1202, 2014. Cited on page 112.

Alicia Zinnecker, Andrea Serrani, Michael Bolender, and David Doman. Combined Reference Governor and Anti-Windup Design for Constrained Hypersonic Vehicles Models. In *AIAA Guidance, Navigation, and Control Conference*. AIAA Paper 2009-6283, 2009. Cited on page 148.

**PhD Dissertations**
**Division of Automatic Control**
**Linköping University**

**M. Millnert:** Identification and control of systems subject to abrupt changes. Thesis No. 82, 1982. ISBN 91-7372-542-0.

**A. J. M. van Overbeek:** On-line structure selection for the identification of multivariable systems. Thesis No. 86, 1982. ISBN 91-7372-586-2.

**B. Bengtsson:** On some control problems for queues. Thesis No. 87, 1982. ISBN 91-7372-593-5.

**S. Ljung:** Fast algorithms for integral equations and least squares identification problems. Thesis No. 93, 1983. ISBN 91-7372-641-9.

**H. Jonson:** A Newton method for solving non-linear optimal control problems with general constraints. Thesis No. 104, 1983. ISBN 91-7372-718-0.

**E. Trulsson:** Adaptive control based on explicit criterion minimization. Thesis No. 106, 1983. ISBN 91-7372-728-8.

**K. Nordström:** Uncertainty, robustness and sensitivity reduction in the design of single input control systems. Thesis No. 162, 1987. ISBN 91-7870-170-8.

**B. Wahlberg:** On the identification and approximation of linear systems. Thesis No. 163, 1987. ISBN 91-7870-175-9.

**S. Gunnarsson:** Frequency domain aspects of modeling and control in adaptive systems. Thesis No. 194, 1988. ISBN 91-7870-380-8.

**A. Isaksson:** On system identification in one and two dimensions with signal processing applications. Thesis No. 196, 1988. ISBN 91-7870-383-2.

**M. Viberg:** Subspace fitting concepts in sensor array processing. Thesis No. 217, 1989. ISBN 91-7870-529-0.

**K. Forsman:** Constructive commutative algebra in nonlinear control theory. Thesis No. 261, 1991. ISBN 91-7870-827-3.

**F. Gustafsson:** Estimation of discrete parameters in linear systems. Thesis No. 271, 1992. ISBN 91-7870-876-1.

**P. Nagy:** Tools for knowledge-based signal processing with applications to system identification. Thesis No. 280, 1992. ISBN 91-7870-962-8.

**T. Svensson:** Mathematical tools and software for analysis and design of nonlinear control systems. Thesis No. 285, 1992. ISBN 91-7870-989-X.

**S. Andersson:** On dimension reduction in sensor array signal processing. Thesis No. 290, 1992. ISBN 91-7871-015-4.

**H. Hjalmarsson:** Aspects on incomplete modeling in system identification. Thesis No. 298, 1993. ISBN 91-7871-070-7.

**I. Klein:** Automatic synthesis of sequential control schemes. Thesis No. 305, 1993. ISBN 91-7871-090-1.

**J.-E. Strömberg:** A mode switching modelling philosophy. Thesis No. 353, 1994. ISBN 91-7871-430-3.

**K. Wang Chen:** Transformation and symbolic calculations in filtering and control. Thesis No. 361, 1994. ISBN 91-7871-467-2.

**T. McKelvey:** Identification of state-space models from time and frequency data. Thesis No. 380, 1995. ISBN 91-7871-531-8.

**J. Sjöberg:** Non-linear system identification with neural networks. Thesis No. 381, 1995. ISBN 91-7871-534-2.

**R. Germundsson:** Symbolic systems – theory, computation and applications. Thesis No. 389, 1995. ISBN 91-7871-578-4.

**P. Pucar:** Modeling and segmentation using multiple models. Thesis No. 405, 1995. ISBN 91-7871-627-6.

**H. Fortell:** Algebraic approaches to normal forms and zero dynamics. Thesis No. 407, 1995. ISBN 91-7871-629-2.

**A. Helmersson:** Methods for robust gain scheduling. Thesis No. 406, 1995. ISBN 91-7871-628-4.

**P. Lindskog:** Methods, algorithms and tools for system identification based on prior knowledge. Thesis No. 436, 1996. ISBN 91-7871-424-8.

**J. Gunnarsson:** Symbolic methods and tools for discrete event dynamic systems. Thesis No. 477, 1997. ISBN 91-7871-917-8.

**M. Jirstrand:** Constructive methods for inequality constraints in control. Thesis No. 527, 1998. ISBN 91-7219-187-2.

**U. Forssell:** Closed-loop identification: Methods, theory, and applications. Thesis No. 566, 1999. ISBN 91-7219-432-4.

**A. Stenman:** Model on demand: Algorithms, analysis and applications. Thesis No. 571, 1999. ISBN 91-7219-450-2.

**N. Bergman:** Recursive Bayesian estimation: Navigation and tracking applications. Thesis No. 579, 1999. ISBN 91-7219-473-1.

**K. Edström:** Switched bond graphs: Simulation and analysis. Thesis No. 586, 1999. ISBN 91-7219-493-6.

**M. Larsson:** Behavioral and structural model based approaches to discrete diagnosis. Thesis No. 608, 1999. ISBN 91-7219-615-5.

**F. Gunnarsson:** Power control in cellular radio systems: Analysis, design and estimation. Thesis No. 623, 2000. ISBN 91-7219-689-0.

**V. Einarsson:** Model checking methods for mode switching systems. Thesis No. 652, 2000. ISBN 91-7219-836-2.

**M. Norrlöf:** Iterative learning control: Analysis, design, and experiments. Thesis No. 653, 2000. ISBN 91-7219-837-0.

**F. Tjärnström:** Variance expressions and model reduction in system identification. Thesis No. 730, 2002. ISBN 91-7373-253-2.

**J. Löfberg:** Minimax approaches to robust model predictive control. Thesis No. 812, 2003. ISBN 91-7373-622-8.

**J. Roll:** Local and piecewise affine approaches to system identification. Thesis No. 802, 2003. ISBN 91-7373-608-2.

**J. Elbornsson:** Analysis, estimation and compensation of mismatch effects in A/D converters. Thesis No. 811, 2003. ISBN 91-7373-621-X.

**O. Härkegård:** Backstepping and control allocation with applications to flight control. Thesis No. 820, 2003. ISBN 91-7373-647-3.

**R. Wallin:** Optimization algorithms for system analysis and identification. Thesis No. 919, 2004. ISBN 91-85297-19-4.

**D. Lindgren:** Projection methods for classification and identification. Thesis No. 915, 2005. ISBN 91-85297-06-2.

**R. Karlsson:** Particle Filtering for Positioning and Tracking Applications. Thesis No. 924, 2005. ISBN 91-85297-34-8.

**J. Jansson:** Collision Avoidance Theory with Applications to Automotive Collision Mitigation. Thesis No. 950, 2005. ISBN 91-85299-45-6.

**E. Geijer Lundin:** Uplink Load in CDMA Cellular Radio Systems. Thesis No. 977, 2005. ISBN 91-85457-49-3.

**M. Enqvist:** Linear Models of Nonlinear Systems. Thesis No. 985, 2005. ISBN 91-85457-64-7.

**T. B. Schön:** Estimation of Nonlinear Dynamic Systems — Theory and Applications. Thesis No. 998, 2006. ISBN 91-85497-03-7.

**I. Lind:** Regressor and Structure Selection — Uses of ANOVA in System Identification. Thesis No. 1012, 2006. ISBN 91-85523-98-4.

**J. Gillberg:** Frequency Domain Identification of Continuous-Time Systems Reconstruction and Robustness. Thesis No. 1031, 2006. ISBN 91-85523-34-8.

**M. Gerdin:** Identification and Estimation for Models Described by Differential-Algebraic Equations. Thesis No. 1046, 2006. ISBN 91-85643-87-4.

**C. Grönwall:** Ground Object Recognition using Laser Radar Data – Geometric Fitting, Performance Analysis, and Applications. Thesis No. 1055, 2006. ISBN 91-85643-53-X.

**A. Eidehall:** Tracking and threat assessment for automotive collision avoidance. Thesis No. 1066, 2007. ISBN 91-85643-10-6.

**F. Eng:** Non-Uniform Sampling in Statistical Signal Processing. Thesis No. 1082, 2007. ISBN 978-91-85715-49-7.

**E. Wernholt:** Multivariable Frequency-Domain Identification of Industrial Robots. Thesis No. 1138, 2007. ISBN 978-91-85895-72-4.

**D. Axehill:** Integer Quadratic Programming for Control and Communication. Thesis No. 1158, 2008. ISBN 978-91-85523-03-0.

**G. Hendeby:** Performance and Implementation Aspects of Nonlinear Filtering. Thesis No. 1161, 2008. ISBN 978-91-7393-979-9.

**J. Sjöberg:** Optimal Control and Model Reduction of Nonlinear DAE Models. Thesis No. 1166, 2008. ISBN 978-91-7393-964-5.

**D. Törnqvist:** Estimation and Detection with Applications to Navigation. Thesis No. 1216, 2008. ISBN 978-91-7393-785-6.

**P-J. Nordlund:** Efficient Estimation and Detection Methods for Airborne Applications. Thesis No. 1231, 2008. ISBN 978-91-7393-720-7.

**H. Tidefelt:** Differential-algebraic equations and matrix-valued singular perturbation. Thesis No. 1292, 2009. ISBN 978-91-7393-479-4.

**H. Ohlsson:** Regularization for Sparseness and Smoothness — Applications in System Identification and Signal Processing. Thesis No. 1351, 2010. ISBN 978-91-7393-287-5.

**S. Moberg:** Modeling and Control of Flexible Manipulators. Thesis No. 1349, 2010. ISBN 978-91-7393-289-9.

**J. Wallén:** Estimation-based iterative learning control. Thesis No. 1358, 2011. ISBN 978-91-7393-255-4.

**J. D. Hol:** Sensor Fusion and Calibration of Inertial Sensors, Vision, Ultra-Wideband and GPS. Thesis No. 1368, 2011. ISBN 978-91-7393-197-7.

**D. Ankelhed:** On the Design of Low Order H-infinity Controllers. Thesis No. 1371, 2011. ISBN 978-91-7393-157-1.

**C. Lundquist:** Sensor Fusion for Automotive Applications. Thesis No. 1409, 2011. ISBN 978-91-7393-023-9.

**P. Skoglar:** Tracking and Planning for Surveillance Applications. Thesis No. 1432, 2012. ISBN 978-91-7519-941-2.

**K. Granström:** Extended target tracking using PHD filters. Thesis No. 1476, 2012. ISBN 978-91-7519-796-8.

**C. Lyzell:** Structural Reformulations in System Identification. Thesis No. 1475, 2012. ISBN 978-91-7519-800-2.

**J. Callmer:** Autonomous Localization in Unknown Environments. Thesis No. 1520, 2013. ISBN 978-91-7519-620-6.

**D. Petersson:** A Nonlinear Optimization Approach to H2-Optimal Modeling and Control. Thesis No. 1528, 2013. ISBN 978-91-7519-567-4.

**Z. Sjanic:** Navigation and Mapping for Aerial Vehicles Based on Inertial and Imaging Sensors. Thesis No. 1533, 2013. ISBN 978-91-7519-553-7.

**F. Lindsten:** Particle Filters and Markov Chains for Learning of Dynamical Systems. Thesis No. 1530, 2013. ISBN 978-91-7519-559-9.

**P. Axelsson:** Sensor Fusion and Control Applied to Industrial Manipulators. Thesis No. 1585, 2014. ISBN 978-91-7519-368-7.

**A. Carvalho Bittencourt:** Modeling and Diagnosis of Friction and Wear in Industrial Robots. Thesis No. 1617, 2014. ISBN 978-91-7519-251-2.

**M. Skoglund:** Inertial Navigation and Mapping for Autonomous Vehicles. Thesis No. 1623, 2014. ISBN 978-91-7519-233-8.

**S. Khoshfetrat Pakazad:** Divide and Conquer: Distributed Optimization and Robustness Analysis. Thesis No. 1676, 2015. ISBN 978-91-7519-050-1.

**T. Ardeshiri:** Analytical Approximations for Bayesian Inference. Thesis No. 1710, 2015. ISBN 978-91-7685-930-8.

**N. Wahlström:** Modeling of Magnetic Fields and Extended Objects for Localization Applications. Thesis No. 1723, 2015. ISBN 978-91-7685-903-2.

**J. Dahlin:** Accelerating Monte Carlo methods for Bayesian inference in dynamical models. Thesis No. 1754, 2016. ISBN 978-91-7685-797-7.

**M. Kok:** Probabilistic modeling for sensor fusion with inertial measurements. Thesis No. 1814, 2016. ISBN 978-91-7685-621-5.

**J. Linder:** Indirect System Identification for Unknown Input Problems: With Applications to Ships. Thesis No. 1829, 2017. ISBN 978-91-7685-588-1.

**M. Roth:** Advanced Kalman Filtering Approaches to Bayesian State Estimation. Thesis No. 1832, 2017. ISBN 978-91-7685-578-2.

**I. Nielsen:** Structure-Exploiting Numerical Algorithms for Optimal Control. Thesis No. 1848, 2017. ISBN 978-91-7685-528-7.