Linköping Studies in Science and Technology
Thesis No. 866

Linear Model Predictive Control

Stability and Robustness

Johan Löfberg

REGLERTEKNIK
AUTOMATIC CONTROL
LINKÖPING

Division of Automatic Control
Department of Electrical Engineering
Linköpings universitet, SE–581 83 Linköping, Sweden
WWW: `http://www.control.isy.liu.se`
Email: `johanl@isy.liu.se`

Linköping 2001

Linear Model Predictive Control
Stability and Robustness

Department of Electrical Engineering,
Linköpings universitet,
SE–581 83 Linköping,
Sweden.

ISBN 91-7219-920-2
ISSN 0280-7971
LiU-TEK-LIC-2001:03

Printed by UniTryck, Linköping, Sweden 2001

Abstract

Most real systems are subjected to constraints, both on the available control effort and the controlled variables. Classical linear feedback is in some cases not enough for such systems. This has motivated the development of a more complicated, nonlinear controller, called model predictive control, MPC. The idea in MPC is to repeatedly solve optimization problems on-line in order to calculate control inputs that minimize some performance measure evaluated over a future horizon.

MPC has been very successful in practice, but there are still considerable gaps in the theory. Not even for linear systems does there exist a unifying stability theory, and robust synthesis is even less understood.

The thesis is basically concerned with two different aspects of MPC applied to linear systems. The first part is on the design of terminal state constraints and weights for nominal systems with all states avaliable. Adding suitable terminal state weights and constraints to the original performance measure is a way to guarantee stability. However, this is at the cost of possible loss of feasibility in the optimization. The main contribution in this part is an approach to design the constraints so that feasibility is improved, compared to the prevailing method in the literature. In addition, a method to analyze the actual impact of ellipsoidal terminal state constraints is developed.

The second part of the thesis is devoted to synthesis of MPC controllers for the more realistic case when there are disturbances acting on the system and there are state estimation errors. This setup gives an optimization problem that is much more complicated than in the nominal case. Typically, when disturbances are incorporated into the performance measure with minimax (worst-case) formulations, NP-hard problems can arise. The thesis contributes to the theory of robust synthesis by proposing a convex relaxation of a minimax based MPC controller. The framework that is developed turns out to be rather flexible, hence allowing various extensions.

Acknowledgments

First of all, I would like to express my gratitude to my supervisors Professor Lennart Ljung and Professor Torkel Glad. Not only for guiding me in my research, but also for drafting me to our group.

Of course, it goes without saying that all colleagues contributes to the atmosphere that makes it a pleasure to endure yet some years at the university. However, there are some that deserve an explicit mentioning. To begin with, Fredrik Tjärnström and Måns Östring both have the dubious pleasure to have their offices close to me, hence making them suitable to harass with various questions. Sometimes even research related. Jacob Roll, Ola Härkegård and David Lindgren read various parts of this thesis, and their comments are gratefully acknowledged. Dr. Valur Einarsson is acknowledged for fruitful discussions on convex optimization.

The work was supported by the NUTEK competence center Information Systems for Industrial Control and Supervision (ISIS), which is gratefully acknowledged. Special thanks goes to ABB Automation Products AB, Dr. Mats Molander in particular.

Finally, I would like to thank friends and family. I can not say that you have improved the quality of this thesis, more likely the opposite, but I thank you for putting out with me during the most busy periods.

Linköping, January 2001

Johan Löfberg

Contents

Notation

Symbols

$\mathcal{E}(x, P)$	Ellipsoid centered at x with shape matrix P
$\mathcal{E}_P$	Ellipsoid centered at the origin with shape matrix P
I	Identity matrix

Operators and Functions

$A \succ (\succeq)\ 0$	A positive (semi-)definite matrix
$A \prec (\preceq)\ 0$	A negative (semi-)definite matrix
A^T	Transpose of a matrix
A^{-1}	Inverse of a matrix
$\mathrm{tr}(A)$	Trace of a matrix
$\det(A)$	Determinant of a matrix
$\lambda_{max}(A)$	Largest eigenvalue of a matrix
$\lambda_{min}(A)$	Smallest eigenvalue of a matrix
$\|\|x\|\|$	Euclidian norm of a vector
$\|x\|$	Elementwise absolute value of a vector

$X \subset Y$	X is a subset of Y
$X \setminus Y$	The set $\{x \ : \ x \in \mathcal{X} \text{ and } x \notin \mathcal{Y}\}$
$\mathbf{Co}(X_1, \ldots, X_n)$	Convex hull of $\{X_i\}$
$\mathbf{V}(X)$	Vertices of polytope
$\mathbf{F}(X)$	Facets of polyhedron
$\partial(X)$	Boundary of set

Abbreviations

BMI	Bilinear Matrix Inequality
LMI	Linear Matrix Inequality
LQ	Linear Quadratic
MAXDET	Determinant Maximization
MPC	Model Predictive Control
QP	Quadratic Program(ming)
SDP	Semidefinite Program(ming)
SOCP	Second Order Cone Program(ming)

1 Introduction

In this thesis, we deal with aspects of linear model predictive control, or MPC for short. Leaving the technical details aside until Chapter 3, this chapter will explain the basic idea of MPC and summarize the content of the thesis.

A provoking analogy between MPC and classical control can be found in [15]: If we want to control the position of a car, MPC is equivalent to look at the road through the windscreen, whereas classical control only is allowed to look in the rear window. Of course, this comparison is unfair, but it describes in a simple way how MPC works; it tries to control a system (the car) by creating predictions about the future (position on the road ahead) using a model (impact of steering and acceleration) while taking care of constraints (traffic rules and car performance).

By using the predictions, the MPC controller calculates the optimal input. In the car, this would be the steering and adjustment of the speed. The calculation of the optimal input can for some applications take a long time. To overcome this, the problem has to be solved over a short prediction horizon, just as when we drive a car and only look some hundred meters ahead. Furthermore, the MPC controller continuously recalculates the input. The same is done when we drive a car, i.e., we do not plan for the following one hundred meters, close our eyes and drive this distance, and then decide on a new input for the next one hundred meters.

The problems that might occur when applying MPC can also be explained with the car analogy. In the car, looking too short ahead might lead to disastrous effects, we might crash with slower cars or drive off the road. In MPC, the effect of a too short horizon is the possibility of poor performance, or even instability. What is needed to solve this problem is an MPC controller that knows that strange things might occur beyond the horizon, and explicitly takes precautions to handle this.

Another problem is that of model uncertainty and disturbances. The model we use to calculate our predictions might be wrong, hence leading us to give bad inputs to be applied to the system.

Problems related to these issues is what we will deal with in this thesis.

1.1 Outline and Contributions

Most results in this thesis are based on application of semidefinite programming, linear matrix inequalities and convex sets. For that reason, repeatedly used concepts and definitions from these fields are compiled in Chapter 2.

The history and background of MPC is shortly reviewed in Chapter 3. Admissible systems and the standard formulation of MPC is introduced. Finally, there is a discussion on stability theory for MPC.

Following the introduced stability theory, these ideas are extended in Chapter 4 and 5. The idea is to use a more advanced terminal state weight based on a piecewise quadratic function. With this extension, it is possible to improve feasibility, in MPC controllers with guaranteed stability, compared to traditional approaches. Although a more advanced terminal state weight is employed, the same optimization routines as in the standard approaches can be used.

Still, feasibility is an issue. Therefore, feasibility analysis of ellipsoidal terminal state constraints, which is used Chapter 4 and 5 and most MPC controllers with guaranteed stability, is performed in Chapter 6. An algorithm to calculate the largest possible set in which the optimization problem is feasible is developed. An exact characterization of this set is also given.

Finally, a novel approach to robust MPC for systems with unknown but bounded disturbances and state estimation errors is proposed in Chapter 7. The chapter starts with a description of state estimation for system with bounded disturbances. Using the same framework as for the state estimation part, a minimax MPC controller is formulated. The obtained optimization problem is relaxed by using the S-procedure and gives a convex optimization problem that can be solved efficiently.

To summarize, the main contributions of this thesis are:

- Extension of the archetypal approach to MPC with guaranteed stability in Chapter 4 and 5. As an intermediate step, some minor improvements in the design of switching controllers is proposed.
- Feasibility analysis of ellipsoidal terminal state constraints, with exact characterization and calculation of the admissible initial set in Chapter 6.
- The LMI solution in Chapter 7 for robust disturbance rejection and treatment of state estimation errors in MPC. As a part of this work, a result on synthesis of linear feedback control for constrained discrete-time system subjected to bounded disturbances is presented.

Related publications

Chapter 4 is primarily based on

[48] J. Löfberg. A Stabilizing MPC Algorithm using Performance Bounds from Saturated Linear Feedback. In *Proceedings of the 39th IEEE Conference on Decision and Control, Sydney, Australia*, 2000.

The material in Chapter 6 is based on

[47] J. Löfberg. Feasibility analysis of MPC with ellipsoidal terminal state constraints. In *Proceedings of Reglermöte, Uppsala, Sweden*, 2000.

Publications not included in thesis

Related work on nonlinear system has been done but is not included in this thesis.

[46] J. Löfberg. Backstepping with local LQ performance and global approximation of quadratic performance. In *Proceedings of American Control Conference, Chicago, USA*, 2000.

2

Mathematical Preliminaries

In this short chapter, some repeatedly used definitions and mathematical concept are gathered for easy reference.

2.1 Convex Optimization and LMIs

In the field of optimization, the crucial property is not linearity but convexity. Recently, there has been much development for problems where the constraints can be written as the requirement of a matrix to be positive semidefinite. This is a convex constraint and motivates the definition of a linear matrix inequality, LMI.

Definition 2.1 (LMI)
An LMI (linear matrix inequality) is an inequality, in the free scalar variables x_i, that for some fixed symmetric matrices F_i can be written as

$$F(x) = F_0 + x_1F_1 + x_2F_2 + \ldots + x_nF_n \succeq 0 \tag{2.1}$$

In the definition above, we introduced the notion of a semidefinite matrix $F \succeq 0$ which means $F = F^T$ and $z^TFz \geq 0\ \forall z$.

As an example of an LMI, the nonlinear constraint $x_1x_2 \geq 1, x_1 \geq 0, x_2 \geq 0$ can be written

$$\begin{bmatrix} x_1 & 1 \\ 1 & x_2 \end{bmatrix} \succeq 0 \tag{2.2}$$

The matrix can be decomposed and we obtain

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + x_1 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + x_2 \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \succeq 0 \tag{2.3}$$

An excellent introduction to LMIs, with special attention to problems in control theory, can be found in [11].

By using LMIs, many convex optimization problems, such as linear programming and quadratic programming, can be unified by the introduction of semidefinite programming [66].

Definition 2.2 (SDP)
An SDP (semidefinite program), is an optimization problem that can be written as

$$\begin{aligned} \underset{x}{\mathit{min}} \quad & c^T x \\ \mathit{subject\ to} \quad & F(x) \succeq 0 \end{aligned} \tag{2.4}$$

SDPs can today be solved with high efficiency, i.e., with polynomial complexity, due to the recent development of solvers using interior-point methods [51]. SDPs arising in this thesis will be solved with [65]. A special class of SDP is MAXDET problems.

Definition 2.3 (MAXDET)
A MAXDET problem (determinant maximization) is an optimization problem that can be written as

$$\begin{aligned} \underset{x}{\mathit{min}} \quad & c^T x - \log\det(G(x)) \\ \mathit{subject\ to} \quad & F(x) \succeq 0 \\ & G(x) \succeq 0 \end{aligned} \tag{2.5}$$

This is an optimization problem that frequently occurs in problems where the analysis is based on ellipsoids. A MAXDET problem can be converted to a standard SDP [51], and thus solved with general SDP solvers. However, there exist special purpose solvers for MAXDET problems which we will use [69]. Another special SDP is SOCP [45].

Definition 2.4 (SOCP)
A SOCP (second order cone program) is an optimization problem with the structure

$$\begin{aligned} \underset{x}{\mathit{min}} \quad & c^T x \\ \mathit{subject\ to} \quad & ||A_i x + b_i|| \le c_i^T x + d_i \end{aligned} \tag{2.6}$$

This problem can easily be rewritten into an SDP. Due to the special structure however, a more efficient method to solve the problem is to use special purpose solvers such as [44].

The LMIs in this thesis have been defined using [49] which is a MATLAB™ package that acts as an interface to the solvers [65], [69] and [44].

As we will see in this thesis, the following two theorems are extremely useful when dealing with LMIs and SDPs. The first theorem can be used to (conservatively) rewrite constraints, in optimization theory often called relaxation.

Theorem 2.1 (S-procedure)
Let $T_0(x), \dots, T_p(x)$ be quadratic functions,

$$T_i(x) = x^T P_i x, \quad P_i = P_i^T, \quad i = 0, \dots, p \tag{2.7}$$

A sufficient condition for

$$T_0(x) \geq 0 \text{ for all } x \text{ such that } T_i(x) \geq 0, \quad i = 1, \dots, p \tag{2.8}$$

to hold is that there exist scalars $\tau_i \geq 0,\ i = 1, \dots, p$ such that

$$T_0(x) - \sum_{i=1}^{p} \tau_i T_i(x) \geq 0 \tag{2.9}$$

In the special case $p = 1$, the condition is also necessary.

Proof See [11]. □

Typically, the S-procedure is used by observing that since $T_i(x) = x^T P_i x$, the variable x can be eliminated and we obtain the constraint

$$P_0 - \sum_{i=1}^{p} \tau_i P_i \succeq 0 \tag{2.10}$$

This is an LMI in the variables τ_i and P_0. The interested reader is referred to [64] for a nice treatment on various facts about the S-procedure.

The second theorem helps us to convert certain nonlinear matrix inequalities into linear matrix inequalities.

Theorem 2.2 (Schur complement)
The matrix inequalities

$$X - Y^T Z^{-1} Y \succeq 0, \quad Z \succ 0 \tag{2.11}$$

and

$$\begin{bmatrix} X & Y^T \\ Y & Z \end{bmatrix} \succeq 0 \tag{2.12}$$

are equivalent.

Proof See [11]. □

2.2 Convex Sets

Many of the results in this thesis are based on calculations with convex sets. The sets we will use are the following

Definition 2.5 (Ellipsoid)
An ellipsoid centered at x_c is the set described by the quadratic inequality

$$\mathcal{E}(x_c, P) = \{x \ : \ (x - x_c)^T P(x - x_c) \leq 1\}, \quad P = P^T \succ 0 \tag{2.13}$$

For simple notation, the following notation is used when $x_c = 0$

$$\mathcal{E}_P = \mathcal{E}(0, P) = \{x \ : \ x^T P x \leq 1\} \tag{2.14}$$

The volume of the ellipsoid is proportional to $\det(P^{-1/2})$. *Since* $\det(P^{-1}) = \big(\det(P^{-1/2})\big)^2$, *the volume is monotonously increasing with* $\det(P^{-1})$.

Definition 2.6 (Polyhedron)
A polyhedron is a non-empty set described by a collection of linear inequalities

$$\mathcal{P} = \{x \ : \ c_i^T x \leq d_i\} \tag{2.15}$$

Definition 2.7 (Polytope)
A polytope is a closed polyhedron.

Definition 2.8 (Convex hull)
The convex hull of a collection of matrices $\{X_i\}$ is the set defined as

$$\mathbf{Co}\,(X_1, \ldots, X_r) = \{X \ : \ X = \sum_{j=1}^{r} \lambda_j X_j\}, \quad \sum_{j=1}^{r} \lambda_j = 1, \ \lambda_j \geq 0 \tag{2.16}$$

In words, all matrices that can be written as an interpolation of matrices in $\{X_i\}$.

We will often work with sets in the context of dynamic systems, and for that purpose, the following definition is fundamental.

Definition 2.9 (Positively invariant set, [9])
A subset Ω of the state-space is said to be positively invariant for the dynamic system $x(k+1) = f(x(k))$ if $f(x(k)) \in \Omega \ \forall x(k) \in \Omega$.

3

MPC

Model predictive control, or MPC, is a control paradigm with a motley background. The underlying ideas for MPC originated already in the sixties as a natural application of optimal control theory. Already in [54], a controller with close connections to MPC was developed, and a more general optimal control based feedback controller was discussed in [40]:

"*One technique for obtaining a feedback controller synthesis from knowledge of open-loop controllers is to measure the current control process state and then compute very rapidly for the open-loop control function. The first portion of this function is then used during a short time interval, after which a new measurement of the process state is made and a new open-loop control function is computed for this new measurement. The procedure is then repeated*".

As we will see in this and the following chapters, this is the definition of the control method that we today call MPC.

3.1 Historical Background

In mid-seventies to mid-eighties, the true birth of MPC took place, but this time it was in the industry. Advocated by the work on Model Predictive Heuristic Control

(MHRC) [58] and Dynamic Matrix Control (DMC) [19], the MPC strategy became popular in the petro-chemical industry. During this period, there was a flood of new variants of MPC. Without going into details, MAC, DMC, EHAC, EPSAC, GMV, MUSMAR, MURHAC, PFC, UPC and GPC were some of the algorithms [15]. Despite the vast number of abbreviations introduced, not much differed between the algorithms. Typically, they differed in the process model (impulse, step, state-space etc.), disturbance (constant, decaying, filtered white noise etc.) and adaptation.

During the nineties, the theory of MPC has matured substantially. The main reason is probably the use of state-space models instead of input-output models. This has simplified, unified and generalized much of the theory. In the case of non-accessible states, the Kalman filter (most easily used in a state-space formulation) simplifies the estimation part, the connections to linear quadratic control give a lot of insight [8], stability theory is almost only possible in a state-space formulation and a lot of recent MPC theory is based on linear matrix inequalities which are most suitable for state-space methods.

In this chapter, we will describe the basics of an MPC algorithm. The admissible systems will be defined and some simple notation will be explained. After the introduction of a standard MPC controller, stability issues will be discussed, since this is a central part in this thesis.

3.2 System Setup

In this thesis, we will exclusively use state-space methods. The reason is that a lot of analysis will be done within a Lyapunov framework, which is most naturally performed in the state-space. The system we analyze will in principle be the same throughout this thesis

$$x(k+1) = Ax(k) + Bu(k) \tag{3.1a}$$

$$y(k) = Cx(k) \tag{3.1b}$$

where $x(k) \in \mathbf{R}^n$, $u(k) \in \mathbf{R}^m$, $y(k) \in \mathbf{R}^p$ denote the state, control input and measured output respectively. It is a standing assumption that the system is both controllable and observable. Besides the dynamics, the system is saturated, and we conceptually write this control constraint as

$$u \in \mathcal{U} \tag{3.2}$$

We assume that $\mathcal{U}$ is a non-empty set described with linear inequalities, i.e., a polyhedron. We call $\mathcal{U}$ the control constraint polytope. As an example, this is the case when there are amplitude constraints, $u_{min} \leq u(k) \leq u_{max}$.

3.3 A Basic MPC Controller

MPC is an optimization based control law, and the performance measure is almost always a quadratic cost. By defining positive definite matrices $Q = Q^T \succ 0$ and

$R = R^T \succ 0$ (the performance weights), our underlying goal is to find the optimal control input that minimizes the infinite horizon performance measure, or cost,

$$J(k) = \sum_{j=k}^{\infty} x^T(j|k)Qx(j|k) + u^T(j|k)Ru(j|k) \tag{3.3}$$

In the unconstrained case, the solution to this problem is given by the linear quadratic (LQ) controller. In the constrained case however, there does not exist any analytic solution. Instead, the idea in MPC is to define a *prediction horizon* N and approximate the problem with a finite horizon cost

$$J(k) = \sum_{j=k}^{k+N-1} x^T(j|k)Qx(j|k) + u^T(j|k)Ru(j|k) \tag{3.4}$$

The term finite horizon is crucial. It is due to the finite horizon that we are able to solve the problem, but at the same time, the finite horizon will introduce problems. This will be discussed more in Section 3.4.

By using the model (3.1), we can predict the state $x(k+j|k)$, given a *future control sequence* $u(\cdot|k)$ and the current state $x(k|k)$. Until Chapter 7, we will assume $C = I$, hence no state estimation is required and $x(k|k) = x(k)$. This gives the prediction

$$x(k+j|k) = A^j x(k|k) + \sum_{i=0}^{j-1} A^{j-i-1}Bu(k+i|k) \tag{3.5}$$

Using these predictions, we define the following optimization problem

$$\boxed{\begin{array}{rl} \min\limits_{u} & \sum_{j=k}^{k+N-1} x^T(j|k)Qx(j|k) + u^T(j|k)Ru(j|k) \\ \text{subject to} & u(k+j|k) \in \mathcal{U} \\ & x(k+j|k) = Ax(k+j-1|k) + Bu(k+j-1|k) \end{array}} \tag{3.6}$$

At this point, we are able to define a basic MPC controller

Algorithm 3.1 (Basic MPC controller)

1. *Measure* $x(k|k)$
2. *Obtain* $u(\cdot|k)$ *by solving (3.6)*
3. *Apply* $u(k) = u(k|k)$
4. *Time update,* $k := k+1$
5. *Repeat from step* 1

Already in [54], it was realized that the optimization problem above is a quadratic program (QP), i.e., minimization of a quadratic objective subject to linear constraints. The earliest reference that takes advantage of this fact is probably [22], although it had been in use in the industry for quit some time before. QP is a classical optimization problem for which there exist a large number of efficient solution methods, and this is probably one of the reasons why MPC has become so popular in practice.

3.3.1 QP Formulation of MPC

To put the optimization problem in a form suitable for QP, we introduce stacked vectors with future states and control inputs

$$X = \begin{bmatrix} x(k|k) \\ x(k+1|k) \\ \vdots \\ x(k+N-1|k) \end{bmatrix}, \quad U = \begin{bmatrix} u(k|k) \\ u(k+1|k) \\ \vdots \\ u(k+N-1|k) \end{bmatrix} \tag{3.7}$$

The predicted states can be written as

$$X = Hx(k|k) + SU \tag{3.8}$$

where $H \in \mathbf{R}^{Nn \times n}$ and $S \in \mathbf{R}^{Nn \times Nm}$ are defined using (3.5) (see, e.g., [15] or [26]). We create enlarged versions of the Q and R matrix, $\bar{Q} = \operatorname{diag}(Q, Q, \dots, Q) \in \mathbf{R}^{Nn \times Nn}$ and $\bar{R} = \operatorname{diag}(R, R, \dots, R) \in \mathbf{R}^{Nm \times Nm}$. Since the constraint on the input is defined by linear inequalities, they can be written as $EU \leq f$ for some suitably defined matrix E and vector f. The optimization problem (3.6) can now be written as

$$\boxed{\begin{array}{ll} \min\limits_{U} & (Hx(k|k) + SU)^T \bar{Q} (Hx(k|k) + SU) + U^T \bar{R} U \\ \text{subject to} & EU \leq f \end{array}} \tag{3.9}$$

For overviews on solution methods of QPs, with special attention to MPC, see, e.g, [15] where the classical active set method is reviewed, or [68] where more recent advances in convex optimization are utilized.

Remark 3.1
The basic MPC algorithm can be extended in numerous ways within the QP framework. Typical variations are different horizons on states and inputs, state and input constraints such as rate, rise-time, and overshoot constraints. Tracking is dealt with by substituting $Hx(k|k) + SU$ with $Hx(k|k) + SU - X_{ref}$ and $U^T \bar{R} U$ with $(U - U_{ss})^T \bar{R} (U - U_{ss})$ where X_{ref} is the desired future state-trajectory and U_{ss} is the corresponding steady state control input. See, e.g, [15, 56, 68] for more detailed discussions on what can be done.

3.4 Stability of MPC

From a theoretical point of view, the main problem with MPC has been the lack of a general and unifying stability theory. Already without input constraints, the MPC controller as defined in Algorithm 3.1 can generate an unstable closed loop. Asymptotic stability can in the unconstrained case always be obtained by choosing R large enough or having a sufficiently long horizon [23], but as argued in [8], without constraints we could just as well solve the infinite horizon problem and obtain an LQ controller. Based on this, the unconstrained case is in some sense a non-issue and will not be dealt with in this thesis.

In the constrained case, the situation is different. For an unstable input constrained system, there does not even exist a globally stabilizing controller [59]. Since global stability is impossible in the general case, the term stability will in this thesis therefore mean local stability, but we will not explicitly write this. Typically, stability will depend upon feasibility of the optimization problem.

Although the system we analyze is linear, the input constraint introduces a nonlinearity which severely complicates the stability analysis. Another complication is that the control law is generated by the solution of an optimization, i.e., the control law cannot be expressed in a closed form. These two obstacles prevented the development of stability results in the early days of MPC. The situation was even more complicated by the fact that the analysis often was performed in an input-output setting. As state-space formulations became standard in MPC, stability results begun appearing in late eighties and early nineties.

The central concept that started to appear was not to study the impact of different choices of the tuning parameters (Q, R and N), since these parameters in general affect stability in a non-convex manner [55]. Instead, the idea is to reformulate the underlying optimization problem in order to guarantee stability. An excellent survey on stability theory for MPC can be found in [50].

3.4.1 A Stabilizing MPC Controller

In this section, a rather general approach to stabilizing MPC will be introduced. The method is based on three ingredients: a nominal controller, a terminal state domain that defines a terminal state constraint, and a terminal state weight. Having these, it is possible to summarize many proposed schemes in the following theorem.

Theorem 3.1 (Stabilizing MPC)
Suppose the following assumptions hold for a nominal controller $L(x)$, a terminal state domain $\mathcal{X}$ and a terminal state weight $\Psi(x)$

1. $0 \in \mathcal{X}$
2. $Ax + BL(x) \in \mathcal{X},\ \forall x \in \mathcal{X}$
3. $\Psi(0) = 0,\ \Psi \succ 0$
4. $\Psi(Ax + BL(x)) - \Psi(x) \leq -x^T Qx - L^T(x)RL(x),\ \forall x \in \mathcal{X}$

5. $L(x) \in \mathcal{U}, \ \forall x \in \mathcal{X}$

Then, assuming feasibility at the initial state, an MPC controller using the following optimization problem will guarantee asymptotic stability

$$\boxed{\begin{array}{rl} \min\limits_{u} & \sum_{j=k}^{k+N-1} x^T(j|k)Qx(j|k) + u^T(j|k)Ru(j|k) + \Psi(x(k+N|k)) \\ \textit{subject to} & x(k+j|k) = Ax(k+j-1|k) + Bu(k+j-1|k) \\ & u(k+j|k) \in \mathcal{U} \\ & x(k+N|k) \in \mathcal{X} \end{array}} \tag{3.10}$$

Proof The proof is based on using the optimal cost as a Lyapunov function. Let us denote the cost $J(k)$, and the optimal $J^*(k)$. The optimal cost at time k is obtained with the control sequence $[u^*(k|k) \ \ldots \ u^*(k+N-1|k)]$ The use of a $*$ is a generic notation for variables related to optimal solutions.

A feasible solution at time $k+1$ is $[u^*(k+1|k) \ \ldots \ u^*(k+N-1|k) \ L(x^*(k+N|k))]$. To see this, we first recall that $x^*(k+N|k) \in \mathcal{X}$ according to the optimization. Using Assumption 5, we see that $L(x^*(k+N|k))$ satisfies the control constraint, and Assumption 2 assures satisfaction of the terminal state constraint on $x(k+N+1|k)$. The cost using this (sub-optimal) control sequence will be

$$\begin{aligned} J(k+1) &= \sum_{j=k+1}^{k+N-1} [x^{*T}(j|k)Qx^*(j|k) + u^{*T}(j|k)Ru^*(j|k)] + x^T(k+N|k)Qx(k+N|k) \\ &\quad + L^T(x(k+N|k))RL(x(k+N|k)) + \Psi(x(k+N+1|k)) \\ &= \sum_{j=k}^{k+N-1} [x^{*T}(j|k)Qx^*(j|k) + u^{*T}(j|k)Ru^*(j|k)] + \Psi(x^*(k+N|k)) \\ &\quad + \Psi(x(k+N+1|k)) - \Psi(x^*(k+N|k)) \\ &\quad + x^T(k+N|k)Qx(k+N|k) + L^T(x(k+N|k))RL(x(k+N|k)) \\ &\quad - x^T(k|k)Qx(k|k) - u^{*T}(k|k)Ru^*(k|k) \end{aligned}$$

In the equation above, we added and subtracted parts from the optimal cost at time k. This is a standard trick in stability theory of MPC, and the reason is that the first line in the last equality now corresponds to the optimal cost at time k, i.e., $J^*(k)$.

According to Assumption 4, the sum of the second and third row in the last equality is negative. Using this, we obtain

$$J(k+1) \leq J^*(k) - x^T(k|k)Qx(k|k) - u^{*T}(k|k)Ru^*(k|k)$$

Since our new control sequence was chosen without optimization (we only picked a feasible sequence) we know that $J(k+1) \geq J^*(k+1)$. In other words, we have

$$J^*(k+1) \leq J^*(k) - x^T(k|k)Qx(k|k) - u^{*T}(k|k)Ru^*(k|k)$$

which proves that $J^*(k)$ is a decaying sequence, hence $x(k)$ converges to the origin. $\square$

Details and more rigorous proofs can be found in, e.g., [41] and [50].

The assumptions in the theorem are easily understood with a more heuristic view. If we use the controller $u(k) = L(x(k))$, and start in $\mathcal{X}$, we know that

$$\Psi(x(k+1)) - \Psi(x(k)) \leq -x^T(k)Qx(k) - u^T(k)Ru(k) \tag{3.11}$$

By summing the left- and right-hand from time k to infinity, we obtain

$$\Psi(x(\infty)) - \Psi(x(k)) \leq \sum_{j=k}^{\infty} -x^T(j)Qx(j) - u^T(j)Ru(j) \tag{3.12}$$

Now, according to the assumptions, it is easy to see that $\Psi(x(k))$ is a Lyapunov function when we are using the controller $L(x)$, so we know that $x(k) \to 0$. Using this, we obtain

$$\sum_{j=k}^{\infty} x^T(j)Qx(j) + u^T(j)Ru(j) \leq \Psi(x(k)) \tag{3.13}$$

In other words, $\Psi(x)$ is an upper bound of the infinite horizon cost, when we use the (sub-optimal) controller $L(x)$. Obviously, the optimal cost is even lower, so $\Psi(x)$ is an upper bound of the optimal cost also. This is the most intuitive way to interpret the assumptions. The terminal state weight $\Psi(x)$ is an upper bound of the optimal cost in the terminal state domain $\mathcal{X}$.

3.4.2 Methods to Choose $\{\mathcal{X}, L(x), \Psi(x)\}$

So, having a fairly general theorem for stabilizing MPC controllers, what is the catch? The problem is of course to find the collection $\{\mathcal{X}, L(x), \Psi(x)\}$. A number of methods have been proposed over the years.

Terminal state equality

A very simple method, generalizing the basic idea in, e.g., [36], was proposed and analyzed in the seminal paper [35]. The method holds for a large class of systems, performance measures and constraints, and in order to guarantee stability, a terminal state equality is added to the optimization

$$x(k+N|k) = 0 \tag{3.14}$$

In terms of Theorem 3.1, this corresponds to $\mathcal{X} = \{0\}$, $L(x) = 0$ and $\Psi(x) = 0$. The set-up is successful since $L(x) = 0$ trivially satisfies all assumptions of Theorem 3.1 in the origin. Notice that the constraint only is artificial, the state will not reach the origin at time $k + N$, since this is a constraint that continuously is shifted forward in time.

Although this is an extremely simple approach, it has its flaws. Firstly, feasibility is a major problem. The constraint might lead to the need of a long horizon in order to obtain feasibility, see Example 4.4. A related issue is that the terminal state constraint can lead to a control law with a dead-beat behavior if the horizon is short.

Terminal state weight

For stable systems, the following approach can be taken. Since the system is stable, a stabilizing controller is $L(x) = 0$. This controller satisfies the control constraints in the whole state-space, hence $\mathcal{X} = \mathbf{R}^n$. Finally, we select the terminal state weight $\Psi(x)$ to be a quadratic function, $\Psi(x) = x^T P x$. For Assumption 4 in Theorem 3.1 to hold, we must have

$$A^T P A - P \preceq -Q \tag{3.15}$$

Hence, all we have to do is to solve a Lyapunov equation to find P. This was essentially the idea used in [57].

Terminal state weight and constraint

By combining results from the two approaches above, a more general scheme is obtained. In [57], they were combined in the sense that only the unstable modes of the system were forced to the origin in the prediction, whereas a terminal state weight as above was applied to the stable modes.

Later, generalizing the ideas in, e.g., [57], the following idea emerged. First, select the nominal controller as a linear feedback $-Lx$. As above, we then select a quadratic terminal state weight $\Psi(x) = x^T P x$. If we for the moment neglect any control constraints, Assumption 4 can be written as

$$(A - BL)^T P (A - BL) - P \preceq -Q - L^T R L \tag{3.16}$$

Notice that this constraint tells us that the function $x^T P x$ is a Lyapunov function for the unconstrained system, hence the levels sets of this Lyapunov function can be used to define the set $\mathcal{X}$ since Assumption 2 then will be satisfied. Now, if we take the constraints into account, we see that the control constraint is mapped into a state constraint $-Lx \in \mathcal{U}$. According to the restrictions on $\mathcal{U}$, this will be a polyhedron. We now search for the largest possible ellipsoid $x^T P x \leq \gamma$ contained in this polyhedron and use this as the terminal state domain. With these choices, all the assumptions in Theorem 3.1 are satisfied. Details concerning the actual calculations will be clarified in the next chapter. Notice that the terminal state constraint is a convex constraint, but it is quadratic, not linear. The MPC problem can therefore no longer be solved with QP. Instead, one has to rely on second order cone programming (see Definition 2.4).

The described approach is the cornerstone in many algorithms for stabilizing MPC [18, 21, 41, 43, 63, 70], and the results in Chapter 4 and 5 are extensions of this approach.

There is one very important feature with this approach which motivates this choice of terminal state weight. We see that if we pick L to be the LQ controller, the matrix P is the Riccati solution to the LQ problem and gives us the optimal cost for the unconstrained infinite horizon problem

$$x^T(k+N|k) P x(k+N|k) = \min_{u \in \mathbf{R}^m} \sum_{j=k}^{\infty} x^T(j|k) Q x(j|k) + u^T(j|k) R u(j|k) \tag{3.17}$$

Now, if the control constraints are inactive for $i \geq k+N$ in the solution of the constrained infinite horizon problem, and the terminal state constraint is satisfied in this solution, we must have

$$\min_{u \in \mathcal{U}} \quad \sum_{j=k}^{\infty} x^T(j|k)Qx(j|k) + u^T(j|k)Ru(j|k)$$

$$\Leftrightarrow$$

$$\min_{u \in \mathcal{U}} \quad \sum_{j=k}^{k+N-1} x^T(j|k)Qx(j|k) + u^T(j|k)Ru(j|k) + \min_{u \in \mathbf{R}^m} \sum_{j=k+N}^{\infty} x^T(j|k)Qx(j|k) + u^T(j|k)Ru(j|k)$$

$$\Leftrightarrow$$

$$\min_{\substack{u \in \mathcal{U} \\ x(k+N|k) \in \mathcal{X}}} \quad \sum_{j=k}^{k+N-1} x^T(j|k)Qx(j|k) + u^T(j|k)Ru(j|k) + x^T(k+N|k)Px(k+N|k)$$

The finite horizon MPC solution will thus coincide with the infinite horizon problem whenever the control constraints are inactive beyond the prediction horizon and the terminal state constraint is inactive.

Further extensions

A number of extensions to the approach in this section are of course possible. An obvious improvement is not to use the matrix P in both the terminal state constraint and weight, but to search for the largest possible ellipsoid $\mathcal{X}$ satisfying the assumptions of Theorem 3.1. Finding such an ellipsoid can be done without too much effort as we will see later in this thesis, and this extension has been used in, e.g., [5]. Another improvement is to use ideas from [37] and let the nominal feedback matrix L and the matrix P be left as free variables to be chosen in the optimization. This approach was used, as an intermediate step, in [10]. The new optimization problem will remain convex, but SDP has to be used to solve the on-line optimization problem.

Of course, the terminal state domain does not have to be an ellipsoid. In [43], a polyhedral set is used instead.

3.4.3 Other Approaches

The main alternative to an algorithm based on Theorem 3.1 is to explicitly force a Lyapunov function to decrease by using so called contraction constraints. One example is [5]. Basically, a quadratic function x^TPx is chosen and is forced to decay along the trajectory, $x^T(k+j+1|k)Px(k+j+1|k) - x^T(k+j|k)Px(k+j|k) < 0$. A problem with this approach is how to chose P. Another problem is that the constraint is non-convex, hence leading to problems in the on-line optimization. However, stability can be guaranteed as long as an initial solution can be found.

A recent, very promising approach, is introduced in [53] where LMIs are used to study stability and robustness of optimization based controllers. Although the emphasis is on analysis of closed loop properties, the method can hopefully be extended so that it can be used for synthesis.

4

Stabilizing MPC using Performance Bounds from Saturated Controllers

In Section 3.4, we saw that many stabilizing MPC algorithms are based on an upper bound of the achievable performance, or cost, with a nominal *unsaturated* controller. The upper bound is used as a terminal state weight, and the domain where the upper bound holds is used for a terminal state constraint.

A natural choice is to pick the nominal controller to be the corresponding LQ feedback. We showed in Section 3.4.2, that with this choice, the MPC controller actually solves the underlying infinite horizon problem as soon as the constraints go inactive. The problem is that the size of the terminal state domain typically is quite small for this choice of nominal controller. Hence, for initial states far from the origin, the problem will become infeasible.

One solution to this problem is to use a nominal controller for which the terminal state domain can be made larger. The drawback with this approach is that this typically gives a terminal state weight that is a poor measure of the optimal infinite horizon cost, hence leading to decreased performance. Another way to overcome the feasibility problem is to use the extension mentioned in Section 3.4.2 with the terminal weight and constraint being optimization variables. However, this leads to a substantial increase of on-line optimization complexity.

In this chapter we extend the concept of a nominal controller in order to reduce the conflict between the size of the terminal state domain, appropriate terminal state weight and computational complexity. We will study the case with the control constraint $|u(k)| \leq 1$, i.e., saturation. The idea is to let the nominal controller be a saturated linear feedback controller and calculate an upper bound of the infinite horizon cost with this controller

$$J(k) = \sum_{j=k}^{\infty} x^T(j)Qx(j) + u^T(j)Ru(j), \quad u(j) = \mathrm{sat}(-Lx(j)) \tag{4.1}$$

together with an ellipsoidal domain where this bound holds.

Remark 4.1
The idea to use a saturated controller as the nominal controller has recently also been proposed in [21]. The work in [21] is based on a characterization of the domain where a saturated LQ controller actually is optimal. The difference compared to our approach is that the method in [21] only holds for single-input stable systems whereas the approach we will present is completely general. Furthermore, we are not interested in the domain where the saturated LQ controller is optimal. Instead, we look for a domain where the saturated controller merely stabilizes the system.

In order to introduce the main tools and the underlying ideas, we begin with a description of the standard approach introduced in Section 3.4.2. Some additional material needed for our extension will be described in Section 4.2 where a method to model saturation is introduced. Using this model, we derive an upper bound of the achievable performance for a saturated controller in Section 4.4. This bound is improved upon in Section 4.5. In the last part of the chapter, we look at how the improved bound can be incorporated in a stabilizing MPC controller.

4.1 Description of the Standard Approach

The first step in the classical method in Section 3.4.2 is to select a nominal controller $u = -Lx$. A typical choice is the corresponding LQ controller. We then introduce a quadratic terminal state weight $\Psi(x) = x^T Px,\ P = P^T \succ 0$. For Assumption 4 in Theorem 3.1 to hold, we must have

$$(A - BL)^T P(A - BL) - P \preceq -Q - L^T RL \tag{4.2}$$

The condition above is a linear matrix inequality (LMI, see Definition 2.1) in the variable P.

4.1.1 Positively Invariant Ellipsoid

The second step is to define the terminal state domain $\mathcal{X}$. Most common is to select an ellipsoidal domain $\mathcal{E}_W$. Assumption 2 in Theorem 3.1 is nothing but a positively invariance condition, see Definition 2.9, of $\mathcal{E}_W$ for the system controlled with the nominal controller. Clearly, the ellipsoid $\mathcal{E}_W$ is positively invariant if $x^T(k+1)Wx(k+1) \leq x^T(k)Wx(k)$. This can be written as the LMI

$$(A - BL)^T W(A - BL) - W \preceq 0 \tag{4.3}$$

4.1.2 Constraint Satisfaction in Ellipsoid

In the two LMIs above, we assumed that the control constraint was satisfied in $\mathcal{E}_W$ since we used $u = -Lx$. Hence, we must have

$$u = -Lx \in \mathcal{U} \text{ for all } \{x \ : \ x^T W x \leq 1\} \tag{4.4}$$

In terms of Theorem 3.1, this takes care of Assumption 5.

As an example, Figure 4.1 below illustrates a situation where the set defined by the control constraint generates two linear inequalities, resulting in a polyhedron containing the origin. Inside this polyhedron, there is an ellipsoid in which the constraints are satisfied.

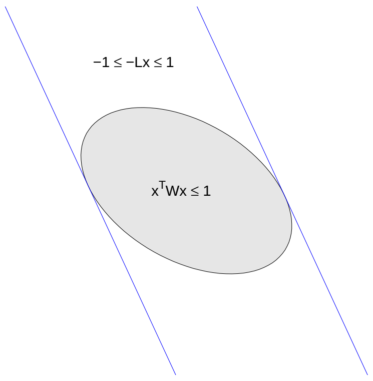

Figure 4.1: Largest possible ellipsoid (satisfying some positively invariance condition) contained in domain where control constraint are satisfied

Taking care of the constraint above in the optimization of W can be done by using the following Lemma.

Lemma 4.1
The maximum of $c^T x$ in the ellipsoidal set $x^T W x \leq 1$ is $\sqrt{c^T W^{-1} c}$

Proof Let $y = W^{1/2}x$. The objective is now maximization of $c^T W^{-1/2} y$ subject to $y^T y \leq 1$. Clearly, the optimal choice is the parallel vector $y = \left(\frac{c^T W^{-1/2}}{||c^T W^{-1/2}||}\right)^T$ which yields the objective $\frac{c^T W^{-1} c}{||c^T W^{-1/2}||} = \sqrt{c^T W^{-1} c}$. □

When we have amplitude constraints on the control input, $|u(k)| \leq 1$, Lemma 4.1 gives us the constraint $L_i W^{-1} L_i^T \leq 1$, where L_i denotes the ith row of L.

4.1.3 Maximization of Positively Invariant Ellipsoid

At this point, we have two constraints on W. Since $\mathcal{E}_W$ will be our terminal state domain, our goal is to maximize the size of $\mathcal{E}_W$. According to Definition 2.5, the volume of $\mathcal{E}_W$ is proportional to $\det(W^{-1})$, so this is our objective. We now have to show that this problem is a convex optimization problem. To do this, we introduce $Y = W^{-1}$. The LMI (4.3) is multiplied with W^{-1} from left and right and a Schur complement is applied. This gives us

$$\boxed{\begin{array}{rl} \max\limits_{Y} & \det(Y) \\ \text{subject to} & \begin{bmatrix} Y & Y(A-BL)^T \\ (A-BL)Y & Y \end{bmatrix} \succeq 0 \\ & L_iYL_i^T \leq 1 \end{array}} \tag{4.5}$$

Hence, we have obtained a MAXDET problem, see Definition 2.3. We will now show how this approach can be extended.

4.2 Saturation Modeling

The algorithm in this chapter is based on a polytopic representation of the saturated system. Polytopic representations are simple means to conservatively model nonlinearities as uncertainties in a linear system [11]. Using a polytopic uncertainty model to analyze stability of a saturated system is a standard procedure, and has been done before, see, e.g, [7]. Before we proceed with the description of the polytopic model, we introduce the concept saturation level.

Definition 4.1 (Saturation level)
For a system with control constraints, $|u(k)| \leq 1$, we define the saturation levels as

$$\gamma_i(k) \quad = \quad \begin{cases} 1 & |u_i(k)| \leq 1 \\ \frac{1}{|u_i(k)|} & |u_i(k)| > 1 \end{cases}$$

By introducing

$$\Gamma(k) \quad = \operatorname{diag}(\gamma_1(k), \dots, \gamma_m(k))$$

the saturated control input can be written as $\Gamma(k)u(k)$

If we know that the control $u_i(k)$ never saturates to a level less than $\gamma_{min,i}$, i.e.,

$$\gamma_{min,i} \leq \gamma_i(k) \leq 1$$

we can (conservatively) describe $\Gamma(k)$ as an interpolation of 2^m diagonal matrices Γ_j having the (i,i) element either 1 or $\gamma_{min,i}$

$$\Gamma(k) = \sum_{j=1}^{2^m} \lambda_j \Gamma_j \qquad \sum_{j=1}^{2^m} \lambda_j = 1, \ \lambda_j \geq 0$$

Using Definition 2.8, we say $\Gamma(k) \in \mathbf{Co}(\{\Gamma_j\})$. This is called a polytopic model of $\Gamma(k)$. Let us look at a simple example to illustrate the definition Γ_j.

Example 4.1 (Polytopic model of saturation) *If $\gamma_1 \geq 0.1$ and $\gamma_2 \geq 0.5$, the matrices to create the polytopic model of $\Gamma(k)$ are*

$$\Gamma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \Gamma_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 1 \end{bmatrix}, \Gamma_3 = \begin{bmatrix} 1 & 0 \\ 0 & 0.5 \end{bmatrix}, \Gamma_4 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.5 \end{bmatrix} \tag{4.6}$$

and $\Gamma(k)$ can (conservatively) be described as

$$\Gamma(k) = \lambda_1\Gamma_1 + \lambda_2\Gamma_2 + \lambda_3\Gamma_3 + \lambda_4\Gamma_4 \tag{4.7}$$

4.3 Invariant Domain for Saturated System

As we have seen before, a central part in the classical method is to find a positively invariant domain where the nominal controller is unsaturated. We are now going to extend these results and look for a positively invariant domain where the controller indeed saturates.

The problem of finding a positively invariant domain for a saturated system has been addressed by several authors, see, e.g., [7, 14, 20, 32, 33, 52].

The perhaps most simple method is to search for a Lyapunov function x^TWx, or equivalently an invariant ellipsoid $\mathcal{E}_W$. For an unstable system, $\mathcal{E}_W$ cannot be made arbitrarily large, since it is impossible to globally stabilize an unstable system subject to control constraints [59]. More advanced schemes using, e.g., the Popov criteria [32, 52] can also be used, but this would only make the derivation in this chapter less clear, and shift focus from the conceptual idea.

If we fix allowed saturation levels $\gamma_{min,i}$, the problem is to find the largest positively invariant set $\mathcal{E}_W$ that lies in the set where $\gamma_i(k) \geq \gamma_{min,i}$. In other words, an ellipsoidal set where the level of saturation is bounded from below. In Figure 4.2, an example is given with $\gamma_{min} = 0.3$.

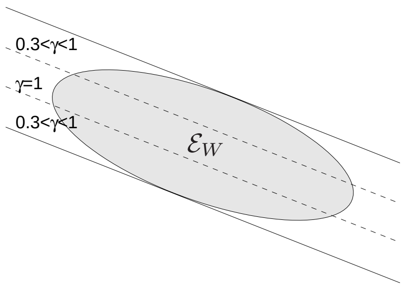

Figure 4.2: Example of largest possible positively invariant domain $\mathcal{E}_W$ inscribed in the domain where $\gamma \geq 0.3$. Dashed lines correspond to $|-Lx| = 1$ and solid lines to $|-Lx| = 1/0.3$

A solution to this problem is easily derived using the same arguments as in Section 4.1.1 and 4.1.2. We get the constraints

$$\begin{aligned} (A - B\Gamma(k)L)^T W (A - B\Gamma(k)L) - W &\preceq 0 && \text{(4.8a)} \\ L_i W^{-1} L_i^T &\leq \left(\frac{1}{\gamma_{min,i}}\right)^2 && \text{(4.8b)} \end{aligned}$$

The difference compared to earlier is two-fold. To begin with, the controller does not deliver $-Lx(k)$ but $-\Gamma(k)Lx(k)$. Secondly, since we have assumed the saturation levels to be bounded from below, this means that the elements in $-Lx(k)$ are bounded from above with $1/\gamma_{min,i}$. By pre- and post-multiplying (4.8a) with $Y = W^{-1}$, we can perform a Schur complement. From the second inequality, we know that $\gamma_i(k) \geq \gamma_{min,i}$, so the polytopic description of $\Gamma(k)$ described in Section 4.2 is applicable, hence we replace $\Gamma(k)$ by its polytopic approximation. The constraint (4.8a) is therefore equivalent to

$$\begin{bmatrix} Y & Y(A - B(\sum_{j=1}^{2^m} \lambda_j \Gamma_j)L)^T \\ (A - B(\sum_{j=1}^{2^m} \lambda_j \Gamma_j)L)Y & Y \end{bmatrix} \succeq 0 \tag{4.9}$$

which can be written as

$$\sum_{j=1}^{2^m} \lambda_j \begin{bmatrix} Y & Y(A - B\Gamma_j L)^T \\ (A - B\Gamma_j L)Y & Y \end{bmatrix} \succeq 0 \tag{4.10}$$

For the LMI above to hold for any admissible λ_j, all matrices in the sum have to be positive semidefinite. As before, we wish to maximize size($\mathcal{E}_W$), so we maximize $\det(W^{-1})$. Put together, we obtain the following optimization problem

$$\boxed{\begin{array}{ll} \max\limits_{Y,\gamma_{min}} & \det(Y) \\ \text{subject to} & \begin{bmatrix} Y & Y(A - B\Gamma_j L)^T \\ (A - B\Gamma_j L)Y & Y \end{bmatrix} \succeq 0 \\ & L_i Y L_i^T \leq \left(\frac{1}{\gamma_{min,i}}\right)^2 \end{array}} \tag{4.11}$$

Now, $\gamma_{min,i}$ are typically free variables as indicated in the optimization formulation. This will make the constraints above BMIs [28], bilinear matrix inequalities, since there are products of free variables in the constraints. Unfortunately, the BMIs make the optimization NP hard, so the globally optimal solution is difficult to find, although there exist methods, typically based on branch and bound schemes [27].

For fixed $\gamma_{min,i}$ however, we have a MAXDET problem. With such a structure, the optimization can be solved (local minimum) with various techniques, typically using some alternating method or linearization scheme. For $m \leq 2$, our problem is most easily solved by gridding in the variables $\gamma_{min,i}$. For the sake of clarity, a simple scheme based on linearizations is outlined in Appendix 4.A.

4.4 A Quadratic Bound on Achievable Performance

Solving the optimization problem (4.11) gives us the domain $\mathcal{X}$ in Theorem 3.1. To create $\Psi(x)$ in the same theorem, we chose a quadratic function $\Psi(x) = x^T Px$. Application of Assumption 4 in Theorem 3.1 yields

$$x^T(k+1)Px(k+1) - x^T(k)Px(k) \leq -x^T(k)Qx(k) - u^T(k)Ru(k) \tag{4.12}$$

By inserting the dynamics $x(k+1) = Ax(k) + Bu(k)$ and the saturated controller $u(k) = -\Gamma(k)Lx(k)$, we obtain the matrix inequality

$$(A - B\Gamma(k)L)^T P(A - B\Gamma(k)L) - P \preceq -Q - L^T\Gamma^T(k)R\Gamma(k)L \tag{4.13}$$

We pre- and post-multiply (4.13) with $Y = P^{-1}$, apply a Schur complement and use the polytopic description of $\Gamma(k)$ as in Equation (4.9) and obtain the LMI

$$\begin{bmatrix} Y & Y(A-B\Gamma_j L)^T & Y & YL^T\Gamma_j^T \\ (A-B\Gamma_j)Y & Y & 0 & 0 \\ Y & 0 & Q^{-1} & 0 \\ \Gamma_j LY & 0 & 0 & R^{-1} \end{bmatrix} \succeq 0 \tag{4.14}$$

Remark 4.2
In many applications, the matrix Q is not invertible. This occurs, e.g., when we only have a performance weight on some outputs. However, if we can write $Q = C^TC$, the Schur complement would instead yield the LMI

$$\begin{bmatrix} Y & Y(A-B\Gamma_j L)^T & YC^T & YL^T\Gamma_j^T \\ (A-B\Gamma_j)Y & Y & 0 & 0 \\ CY & 0 & I & 0 \\ \Gamma_j LY & 0 & 0 & R^{-1} \end{bmatrix} \succeq 0 \tag{4.15}$$

This is a remark that should be kept in mind for the remainder of this thesis.

Our goal is to create a tight bound, i.e., make $x^T Px$ as small as possible. One way to do this is to minimize the largest eigenvalue of P, or equivalently, maximize the smallest eigenvalue of Y. This gives us the following SDP

$$\boxed{\begin{array}{ll} \max\limits_{Y,t} & t \\ \text{subject to} & \begin{bmatrix} Y & Y(A-B\Gamma_j L)^T & Y & YL^T\Gamma_j^T \\ (A-B\Gamma_j)Y & Y & 0 & 0 \\ Y & 0 & Q^{-1} & 0 \\ \Gamma_j LY & 0 & 0 & R^{-1} \end{bmatrix} \succeq 0 \\ & Y \succeq tI \end{array}} \tag{4.16}$$

At this point, we have an upper bound $x^T(k)Px(k)$ of the achievable performance with a saturated linear feedback, together with an ellipsoidal domain $\mathcal{E}_W$ where this bound holds. This is all we need for our extension of a stabilizing MPC controller. However, we take the idea one step further.

4.5 A Piecewise Quadratic Bound

The upper bound we have derived can easily become very conservative (i.e. large). The reason is that if we have chosen saturation levels such that the system is close to unstable in some parts, the upper bound has to take these domains with extremely slow convergence into account.

To understand the idea behind the improved upper bound in this section, we begin with an intuitive example.

Example 4.2 (A one-dimensional system) *Consider the scalar unstable system*

$$x(k+1) = 2x(k) + u(k) \tag{4.17}$$

with the control constraint $|u(k)| \leq 1$. A nominal LQ controller is calculated with $Q = 1$ and $R = 1$, and gives the feedback matrix $L = 1.62$. The optimal cost in unsaturated mode is $x^T P_{LQ} x$ where $P_{LQ} = 4.24$. This bound holds for $|Lx| \leq 1$. We select a saturation level $\gamma_{min} = 0.65$, i.e., $\Gamma_1 = 0.65$ and $\Gamma_2 = 1$. Calculating the upper bound with (4.16) yields $P = 20.9$ and gives us the bound $x^T Px$ which holds when $|Lx| \leq 1/\gamma$, i.e., $|x| \leq 0.95$.
At $|Lx| = 1$ ($x = 0.62$), the unsaturated bound is applicable and yields $0.62^2 \cdot 4.24 = 1.62$. The upper bound, which must be used for $|Lx| > 1$, estimates the achievable performance to $0.62^2 \cdot 20.9 = 7.99$. Of course, the true cost can not be discontinuous. This is basically the property we will exploit in this section; translate the upper bound so that it coincides with the unsaturated bound at the break point $x = 0.62$. In Figure 4.3, the bound $x^T P_{LQ} x$ (valid in dark grey domain) and the translated

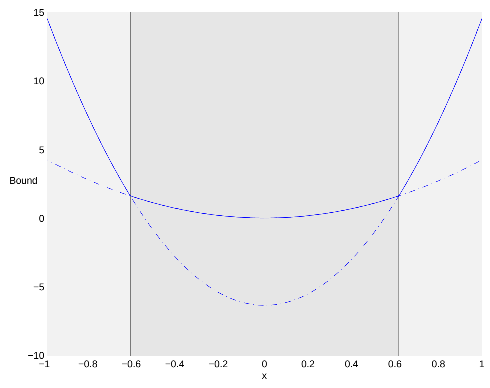

Figure 4.3: The upper bound is translated down to patch the two bounds together at $|x| = 0.62$. The resulting function can be described with a max selector.

upper bound $x^T Px - (7.99 - 1.62)$ (valid in light grey domain) are drawn with dash-dotted lines. A bound (solid line) valid for $|x| \leq 0.95$ can be written as $\max(x^T P_{LQ} x, x^T Px - (7.99 - 1.62))$.

As we saw in the example above, the main idea is to create multiple bounds by defining smaller sets contained in $\mathcal{E}_W$. Using these bounds, we define a less conservative bound using a piecewise quadratic function

$$\max_i \left(x^T P_i x - \rho_i\right) \tag{4.18}$$

The reason we are using this kind of function is that it turns out to be easy and efficient to use in an MPC algorithm.

Remark 4.3
It should be pointed out that piecewise quadratic functions of more general structure are available in the literature, see, e.g., [34]. Using those methods to estimate the cost would most likely give better bounds, but they do not seem to lend themselves to be incorporated in a convex MPC algorithm.

The derivation of the upper bound is a bit messy, so we first state the result.

Theorem 4.1 (Piecewise quadratic upper bound)
An upper bound of the achievable performance (4.1) when $x(k) \in \mathcal{E}_W$ is

$$J(k) \leq \max_i \left(x^T(k) P_i x(k) - \rho_i\right)$$

where the matrices P_i and scalars ρ_i are calculated with

$$\boxed{\begin{array}{rl} \underset{t,P,\rho}{\min} & t \\ \text{subject to} & (A - B\Gamma_{ji}L)^T P_i (A - B\Gamma_{ji}L) - P_i \preceq -Q - L^T \Gamma_{ji}^T R \Gamma_{ji} L \\ & (\rho_{i+1} - \rho_i) I \preceq \beta_i^{-1} W^{-1/2} \left(P_{i+1} - P_i\right) W^{-1/2} \\ & W^{-1/2} P_{n_s} W^{-1/2} \preceq (t + \rho_{n_s}) I \\ & \rho_1 = 0 \end{array}} \tag{4.19}$$

Variables and indexation are defined in the derivation below.

The idea is to create a number of nested ellipsoids $\mathcal{E}_{\beta W}$. $\beta \geq 1$ is a parameter that scales the ellipsoid, recall Definition 2.5. $\beta = 1$ corresponds to the original ellipsoid, and $\beta = \max_i(L_i W^{-1} L_i^T)$ corresponds to the case when the ellipsoid is scaled so that, for all i, $\gamma_i = 1$ in $\mathcal{E}_{\beta W}$, i.e., no saturation occurs. We define $n_s \geq 2$ ellipsoids

$$\mathcal{E}_{\beta_i W}, \quad i = 1 \dots n_s \tag{4.20}$$

with their scaling

$$\beta_1 = \max_i (L_i W^{-1} L_i^T), \quad i = 1 \dots m \tag{4.21a}$$

$$\beta_{i+1} < \beta_i, \quad i = 1 \dots n_s - 1 \tag{4.21b}$$

$$\beta_{n_s} = 1 \tag{4.21c}$$

For each ellipsoid the LMI (4.13) can be used to find an upper bound $x^T P_i x$. The difference in the ellipsoids is the saturation levels. We therefore have to define the matrices Γ_{ji}, where index i represents the different ellipsoids and j represents the vertices in the polytopic model. This gives us the first constraint in Theorem 4.1.

We now notice that the bound $x^T P_i x$ is derived using only a *growth* condition (Assumption 4 in Theorem 3.1). We can therefore subtract a scalar ρ_i. If we want our bound to be as small as possible, our goal is to make ρ_i as large as possible. In the innermost ellipsoid, $\mathcal{E}_{\beta_1 W}$, the bound $x^T P_1 x - \rho_1$ should be be used. Since the cost in the origin equals zero, we must have $\rho_1 = 0$. What about ρ_2? As soon as $x \in \mathcal{E}_{\beta_2 W} \setminus \mathcal{E}_{\beta_1 W}$ (outside the innermost ellipsoid but still in the second ellipsoid), the upper bound should use $x^T P_2 x - \rho_2$. This will be the case if

$$x^T P_2 x - \rho_2 \geq x^T P_1 x - \rho_1 \quad \text{when } x^T \beta_1 W x = 1$$

or, equivalently

$$x^T (P_2 - P_1) x \geq \rho_2 - \rho_1 \quad \text{when } x^T \beta_1 W x = 1 \tag{4.22}$$

To proceed, we need the following lemma

Lemma 4.2
The maximum and minimum of $x^T P x$ on the set $x^T W x = 1$ ($P \succ 0, W \succ 0$) is given by

$$\begin{aligned} \max x^T P x &= \lambda_{max}(W^{-1/2} P W^{-1/2}) \\ \min x^T P x &= \lambda_{min}(W^{-1/2} P W^{-1/2}) \end{aligned}$$

Proof With $z = W^{1/2} x$, the problem is to maximize (minimize) $z^T W^{-1/2} P W^{-1/2} z$ when $z^T z = 1$. The optimal choice of z is the normalized eigenvector corresponding to the largest (smallest) eigenvalue, and the lemma follows immediately. $\square$

By looking at the minimum of the left hand side of (4.22), Lemma 4.2 gives us

$$\beta_1^{-1} \lambda_{min} \left(W^{-1/2} \left(P_2 - P_1 \right) W^{-1/2} \right) \geq \rho_2 - \rho_1 \tag{4.23}$$

The same argument holds in the general case and results in the constraint

$$\beta_{i-1}^{-1} \lambda_{min} \left(W^{-1/2} \left(P_i - P_{i-1} \right) W^{-1/2} \right) \geq \rho_i - \rho_{i-1} \tag{4.24}$$

This is a linear convex constraint in P and ρ, equivalent to the second inequality in Theorem 4.1.

So far we have only introduced inequalities, but nothing to optimize. There are probably many choices, but one suitable term to minimize could be the worst case upper bound on the border of $\mathcal{E}_W$

$$\min_{P,\rho} \max_{x^T W x = 1} x^T P_{n_s} x - \rho_{n_s} \tag{4.25}$$

Once again we use Lemma 4.2 and obtain the objective

$$\min_{P,\rho} \lambda_{max}(W^{-1/2}P_{n_s}W^{-1/2}) - \rho_{n_s} \tag{4.26}$$

This gives us the last inequality in the theorem.

To see the benefits of this improved bound, we look at an example

Example 4.3 (Improved upper bound) *Consider a discretized double integrator (sample-time = 0.4s) with state-space realization (example from [21])*

$$A = \begin{bmatrix} 1 & 0 \\ 0.4 & 1 \end{bmatrix}, B = \begin{bmatrix} 0.4 \\ 0.08 \end{bmatrix}, C = \begin{bmatrix} 0 & 1 \end{bmatrix} \tag{4.27}$$

The nominal feedback $L = [1.90\ 1.15]$ is an LQ controller with $Q = I$ and $R = 0.25$. Solving (4.11) shows that stability can be guaranteed for $\gamma_{min} \geq 0.18$ and gives us an invariant ellipsoid $\mathcal{E}_W$

$$W = \begin{bmatrix} 0.157 & 0.047 \\ 0.047 & 0.057 \end{bmatrix} \tag{4.28}$$

An upper bound $x^T Px$ of the achievable performance is found with the SDP (4.16)

$$P = \begin{bmatrix} 133 & 37 \\ 37 & 52 \end{bmatrix} \tag{4.29}$$

An estimate, using this function, of the worst case cost when $x(k) \in \mathcal{E}_W$ is 991. We now chose to have three ellipsoids, $n_s = 3$, in order to create a piecewise quadratic upper bound. β_1 and β_3 are defined according to Equation (4.21), and β_2 was $= 0.9^{-2}$. This means that the middle ellipsoid $\mathcal{E}_{\beta_2 W}$ has a radius 0.9 times the radius of the largest ellipsoid $\mathcal{E}_{\beta_3 W} = \mathcal{E}_W$. Solving the optimization problem yields

$$\begin{aligned} P_1 &= \begin{bmatrix} 2.46 & 1.35 \\ 1.35 & 4.14 \end{bmatrix}, P_2 = \begin{bmatrix} 45.1 & 13.9 \\ 13.9 & 19.4 \end{bmatrix}, P_3 = \begin{bmatrix} 152.4 & 46.6 \\ 46.6 & 57.7 \end{bmatrix} \\ \rho_1 &= 0,\ \rho_2 = 8.67,\ \rho_3 = 553 \end{aligned} \tag{4.30}$$

The estimate of the worst case cost is now 476. However, much more important to notice is the optimal cost for the unsaturated system with the LQ controller

$$J_{LQ}(k) = x^T(k) \begin{bmatrix} 2.46 & 1.35 \\ 1.35 & 4.14 \end{bmatrix} x(k) \tag{4.31}$$

Close to the origin, our bound will be $x^T(k)P_1x(k)$, i.e., the bound is tight close to the origin. This implies, according to the discussion in Section 3.4.2, that an MPC controller using the piecewise quadratic upper bound will converge to an LQ controller as the origin is approached. This should be compared to the simple quadratic bound which will overestimate the unsaturated cost severely.

4.6 MPC Design

Our motivation for performance estimation of a saturated system has been to incorporate the estimate in an MPC algorithm. For simple notation, we define the terminal state weight

$$\Psi(x) \quad = \quad \max_i \left(x^T P_i x - \rho_i\right), \quad i = 1 \ldots n_s \tag{4.32}$$

and the cost to minimize in the MPC algorithm

$$J(k) = \sum_{j=k}^{k+N-1} x^T(j|k)Qx(j|k) + u^T(j|k)Ru(j|k) + \Psi(x(k+N|k))$$

The main result in this section is the following theorem

Theorem 4.2 (Stability of MPC algorithm)
The MPC controller is defined by calculating $u(k|k)$ with the following optimization

$$\boxed{\begin{array}{rl} \min\limits_u & J(k) \\ \textit{subject to} & |u(k+j|k)| \leq 1 \\ & x(k+N|k) \in \mathcal{E}_W \end{array}}$$

Asymptotic stability is guaranteed if the problem is feasible in the initial state.

Proof Follows from Theorem 3.1 and the construction of Ψ and $\mathcal{X}$. $\square$

4.6.1 SOCP Formulation of MPC Algorithm

The max selector in the terminal state weight can be efficiently implemented in the optimization problem. To do this, we notice that a max function can be rewritten using an epigraph [12] formulation.

$$\begin{array}{rl} \min\limits_x & \max\left(f_1(x), f_2(x)\right) \\ \Leftrightarrow & \\ \min\limits_x & t \\ \text{subject to} & f_1(x) \leq t, \quad f_2(x) \leq t \end{array} \tag{4.33}$$

By defining suitable matrices X, U, H, S, $\bar{Q}$ and $\bar{R}$ as in Section 3.3.1, the optimization problem in Theorem 4.2 can be written as

$$\boxed{\begin{array}{rl} \min\limits_{U,s,t} & s + t \\ \text{subject to} & |U| \leq 1 \\ & (Hx(k|k) + SU)^T \bar{Q} (Hx(k|k) + SU) + U^T \bar{R} U \leq s \\ & x^T(k+N|k) P_i x(k+N|k) - \rho_i \leq t \\ & x^T(k+N|k) W x(k+N|k) \leq 1 \end{array}} \tag{4.34}$$

The difference compared to a standard MPC algorithm is the quadratic inequalities, which force us to use second order cone programming, SOCP (see Definition 2.4), instead of QP. Whereas the quadratic terminal state constraint easily is written as a second order cone constraint,

$$||W^{1/2}x(k+N|k)|| \leq 1 \tag{4.35}$$

the other inequalities require more thought. However, straightforward calculations show that the following second order cone constraints are obtained

$$\left|\left|\begin{matrix} 2\bar{R}^{1/2}U \\ 2\bar{Q}^{1/2}\left(Hx(k|k)+SU\right) \\ 1-s \end{matrix}\right|\right| \leq 1+s, \quad \left|\left|\begin{matrix} 2P_i^{1/2}x(k+N|k) \\ 1-t-\rho_i \end{matrix}\right|\right| \leq 1+t+\rho_i, \tag{4.36}$$

Notice that introduction of SOCP always is needed when ellipsoidal terminal state constraints are used, and is not due to our extensions.

4.7 Examples

Now, let us look at some examples where we use the proposed approach to derive terminal state constraints and weights. We begin with a comparison of different terminal state constraints.

Example 4.4 (Comparison of terminal state constraints) *The main motivation for the work in this chapter has been to develop a method with less demanding terminal state constraints. We continue on the double integrator from Example 4.3 and calculate terminal state domains using four different methods.*

1. *Terminal state equality $x(k+N|k)=0$.*
2. *Terminal state inequality based on unsaturated LQ controller (Section 4.1).*
3. *Terminal state inequality based on optimal saturated LQ controller [21].*
4. *Terminal state inequality based on saturated controller (proposed).*

Each of these methods gives an ellipsoidal terminal state domain $\mathcal{E}_{W_i}$.

$$W_1 = \infty \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \; W_2 = \begin{bmatrix} 3.89 & 2.57 \\ 2.57 & 1.85 \end{bmatrix} \tag{4.37}$$

$$W_3 = \begin{bmatrix} 0.66 & 0.36 \\ 0.36 & 1.12 \end{bmatrix}, \; W_4 = \begin{bmatrix} 0.157 & 0.047 \\ 0.047 & 0.057 \end{bmatrix} \tag{4.38}$$

W_1 is a conceptual way to describe the terminal state equality as a terminal state inequality, W_2 was obtained by solving (4.5), W_3 is taken from [21] and W_4 was derived using (4.11).
The four ellipsoidal domains are illustrated in Figure 4.4 (of course, $\mathcal{E}_{W_1}$ degenerates to a point). We see that the proposed method gives a substantially larger terminal

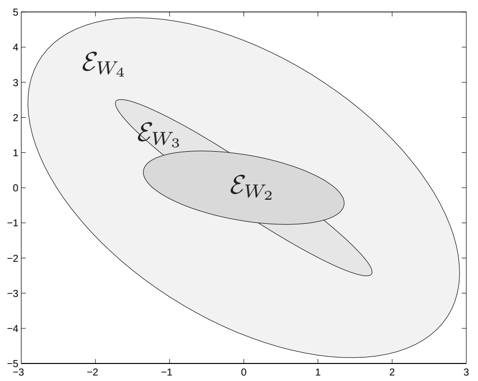

Figure 4.4: Terminal state domains.

state domain (the area of $\mathcal{E}_{W_4}$ is approximately 10 times larger than the area of $\mathcal{E}_{W_2}$ and $\mathcal{E}_{W_3}$). However, a more important property is how initial feasibility is affected. By using the algorithms developed in Chapter 6, we calculate the admissible initial set, i.e., all states for which terminal state constraint will be feasible. We select a prediction horizon $N = 5$. The results are shown in Figure 4.5. Although we have

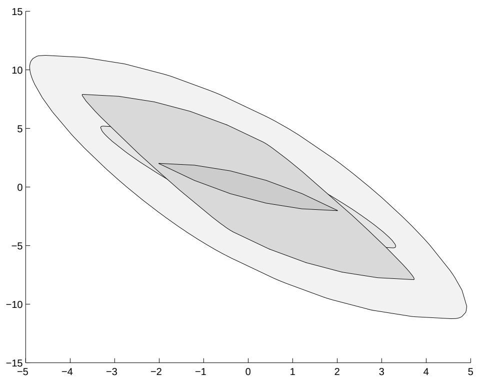

Figure 4.5: Initial set where terminal state constraint $x^T(k+N|k)W_ix(k+N|k) \leq 1$ will be feasible. The set in the bottom corresponds to W_4. The admissible initial set when using W_3 is to a large degree covered by the admissible set when W_2 is used. On the top, the admissible initial set when a terminal state equality is used.

managed to increase the size of the admissible initial set, the improvement is not as large as one would have hoped for, having the substantial enlargement of the terminal state domain in mind.

Of course, applicability of the approach depends on the system. We now look at a system where we obtain a substantial enlargement of the admissible initial set.

Example 4.5 *By discretizing the system*

$$G(s) = \frac{1}{s^2+1} \tag{4.39}$$

with a sample-time $1s$, *we obtain a state-space realization with*

$$A = \begin{bmatrix} 0.54 & -0.84 \\ 0.84 & 0.54 \end{bmatrix}, B = \begin{bmatrix} 0.84 \\ 0.46 \end{bmatrix} \tag{4.40}$$

An LQ controller is calculated ($Q = I$ *and* $R = 1$) *and gives us a nominal feedback matrix* $L = [0.72 \ \ -0.24]$. *By solving the SDP (4.11), we find* $\gamma_{min} = 0.011$ *and* W_4. *We also solve the optimization problem (4.5) and find an invariant domain for the unsaturated nominal controller.*

$$W_4 = 1e^{-4}\begin{bmatrix} 0.696 & 0.0065 \\ 0.0065 & 0.6961 \end{bmatrix}, \ W_2 = \begin{bmatrix} 0.68 & 0.065 \\ 0.065 & 0.4 \end{bmatrix} \tag{4.41}$$

Clearly, the terminal state domain defined using a saturated controller is magnitudes larger than one using the standard approach with an unsaturated nominal feedback. As we saw in the previous example, this does not have to mean that the initial admissible set is substantially larger, so we calculate these sets also and depict these in Figure 4.6. We see that the size of the admissible initial set is

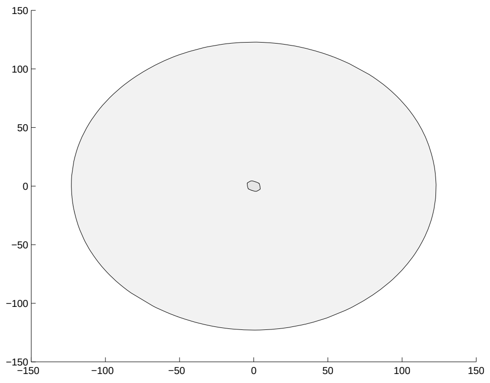

Figure 4.6: Initial set where terminal state constraint $x^T(k+N|k)W_i x(k+N|k) \leq 1$ will be feasible. The large, almost ellipsoidal, set corresponds to W_4. The admissible initial set when using W_2 is the small, hardly visible, set in the middle.

substantially improved.

We conclude this chapter with an example where we apply the proposed MPC algorithm.

Example 4.6 (Piecewise quadratic upper bound in MPC) *We apply the proposed MPC algorithm in Theorem 4.2 to the sampled double integrator from Example 4.3 and 4.4. The tuning variables are the same as in previous examples, i.e.,* $Q = I$, $R = 0.25$, $L = [1.90\ 1.15]$ *and* $N = 5$. *The terminal state domain and admissible initial set with this setup was calculated in Example 4.4. To create the piecewise quadratic terminal state weight, we divide the terminal state ellipsoid into 10 nested ellipsoid. The partition is made linearly, i.e.,* β_i *is linearly spaced between* β_1 *and* β_{10}. *The reason for choosing such a fine partitioning was to test how the optimization performs. In this example, partitioning the ellipsoid with, say, two inner ellipsoids gives the same performance.*
Besides the proposed algorithm we also implemented a standard MPC controller for comparison. The stabilizing controller from Section 3.4.2 would not be feasible from the initial conditions tested, so the terminal state constraint was simply neglected for this controller.
The system was simulated with initial conditions close to the boundary of the admissible initial set (shaded domain). Some typical trajectories are shown in Figure 4.7. For some initial states, the difference in behavior is substantial, whereas

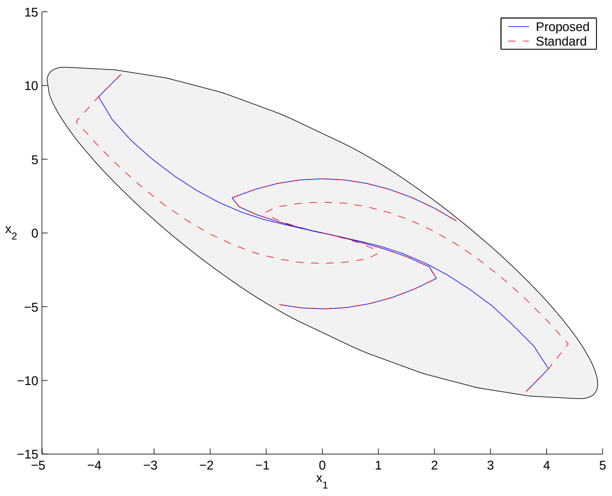

Figure 4.7: Typical closed loop trajectories. The initial states are chosen to lie close to the boundary of the admissible initial set (shaded domain). Trajectories resulting from the proposed MPC algorithm are shown in solid, whereas dashed lines correspond to a standard MPC algorithm.

the controllers perform almost identical from other initial states. By calculating the infinite horizon cost (3.3) for both controllers, it was noticed that the proposed controller always had a better performance. However, the difference in this example was small, typically below 10 percent.
Although the difference in performance was small, a plot of the output y *from*

the double integrator reveals that the difference in qualitative behavior can be significant. Figure 4.8 shows the output for the same trajectories as in 4.7. We see that the overshoot is considerably reduced (or even removed) for some initial states with the proposed controller.

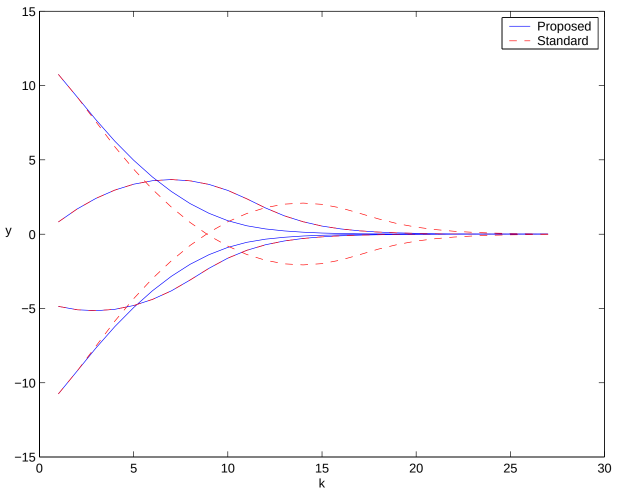

Figure 4.8: Output with different initial states and controllers. Clearly, the proposed MPC algorithm has reduced the overshoot for some initial states.

4.8 Conclusion

By using a polytopic model of a saturated controller, we derived an upper bound of the achievable performance with a saturated controller, and an invariant ellipsoidal domain where this bound holds. The bound was expressed in a special piecewise quadratic form that allowed it to be efficiently incorporated in an MPC algorithm.

We saw that for some examples, the improvement of the initial admissible set was substantial, compared to what was obtained using classical methods to define the terminal state constraint, while the improvement was less convincing for other examples. Although the results are highly problem dependent, we hope that the approach in some sense shows that there are other ways than using an unsaturated linear controller to define terminal state weights and constraints, a central part in many MPC algorithms with guaranteed stability. In fact, the results in this chapter have motivated the development of a similar approach which we will present in the next chapter.

Appendix

4.A Solution of BMI in Equation (4.11)

In order to find a locally optimal solution to the BMI in equation (4.11), a simple scheme based on linearization of the BMI can be used.

To begin with, we fix $\gamma_{min,i} = 1$ and solve the optimization problem (4.11). This gives us an ellipsoidal domain (i.e. the matrix Y) in which all control constraints are satisfied.

We now perturb the solution and define the perturbation on $\gamma_{min,i}$ to be δ_i and Δ on Y, i.e. $\hat{Y} = Y + \Delta$ and $\hat{\gamma}_{min,i} = \gamma_{min,i} + \delta_i$. Using Definition 4.1, this will implicitly define the perturbed matrices $\hat{\Gamma}_j$.

The first LMI in Equation (4.11) is linearized by inserting $\hat{Y}$ and $\hat{\Gamma}_j$, and neglecting any terms having products of the two perturbation. The right-hand side of the second LMI is also linearized and higher order terms of δ_i are neglected. Further on, we limit the size of the perturbations by introducing, say, $\delta_i^2 \leq 0.1\gamma_{min,i}^2$ and $\Delta^T\Delta \preceq 0.1Y^TY$. These constraint can be rewritten using Schur complements. Finally, $\hat{Y}$ must remain positive definite. Put together, we obtain

$$\boxed{\begin{aligned} \max \quad & \det(Y+\Delta) \\ \text{subject to} \quad & \begin{bmatrix} Y+\Delta & Y(A-B\hat{\Gamma}_jL)^T + \Delta(A-B\Gamma_jL)^T \\ * & Y+\Delta \end{bmatrix} \succeq 0 \\ & L_i(Y+\Delta)L_i^T \leq \left(\tfrac{1}{\gamma_{min,i}}\right)^2 - 2\tfrac{\delta_i}{(\gamma_{min,i})^3} \\ & \begin{bmatrix} 0.1\gamma_{min}^2 & \delta_i \\ \delta_i & 1 \end{bmatrix} \succeq 0 \\ & \begin{bmatrix} 0.1Y^TY & \Delta \\ \Delta & I \end{bmatrix} \succeq 0 \\ & Y+\Delta \succeq 0 \end{aligned}} \tag{4.A.42}$$

The star is used to save space, and means that the matrix is symmetric. With this linearization of the BMI, we can use the following algorithm

Algorithm 4.A.1 (BMI solver)

```
1.Find initial solution with $\gamma_{min,i} = 1 \Rightarrow Y$
2.Define the perturbations $\Delta$ and $\delta_i$
3.Solve the SDP (4.A.42)
4.Update solution, $Y := Y + \Delta$ and $\gamma_{min,i} = \gamma_{min,i} + \delta_i$
5.Repeat from step 2 until stopping criterion fulfilled.
```

Of course, this solver is local by nature, so a global optimal solution can not be guaranteed. However, from a practical point of view, the solution is reasonably good.

5

Stabilizing MPC using Performance Bounds from Switching Controllers

In this chapter, we use the main ideas from the previous chapter to derive a similar approach. The idea is to use a switching controller as the nominal controller in Theorem 3.1. By using pretty much the same ideas as in the previous chapter, we calculate a terminal state weight and terminal state domain to be used in MPC.

5.1 Switching Linear Feedback

A well studied approach to handle constraints is to use a switching controller, i.e., a controller that uses different feedback laws in different domains of the state-space. By tuning the feedback laws so that the constraints are guaranteed to hold in each domain, constraints are handled in a simple and heuristically attracting way.

The problem is to determine how to partition the state-space and select the feedback law in order to guarantee stability and obtain good performance. A simple choice is to use linear feedback laws in each of the domains. With this setup, we obtain a piecewise linear system. Although there exists a fair amount of theory for this class of systems, see [34] and references therein, we will only look at a special case.

5.2 Ellipsoidal Partitioning of the State-Space

A simple way to partition the state-space is to use nested ellipsoids, just as we did in Section 4.5. In the context of switching controllers, this was introduced in [67] and similar ideas have been used in, e.g, [21]. The idea is to attach nested positively invariant ellipsoids $\mathcal{E}_{W_j} \subset \mathcal{E}_{W_{j+1}}$ to feedback gains L_j. In each ellipsoid $\mathcal{E}_{W_j}$, the control law $-L_j x$ should satisfy the control constraints and stabilize the system. Whenever the state enters $\mathcal{E}_{W_j}$, the control law is switched to $-L_j x$. A formal stability proof is omitted, but stability is easy to realize intuitively. If the controller related to $\mathcal{E}_{W_j}$ is asymptotically stable, the state will in finite time reach $\mathcal{E}_{W_{j-1}}$. The same holds for this ellipsoid, and finally, after a finite number of switches, the state will reach the inner ellipsoid $\mathcal{E}_{W_1}$. Here, the controller is switched to the last feedback $-L_1 x$ which drives the state to the origin.

5.3 Piecewise LQ

Our underlying goal is to have a controller that minimizes a standard infinite horizon quadratic cost with weight matrices Q and R. As the set of feedback gains, we therefore create detuned versions of the desired controller. To do this, we introduce a sequence of n_s control weights,

$$R_{n_s} \succ R_{n_s-1} \ldots \succ R_1 = R \succ 0 \tag{5.1}$$

and calculate the different feedback gains L_j by solving the Riccati equation

$$(A - BL_j)^T P_{LQ,j}(A - BL_j) - P_{LQ,j} = -Q - L_j^T R_j L_j \tag{5.2}$$

For each of the different feedback gains, $P_{LQ,j}$ can be used to define a positively invariant ellipsoids. Notice that we use a slightly different notation in this chapter. The indexation L_j does not mean the jth row of L. Instead, it means the jth feedback matrix.

Example 5.1 (Nested ellipsoids) *Once again we look at the double integrator from Example 4.3. We have $Q = I$ and the desired control weight $R_1 = 0.25$. We also define detuned controllers with $R_2 = 2.5$ and $R_3 = 25$. This yields $L_1 =$* $[1.9\ 1.15]$, $L_2 = [1.10\ 0.49]$ *and* $L_3 = [0.62\ 0.18]$. *Each of the feedback gains will satisfy the control constraint inside a polyhedron in the state-space. The borders of these are drawn with solid lines in Figure 5.1. For each of the feedback gains, the level sets of $x^T P_{LQ,j} x$ define a positively invariant ellipsoid $\mathcal{E}_{W_j}$.*

5.4 Creating Nested Ellipsoids

In [67] and [21], the matrices $P_{LQ,j}$ are used to define the nested ellipsoids. An obvious improvement is to maximize the size of the invariant ellipsoids for each of the feedback gains, i.e., we find the largest positively invariant ellipsoid for each L_j,

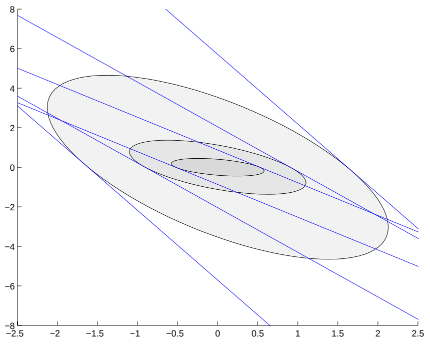

Figure 5.1: 3 nested positively invariant ellipsoids.
.

instead of using $P_{LQ,j}$. We define ellipsoids $\mathcal{E}_{W_j}$, and as in the previous chapter, we can find the largest possible positively invariant ellipsoids by introducing $Y_j = W_j^{-1}$ and solve a set of coupled MAXDET problems.

$$\boxed{\begin{array}{ll} \max\limits_{Y_j} & \det(Y_j) \\ \text{subject to} & \begin{bmatrix} Y_j & Y_j(A-BL_j)^T \\ (A-BL_j)Y_j & Y_j \end{bmatrix} \succeq 0 \\ & L_{j[i]}Y_jL_{j[i]}^T \leq 1 \\ & Y_j \prec Y_{j+1} \end{array}} \tag{5.3}$$

As before, the first inequality guarantees the ellipsoids to be positively invariant and the second takes care of the control constraints. Due to the slightly different notation in this chapter, we denote the ith row of the jth feedback matrix $L_{j[i]}$. The last inequality is introduced to assure that the ellipsoids are nested.

Example 5.2 (Nested ellipsoids, continued) *Once again we study the double integrator. We calculate the largest possible positively invariant ellipsoids and compare these to the ones based on $P_{LQ,j}$, calculated in the previous example. It is evident from Figure 5.2 that using $P_{LQ,j}$ to define the positively invariant ellipsoids can lead to conservative results.*

5.5 Estimating the Achievable Performance

What we need for MPC is an upper bound of the achievable performance when the switching controller is used. As usual, with a fixed feedback gain, the quadratic bound x^TP_jx holds for initial states in $\mathcal{E}_{W_j}$ if

$$(A-BL_j)^TP_j(A-BL_j) - P_j \preceq -Q - L_j^TRL_j \tag{5.4}$$

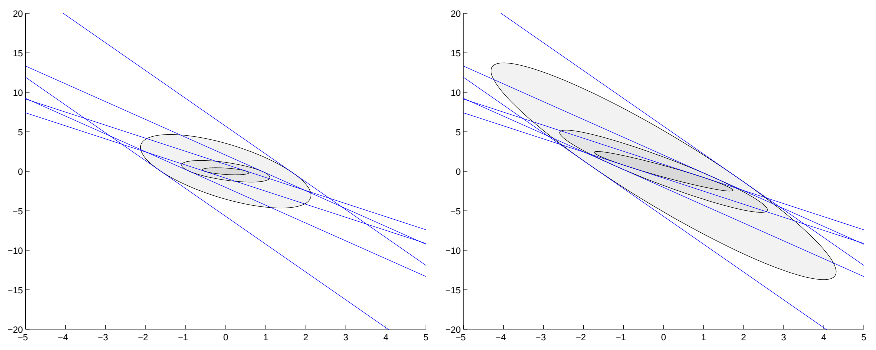

Figure 5.2: To the left, the ellipsoids are defined using $P_{LQ,j}$. To the right, they have been optimized in order to make the domains as large as possible. Clearly, the size of the new ellipsoids are substantially larger, hence the switching controller can be applied for a larger set of initial states.

.

We now exploit the same fact as in Chapter 4, i.e., the condition above is only a growth condition. Using the same motivation and derivation as in Theorem 4.1, we obtain the following SDP

$$\boxed{\begin{array}{rl} \min\limits_{t,P,\rho} & t \\ \text{subject to} & (A-BL_j)^T P_j (A-BL_j) - P_j \preceq -Q - L_j^T R L_j \\ & (\rho_{j+1}-\rho_j)I \preceq W_j^{-1/2}\left(P_{j+1}-P_j\right)W_j^{-1/2} \\ & W_{n_s}^{-1/2} P_{n_s} W_{n_s}^{-1/2} \preceq (t+\rho_{n_s})I \\ & \rho_1 = 0 \end{array}} \tag{5.5}$$

After solving this SDP, the terminal state weight $\max_j \left(x^T P_j x - \rho_j\right)$ and the terminal state constraint $x^T(k+N|k)W_{n_s}x(k+N|k) \leq 1$ can be used in an MPC algorithm as in Theorem 4.2.

An MPC controller with connections to this approach was reported in [16, 17]. In these contributions, the terminal state weight and terminal state domain were chosen on-line. The idea was to find a sufficiently detuned LQ controller at each sample-instant. The LQ feedback is picked so that a terminal state constraint, defined using P_{LQ} as in Section 3.4.2, would be feasible with the old control sequence. By using $x^T P_{LQ} x$ as a terminal state weight together with the terminal state constraint, stability can be proven by standard arguments. Our approach can be seen as a generalization, with the difference that the explicit search for a suitably detuned controller is done automatically in the optimization. Furthermore, since we use different matrices to define the terminal state weight and constraint, we obtain improved feasibility.

Example 5.3 *We continue on the double integrator example and apply the proposed MPC algorithm. We pick a sequence of five feedback gains, calculated with the control weights $R = R_1 = 0.25$, $R_2 = 2.5$, $R_3 = 25$, $R_4 = 250$ and $R_5 = 2500$. As in the MPC example in the previous chapter, we also implemented a standard MPC algorithm with the terminal state weight chosen as the optimal cost for the unconstrained problem, $x^T P_{LQ,1} x$.*
The admissible initial set was calculated and used to select suitable initial states in some simulations, see Figure 5.3. The result is the same as in the previous chapter,

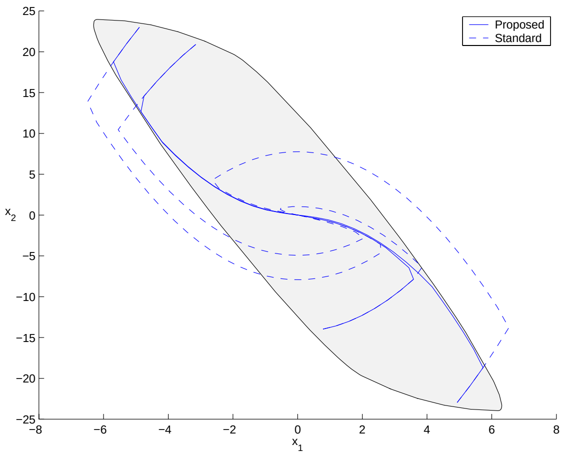

Figure 5.3: Closed loop trajectories. The initial states are chosen to lie close to the boundary of the admissible initial set (shaded domain). Trajectories resulting from the proposed MPC algorithm are shown in solid, whereas dashed lines correspond to a standard MPC algorithm.

i.e., for some initial states, there is a major improvement of the qualitative behavior.

5.6 Conclusions

By a straightforward extensions of the piecewise quadratic bound in Chapter 4, we were able to estimate the achievable performance of a switching controller, and use this in an MPC controller. The derived MPC controller turned out to have close connections to an approach previously reported in the literature. In this related approach, the main idea was to, at each sample-instant, explicitely search for a suitable terminal state constraint and terminal state weight. This search is incorporated into the optimization problem in our proposed MPC controller. We also showed that we are able to improve initial feasibility, compared to the related approach.

As an intermediate result, we also showed how the classical approach to define invariant nested ellipsoids for switching controllers easily could be improved upon.

6

Feasibility with Ellipsoidal Terminal State Constraints

As we have seen in the preceding chapters, a common concept in MPC strategies with guaranteed stability is to add ellipsoidal terminal state constraints to the optimization. For the stability results to hold, the first line in the stability theorems often read "*Assume that the problem is feasible for the initial state at time 0...*". In this chapter we will study the problem of finding the initial states for which the optimization will be feasible when an ellipsoidal terminal state constraint is used. We denote this the *admissible initial set*. The problem is interesting since it can help us to determine the practical impact of a terminal state constraint. Furthermore, when performing simulations as in the previous chapter, it is possible to select initial conditions from which the problem is close to infeasible, i.e., we will have a possibility to test demanding initial conditions.

There does not seem to be much done in the MPC literature on this issue, but previous results are available for the similar topic reachability analysis for constrained systems, see [9] and references therein. We will present two methods to do the feasibility analysis, i.e., estimate the admissible initial set. The first approach is a variant of the work in [29], while the second is derived with more knowledge of the underlying structure in order to get better estimates. By using this new method to calculate the admissible initial set, we will actually be able to get an exact geometrical characterization of the set. In both algorithms we present, the final goal will however be to find an ellipsoidal approximation of the admissible initial set.

6.1 Problem Formulation

The underlying optimization problem we study can in a general form be stated as

$$\begin{aligned} \min_{u} \quad & G(x(k),u) \\ \text{subject to} \quad & u(k+j|k) \in \mathcal{U} \\ & x(k+j+1|k) = Ax(k+j|k) + Bu(k+j|k) \\ & x^T(k+N|k)Wx(k+N|k) \leq 1, \quad W \succ 0 \end{aligned} \tag{6.1}$$

Notice that the objective is entirely irrelevant, thus the loose description $G(x(k),u)$. The control constraint is assumed to be a polytope, i.e., all control inputs are constrained. For simple notation, we assume $W = I$. The general case can be dealt with by first performing a coordinate transformation.

The problem we want to solve in this chapter is to find the largest possible set $\mathcal{I}$ with the property that if $x(k) \in \mathcal{I}$, then the optimization problem above is feasible. We call the set $\mathcal{I}$ the admissible initial set. When the derived set is an approximation of the true admissible initial set, we will emphasize this by denoting the found set $\hat{\mathcal{I}}$. We will present two methods to find $\hat{\mathcal{I}}$. In both methods, the algorithms first find a polytopic estimate of $\mathcal{I}$, and then approximate this with the largest possible ellipsoid inscribed in the polytope.

6.2 Some Mathematical Preliminaries

Some mathematical concepts are needed before we present the main results. To begin with, we introduce the vertices of a polytope, $\mathbf{V}(\cdot)$. It can be shown, see, e.g., [71], that a polytope can be defined as the convex hull (see Definition 2.8) of a number of vertices, i.e. $\mathcal{U} = \mathbf{Co}\,(\mathbf{V}(\mathcal{U}))$. Another notation is $\mathbf{F}(\cdot)$ which denotes the facets of a polytope [71]. Finally, the notation $\partial(\cdot)$ means the boundary of a set. In the case of a polytope, this corresponds to the collection of all facets.

Example 6.1 (Polytopes: vertices and facets) *As an example, we study a case where we have two control inputs and the constraint $|u| \leq 1$.*
The control constraint polytope can be written as

$$\mathcal{U} = \{u \ : \ \begin{bmatrix}1 & 0\end{bmatrix} u \leq 1, \begin{bmatrix}0 & 1\end{bmatrix} u \leq 1, \begin{bmatrix}-1 & 0\end{bmatrix} u \leq 1, \begin{bmatrix}0 & -1\end{bmatrix} u \leq 1\}$$

The vertices of this polytope are

$$\mathbf{V}(\mathcal{U}) = \{\begin{bmatrix}1\\1\end{bmatrix}, \begin{bmatrix}-1\\1\end{bmatrix}, \begin{bmatrix}1\\-1\end{bmatrix}, \begin{bmatrix}-1\\-1\end{bmatrix}\}$$

and the facets are the sets

$$\begin{aligned} \boldsymbol{F}(\mathcal{U}) = \{&\{u \ : \ u_1 = 1, |u_2| \leq 1\}, \{u \ : \ u_1 = -1, |u_2| \leq 1\}, \\ &\{u \ : \ u_2 = 1, |u_1| \leq 1\}, \{u \ : \ u_2 = -1, |u_1| \leq 1\}\} \end{aligned}$$

6.2.1 Convex Hull Calculations

The algorithms we will present use convex hull calculations. The classical problem of finding the convex hull is to, given a set of points $\{x_i \in \mathbf{R}^n\}$, find the set of points which defines the smallest possible polytope containing all the original points. In our formalism, finding $\mathbf{V}(\mathbf{Co}(\{x_i\}))$. There are many algorithms available to solve the convex hull problem. In our implementations, we have used the freely available program QHULL [1, 24].

Example 6.2 (Convex hull calculations) *As an example, we generate 40 random points in two dimensions. By using* QHULL *[24], we find that 8 points, connected by lines in the figure below, are needed to define the convex hull.*

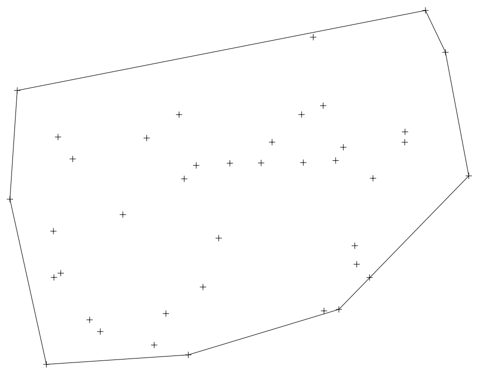

Figure 6.1: Convex hull of random points in $\mathbf{R}^2$.

6.3 Approximate Method

The first method we propose is a direct application of the results in [29]. We state and use the conceptual ideas of this work, but for a more detailed analysis the reader is referred to [29]. The central idea is to use an inverse model of the system

$$x(k-1) = A^{-1}x(k) - A^{-1}Bu(k-1) \tag{6.2}$$

Notice that this requires A to be invertible. The problem analyzed in [29] can be viewed as the problem to find the set of initial conditions $x(k)$, for which it is possible to reach $x(k+N) = 0$, and the idea is to calculate backwards: Given $x(k+N) = 0$, what values can $x(k+N-1)$ take? By using the fact that any admissible choice of $u(k-1)$ lies in the control constraint polytope, all admissible

$x(k+N-1)$ must satisfy

$$\begin{aligned} x(k+N-1) &= A^{-1}x(k+N) - A^{-1}Bu(k+N-1) \\ &= -A^{-1}Bu(k+N-1) \\ &\in -A^{-1}B\mathbf{Co}\left(\mathbf{V}(\mathcal{U})\right) \\ &= \mathbf{Co}\left(-A^{-1}B\mathbf{V}(\mathcal{U})\right) \end{aligned} \tag{6.3}$$

Hence, the admissible states $x(k+N-1)$ lie in the polytope $\mathbf{Co}\left(-A^{-1}B\mathbf{V}(\mathcal{U})\right)$, i.e., in the polytope with vertices $\mathbf{V}(\mathbf{Co}\left(-A^{-1}B\mathbf{V}(\mathcal{U})\right))$. The idea is now iterated, given the set $x(k+N-1)$, we calculate the admissible $x(k+N-2)$. The algorithm can be summarized as

Algorithm 6.1 (Initial States which can reach the origin)

```
1. Let $\mathcal{X} = \{x_i\} = \{0\}$
2. Find the reachable points, $r_{ij} = \{A^{-1}x_i - A^{-1}Bu_j \quad \forall x_i \in \mathcal{X},\ u_j \in \mathbf{V}(\mathcal{U})\}$
3. Define the new initial points $\mathcal{X}=\mathbf{V}(\mathbf{Co}\left(\{r_{ij}\}\right))$
4. Repeat from the second step $N-1$ times
```

In order to use this algorithm in our application, we do not have to change much. The only difference is the first step. We do not have the constraint $x(k+N) = 0$, instead we have an ellipsoidal constraint. To solve this, a simple approximation is to create an inner polytopic approximation of the constraint ellipsoid. To do this, we generate a set of initial points $\mathcal{X} = \{x_i\}$ with the property $x_i^T x_i = 1$. These points describe a polytope inscribed inside the terminal ellipsoid.

Algorithm 6.2 (Approximation of admissible initial set)

```
1. Let $\mathcal{X} = \{x_i\},\ x_i^T x_i = 1, i = 1 \ldots k$
2. Find the reachable points, $r_{ij} = \{A^{-1}x_i - A^{-1}Bu_j \quad \forall x_i \in \mathcal{X},\ u_j \in \mathbf{V}(\mathcal{U})\}$
3. Define the new initial points $\mathcal{X} = \mathbf{V}(\mathbf{Co}\left(\{r_{ij}\}\right))$
4. Repeat from the second step $N-1$ times
```

Clearly, if we can reach the inner polytopic approximation of the terminal ellipsoid, we can reach the ellipsoid.

The problem arising here is the choice of which, and how many, points to select in the initial step. In the implementation for this work, we have used randomly distributed points. The effects of the random approximation will be discussed later in the examples. The result from these calculations will be a set of points $\mathcal{X}$. These points describe a polytope $\hat{\mathcal{I}}$ defined by a set of linear inequalities

$$c_i^T x \leq d_i \tag{6.4}$$

All points satisfying these inequalities are admissible initial states. The counterpart does not hold. Due to the approximation of the terminal ellipsoid, the set is not guaranteed to be maximal, hence the notation $\hat{\mathcal{I}}$

6.4 Exact Calculation of Admissible Initial Set

The second approach we present is a slightly more complex method. It does however have some advantages to Algorithm 6.2. First, this method does not assume A to be invertible. Secondly, it will allow us to get an exact representation of the admissible initial set. Another important feature is that when we create ellipsoidal approximations of the admissible initial set, the approximation can often be made better compared to the result with the approximative method introduced in the previous section.

The method can be divided into two steps. The first step is to find the reachable set from $x(k) = 0$. This can be done almost exactly as in Algorithm 6.1, but we can now do the calculations in the "correct" direction, instead of backwards.

Algorithm 6.3 (Reachable set from origin)

```
1.Let $\mathcal{X} = \{x_i\} = \{0\}$
2.Find the reachable points, $r_{ij} = \{Ax_i + Bu_j \quad \forall x_i \in \mathcal{X},\ u_j \in \mathbf{V}(\mathcal{U})\}$
3.Define the new initial points $\mathcal{X} = \mathbf{V}(\mathbf{Co}(\{r_{ij}\}))$
4.Repeat from second step $N - 1$ times
```

After the last iteration, $\mathcal{R} = \mathbf{Co}(\mathcal{X})$ describes the reachable set in N steps from $x(k) = 0$ with the constrained control.

Example 6.3 (Reachable set from origin) *Let us study a simple example where*

$$A = \begin{bmatrix} 1.14 & -1.11 \\ 1 & 0 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

The algorithm above is applied in order to find the states which we can reach in one, two and three steps with the constrained control $|u(k)| \leq 1$.

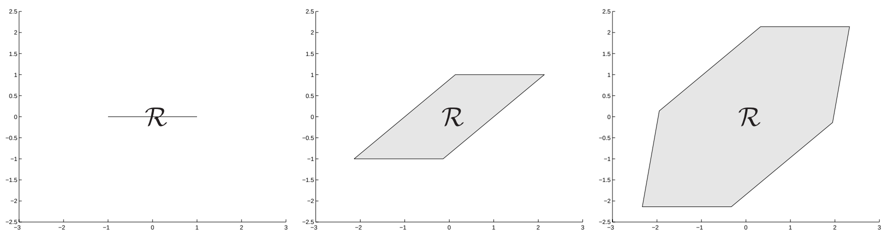

Figure 6.2: Reachable set from the origin in one (left), two (middle) and three (right) steps.

To be able to describe the second part of the algorithm more clearly, it is beneficial to use the following description of the terminal state

$$x(k+N) = A^N x(k) + \sum_{i=0}^{N-1} A^{N-1-i} B u(k+i) \tag{6.5}$$

We notice that in Algorithm 6.3, we have calculated the reachable set for the summation part. We introduce the auxiliary variable

$$w = A^N x(k) \tag{6.6}$$

Furthermore, we call the terminal ellipsoid (which actually is a sphere due to the initial scaling) $\mathcal{T}$.

6.4.1 Geometrical Characterization of Admissible Initial Set

We now go back to the original problem: For which states $x(k)$ is the ellipsoid $x^T(k+N)x(k+N) \leq 1$ reachable with the constrained control? To begin with, we notice that according to (6.5) and (6.6), all reachable points can be written as $w + v,\ v \in \mathcal{R}$. Hence, the admissible states are, by definition, those w for which there exist a $v \in \mathcal{R}$ such that $w + v \in \mathcal{T}$. The interesting points are those which just barely can reach $\mathcal{T}$, so we look at the case $w + v \in \partial(\mathcal{T})$, where $\partial(\mathcal{T})$ denotes the boundary of $\mathcal{T}$. The way to find these w is most easily described with a two-dimensional example, see Figure 6.3 and 6.4. For each point z on the boundary of

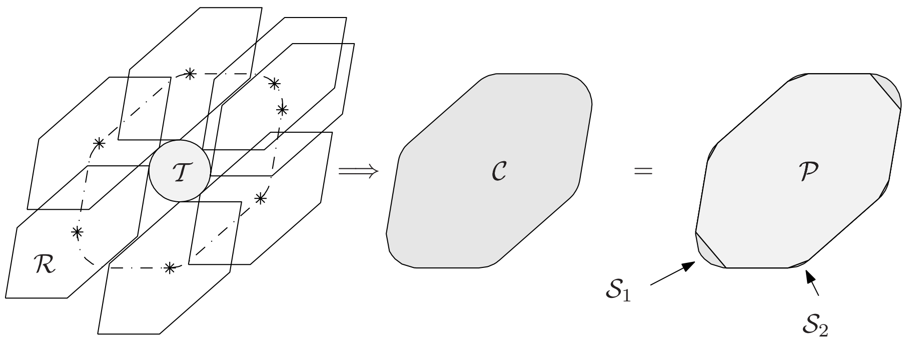

Figure 6.3: Generating the set $\mathcal{C}$ by revolving the reachable box $\mathcal{R}$ around the terminal ellipsoid $\mathcal{T}$. The set $\mathcal{C}$ can constructed as the convex hull of a number of translated ellipsoids $\mathcal{S}_i$ (in the corners, to a large degree covered by the inner polytopic approximation $\mathcal{P}$).

the terminal ellipsoid $\mathcal{T}$ we place the origin of $\mathcal{R}$ at a position w such that $\partial(\mathcal{R})$ is in contact with $\partial(\mathcal{T})$ in z. This is done for all points (infinitely many) on $\partial(\mathcal{T})$. As we see in Figure 6.3, this corresponds to revolving $\mathcal{R}$ around $\partial(\mathcal{T})$. We let $\mathcal{C}$

denote the convex hull of all points w. By construction, $\mathcal{C}$ will contain all w for which there exist a $v \in \mathcal{R}$ such that $w + v \in \mathcal{T}$

The important idea now is that we do not have to calculate these points for all (infinitely many) points on $\partial(\mathcal{T})$. There are two different cases to take into account. The first is if a vertex of $\mathcal{R}$ is in contact with the ellipsoid. If we move around the vertex locally, and the vertex is kept in contact with the ellipsoid, the origin of $\mathcal{R}$ will describe a segment of the boundary of the terminal ellipsoid, translated away from the origin. The second case is if a higher dimensional manifold (compared to a vertex) of $\partial(\mathcal{R})$ (such as a facet or intersection of facets) is in contact with $\partial(\mathcal{T})$, i.e., tangent to the ellipsoid. If $\mathcal{R}$ is moved locally in the space spanned by the manifold around this point (for example, along a line for the intersection of three-dimensional facets), but kept in contact with the ellipsoid, the origin of $\mathcal{R}$ will describe a planar motion. Since we are moving between the vertices when we move along these manifolds, the surfaces generated by these planar motions will patch the ellipsoidal segments together linearly. The complete set can therefore be seen as the convex hull of the ellipsoidal segments. In Figure 6.3, we see that there are 6 ellipsoidal segments, patched together with 6 lines. Notice that the surfaces defined by the planar motions (i.e., the straight lines patching the ellipsoids together) define an inner polytopic approximation of the set.

6.4.2 The Polytopic Part

We start by finding the polytopic part of $\mathcal{C}$, i.e., the light shaded part in Figure 6.3. Let us denote this polytope $\mathcal{P}$. Each facet of $\mathcal{R}$, $\mathbf{F}_j(\mathcal{R})$, will generate a facet of $\mathcal{P}$, $\mathbf{F}_j(\mathcal{P})$. To be more precise, $\mathbf{F}_j(\mathcal{P})$ will be defined by the vertices of the facet $\mathbf{F}_j(\mathcal{R})$. The calculation of this is most easily described in a two-dimensional example.

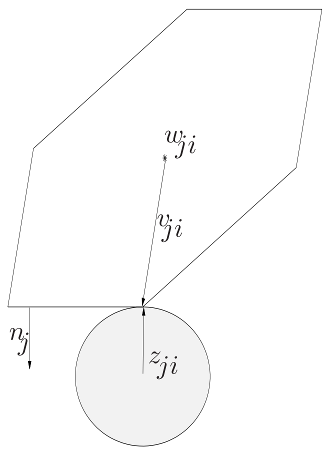

Figure 6.4: Generating the facets of $\mathcal{P}$ by finding the points generated by the facets of $\mathcal{R}$.

We want to find the facet generated by $\mathbf{F}_j(\mathcal{R})$ with the normal n_j. We assume that n is normalized, $n_j^T n_j = 1$, and is pointing outwards. Let the vertices of $\mathbf{F}_j(\mathcal{R})$ be v_{ji} and let w_{ji} denote the generated vertices of $\mathbf{F}_j(\mathcal{P})$ which we want to find. The important points are when a point v_{ji} is on the boundary of the ellipsoid, and the facet $F_j(\mathcal{R})$ is tangent to the ellipsoid, see Figure 6.4. These points will define $\mathbf{F}_j(\mathcal{P})$.

$\mathbf{F}_j(\mathcal{R})$ will be tangent to the ellipsoid when the normal of the ellipsoid is parallel to the normal of the facet. Hence, the tangent point z_{ji} on $\mathcal{T}$ will be $-n_j$ (the ellipsoid is a normalized unit ball). We then directly obtain that the origin w_{ji} of $\mathcal{R}$ will be placed at $-n_j - v_{ji}$. The points

$$\{-n_j - v_{ji}\} \tag{6.7}$$

will thus define the facet $\mathbf{F}_j(\mathcal{P})$ generated by $\mathbf{F}_j(\mathcal{R})$.

6.4.3 The Ellipsoidal Part

To be able to create the ellipsoidal segments, the following theorem is needed

Theorem 6.1 (Sphere center given points on boundary)
Given n points $\{x_i \in \mathbf{R}^n\}$, the center x_c of the sphere $(x-x_c)^T(x-x_c)$, for which it holds that $(x_i - x_c)^T(x_i - x_c) = 1$, can be calculated as

$$\begin{aligned} x_c &= b \pm \alpha c \\ \alpha &= \sqrt{1 - ||x_1 - b||^2} \end{aligned}$$

where b is the Euclidean center of the points $\{x_i\}$ and c is the normalized normal to the plane spanned by $\{x_i\}$.

Proof Let $\mathcal{P}$ denote the plane spanned by $\{x_i\}$ and c the normalized normal for this plane. The plane and the normal will span $\mathbf{R}^n$, so we can write $x_c = b + \alpha c$ where α is an unknown scalar and $b \in \mathcal{P}$. If x_c is the center of the sphere, the definition of a sphere and x_c gives us

$$\begin{aligned} (x_i - x_c)^T(x_i - x_c) &= 1 \Leftrightarrow \\ (x_i - (b + \alpha c))^T(x_i - (b + \alpha c)) &= 1 \Leftrightarrow \\ (x_i - b)^T(x_i - b) - 2\alpha c^T(x_i - b) + \alpha^2 c^T c &= 1 \end{aligned} \tag{6.8}$$

Since $c^T c = 1$ and $(x_i - b) \perp c$ we have

$$(x_i - b)^T(x_i - b) + \alpha^2 = 1$$

For a solution to exist, b must be the euclidian center of the points $\{x_i\}$, i.e.

$$(x_i - b)^T(x_i - b) = (x_j - b)^T(x_j - b)$$

This constraint on b, together with the orthogonality constraint $(x_i - b) \perp c$ will give a linear equation to find b. □

The ellipsoidal segments will be generated when vertices of $\mathcal{R}$ are moved onto the boundary of the terminal ellipsoid. A vertex v_i is a vertex in a number of facets $\mathbf{F}_{ji}(\mathcal{R})$ with normals n_{ji}. What we need to define the ellipsoidal segments is a collection of points on the generated segment. By using the theorem above, we can use these to find the position of the translated ellipsoid. Natural points to pick are the points when v_i lies on the terminal ellipsoid, and $\mathbf{F}_{ji}(\mathcal{R})$ is tangent to the ellipsoid. These points can be derived as in the derivation of $\mathcal{P}$ and will be

$$\{-n_{ji} - v_i\} \tag{6.9}$$

Given these points, the center of the translated sphere can be calculated. We denote these points $x_{c,i}$. This will give us a collection of spheres $\mathcal{S}_i$.

6.4.4 Complete Characterization

We can now construct our complete characterization of the admissible initial set

$$\mathcal{I} = \{x(k) \ : \ w \in \mathbf{Co}\left(\{\mathcal{S}_i\}\right)\} \tag{6.10}$$

No approximations have been done in the derivation of this set, so it is an exact definition of the set. It is a characterization in w, but according to earlier, w is just a linear transformation of $x(k)$.

In the next section, we will make an ellipsoidal approximation of this set, and the set derived with Algorithm 6.2. To do this, we must have a polytopic description of our admissible initial set. This means we have to approximate the segments of the spheres $\mathcal{S}_i$. A simple way to this is to create points x_{ij} on the boundary of $\mathcal{S}_i$, with the restriction that they lie outside the inner polytopic approximation $\mathcal{P}$. We then create the polytope $\hat{\mathcal{I}} = \mathbf{Co}\left(\{x_{ij}, V(\mathcal{P})\}\right)$

We summarize the procedure with the following algorithm

Algorithm 6.4 (Approximation of admissible initial set)

1. *Calculate the set $\mathcal{R}$ with Algorithm 6.3.*
2. *Calculate the polytope $\mathcal{P}$ by picking the vertices (6.7)*
3. *Generate the translated spheres $\mathcal{S}_j$ by picking the points (6.9) and using Theorem 6.1.*
4. *Approximate the spheres by introducing r random points $x_{ij} \in \partial(\mathcal{S}_i) \setminus \mathcal{P}$ for each sphere $\mathcal{S}_i$*
5. *Let $\hat{\mathcal{I}} = \mathbf{Co}\left(\{x_{ij}, \mathbf{V}(\mathcal{P})\}\right)$*

6.5 Ellipsoidal Approximation of Admissible Initial Set

In both methods above, the resulting approximation $\hat{\mathcal{I}}$ of the admissible initial set $\mathcal{I}$ is a polytope defined by some linear inequalities $c_i^T x \leq d_i$ In order to get a more compact description of this set (the number of inequalities rapidly grow large), we want to have an ellipsoidal description instead, i.e. find an ellipsoidal approximation of the admissible initial set.

$$\hat{\mathcal{I}} = \{x(k) \ : \ x^T(k)Px(k) \leq 1\} \tag{6.11}$$

To find such an ellipsoid, we look for the largest possible ellipsoid contained in the polytope $\hat{\mathcal{I}}$ defined by the linear inequalities. By introducing $Y = P^{-1}$ and using Lemma 4.1, we see that this is a MAXDET problem

$$\boxed{\begin{array}{rl} \max\limits_{Y} & \det(Y) \\ \text{subject to} & c_i^T Y c_i \leq d_i^2 \end{array}} \tag{6.12}$$

6.6 Examples

We study the two algorithms by applying them to two examples.

Example 6.4 (Third order unstable system) *The system we analyze is given by the zero-order hold sampled version of*

$$G(s) = \frac{1}{s^3 - 0.1s^2 + s} \tag{6.13}$$

The system is sampled with a sample-time 1 second, the constraint on the control input is $|u(k)| \leq 1$, the prediction horizon $N = 5$, and the terminal constraint is

$$x^T(k+5)x(k+5) \leq 1 \tag{6.14}$$

Both of the proposed algorithms have a step where a randomized approximation is performed. In Algorithm 6.2, the terminal state ellipsoid is approximated as a polytope with vertices generated randomly, and in Algorithm 6.4, the exact admissible initial set is approximated as a polytope by introducing random points to remove the ellipsoidal segments. To see the impact of these approximations, we study some different settings. In Algorithm 6.2, the randomization in the first step is performed with the following number of random points

$$\begin{bmatrix} 25 & 50 & 75 & 100 & 200 & 400 \end{bmatrix} \tag{6.15}$$

and in Algorithm 6.4 we pick

$$\begin{bmatrix} 1 & 2 & 5 & 10 & 20 & 30 \end{bmatrix} \tag{6.16}$$

points to approximate each ellipsoidal segment. Notice that the numbers not are comparable.
In order to compare the two algorithms, we plot the volume of the resulting ellipsoidal approximation as a function of the CPU-time spent[1] *(which depends on the number of random points introduced). For each setting, the calculations were repeated 10 times to see the effect of the realizations. It is quite evident that*

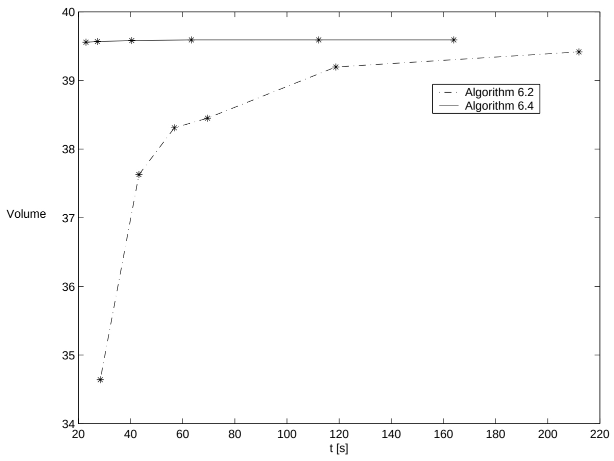

Figure 6.5: Average CPU-time used for the different algorithms and the average volume of the estimated admissible initial set.

for this example, Algorithm 6.4 outperforms Algorithm 6.2 in terms of CPU-time needed for a reasonably good estimate. If more CPU-time is allowed, the difference between the two algorithms is reduced.
Since the algorithms are based on a randomized approximation, it is interesting to see how sensitive they are to different realizations. As we can see in the table below, Algorithm 6.4 is much less sensitive.

	Variance of estimated volume (test case)					
	#1	*#2*	*#3*	*#4*	*#5*	*#6*
Algorithm 6.2	*8.54*	*0.55*	*0.31*	*0.20*	*0.07*	*0.01*
Algorithm 6.4	*0*	*8e-5*	*15e-5*	*1e-6*	*0*	*0*

It should be mentioned that the code for Algorithm 6.2 is very simple, and is therefore optimized, whereas the code for algorithm for Algorithm 6.4 currently is implemented without taking speed into account. Furthermore, a large part of

[1]In MATLAB™using MAXDET [69] and qhull [24] on Sun ULTRA SPARC 10

the time for both algorithms is spent in the communication with QHULL, *since data-exchange in the current implementation is done via text-files.*

Example 6.5 (Second order system) *To be able to visualize the admissible initial set, we apply the algorithms to the sampled version of the second order system (sample-time, horizon and constraints same as in the previous example)*

$$G(s) = \frac{1}{s^2 + 1} \tag{6.17}$$

In this example, the two algorithms performed similarly. The admissible initial set $\mathcal{I}$ together with its ellipsoidal approximation $\hat{\mathcal{I}}$ is given in the figure below. The terminal state domain is the unit circle.

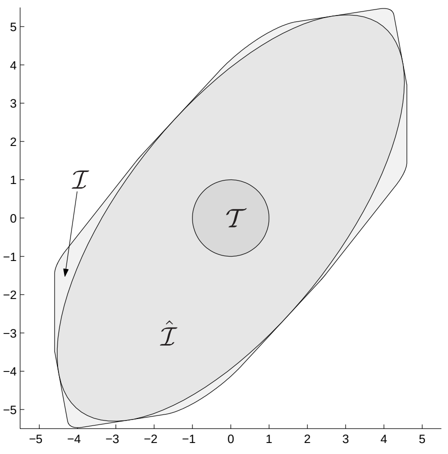

Figure 6.6: Admissible initial set $\mathcal{I}$ and its ellipsoidal approximation $\hat{\mathcal{I}}$.

For this system, we see that the ellipsoidal approximation of the admissible initial set is quite close to the exact set.

6.7 Conclusions

We have looked at the problem of finding the set of initial states for which it is possible to reach an ellipsoid centered around the origin, with a constrained control, in a specified number of steps. It was shown that the set can be described as the convex hull of a finite number of ellipsoids. Two algorithms were developed to calculate polytopic and ellipsoidal approximations of the set.

The algorithms were applied to two test examples. Although Algorithm 6.4 performed better than Algorithm 6.2 for these examples, practice has shown that they typically perform equivalently.

As interesting extension that needs further research is whether it is possible to transfer some of the results to uncertain and disturbed systems.

7

Robust MPC with Disturbances and State Estimation

So far, we have only dealt with MPC for systems without disturbances and the state exactly known. In this chapter, we propose an approach to design MPC controllers in the case of estimated states and/or unknown but bounded disturbances acting on the system and the output measurements. It will be shown that the joint uncertainty caused by the estimation error and the disturbances can be dealt with (conservatively) in a quite general LMI framework.

This chapter is organized as follows. In Section 7.1 we define the admissible systems, disturbances and the optimization formulation we want to use in MPC. This is followed by a description of a state estimation procedure in Section 7.2. The state estimate is used in an MPC algorithm developed in Section 7.3. This MPC algorithm is extended and generalized in various ways, and as an intermediate result, an approach to synthesize linear feedback controllers for disturbed systems is obtained. Finally, an example with a couple of experiments is presented.

7.1 Problem Formulation

The system we analyze is an extension of the nominal system in Section 3.2. To begin with, we assume that there are disturbances acting on the system dynamics. Furthermore, the state is not available. Instead, we have a disturbed output measurement

$$x(k+1) = Ax(k) + Bu(k) + B_\xi \xi(k) \tag{7.1a}$$
$$y(k) = Cx(k) + D_\eta \eta(k) + D_\zeta \zeta(k) \tag{7.1b}$$

As in the previous chapters, we also assume that the system is constrained. For simplicity, we will initially only consider the unit control constraint $|u(k)| \leq 1$.

The disturbances are unknown but bounded by ellipsoids (the class of disturbances will be extended later)

$$\eta(k) \in \{\eta : \eta^T W_\eta \eta \leq 1\} \tag{7.2a}$$
$$\xi(k) \in \{\xi : \xi^T W_\xi \xi \leq 1\} \tag{7.2b}$$
$$\zeta(k) \in \{\zeta, x : \zeta^T W_\zeta \zeta \leq x^T W_x x\} \tag{7.2c}$$

From the description above, we see that we only model an unstructured disturbance on the system dynamics, i.e., we do not know how $x(k)$ influences $\xi(k)$ (our MPC algorithm would not be convex otherwise). On the measurement however, we are able to incorporate a disturbance $\zeta(k)$ whose size is dependent on the system state.

The problem we want to solve is a minimax problem over the admissible future disturbances ξ, and possible state estimation error $x(k) - \hat{x}(k)$.

$$\boxed{\min_u \max_{x(k),\xi} \quad \sum_{j=0}^{N-1} x^T(k+j+1)Qx(k+j+1) + u^T(k+j)Ru(k+j)} \tag{7.3}$$

Notice that we thus explicitely want to take the possible estimation error into account. A standard solution is otherwise to neglect this mismatch and instead use $\hat{x}(k+j+1|k)$ in the objective [38, 41, 56].

Previous work

The principle problem above, or special cases thereof, has been studied before in the MPC framework. The case with full state information is dealt with in, e.g., [2] and [62]. Although computationally efficient, the approach in [2] suffers from a lack of a general performance measure. The work in [62] on the other hand allows (almost) arbitrary performance measures. This is at the expense of a simple disturbance model and exponential complexity in the computations (the disturbance model is a polytope and all vertices of the future possible disturbance polytopes are enumerated in the optimization).

The case with both estimation error and disturbances is studied in [3, 4]. The main contribution of our work, compared to the approach in [3, 4] is two-fold. To begin with, the estimation and control is done in a somewhat unified framework. Another, perhaps more important difference is that our approach allows a more general disturbance description and optimization objective. In [3, 4], the disturbance model is limited to element-wise bounds on the disturbances, and the performance objective is basically the deviation of the control input from the steady state input.

7.2 Deterministic State Estimation

In order to solve the minimax optimization problem, we must have an estimate $\hat{x}(k)$. Furthermore, since the optimization is performed over all admissible estimation errors, we must have a bound on the estimation error. Finding such an estimate is a classical problem, see, e.g, [39, 61] for approaches based on ellipsoidal calculus. A method to solve the similar one-step-ahead prediction problem using SDP was proposed in [25]. Adapting the basic ideas of those results to our disturbance model, state estimation and a slightly different optimality condition (determinant maximization instead of trace minimization in order to be consistent with the rest of the thesis), yields the following state estimation procedure.

The idea is to work with an ellipsoidal set in which the true state is guaranteed to lie. Given an old estimate $x(k-1) \in \mathcal{E}\left(\hat{x}(k-1), P(k-1)\right)$, we want to find a new estimate, given a measurement $y(k)$ and the disturbance model, such that $x(k) \in \mathcal{E}\left(\hat{x}(k), P(k)\right)$. More formally, the following implication should hold:

$$\begin{aligned} (x(k)-\hat{x}(k))^T P(k)(x(k)-\hat{x}(k)) &\leq 1 \\ \text{when} & \\ (x(k-1)-\hat{x}(k-1))^T P(k-1)(x(k-1)-\hat{x}(k-1)) &\leq 1 \end{aligned} \tag{7.4}$$

For each new output measurement, we want to find a new estimate $\hat{x}(k)$ and the corresponding matrix $P(k)$. Since $P(k)$ describes the confidence in the estimate, our goal is to minimize the size of the ellipsoid $\mathcal{E}\left(\hat{x}(k), P(k)\right)$. We will now show how this can be solved with SDP.

To begin with, we introduce a stacked vector defining a basis for all variables

$$z = \begin{bmatrix} x(k-1) \\ \xi(k-1) \\ \eta(k) \\ \zeta(k) \\ 1 \end{bmatrix} \tag{7.5}$$

and transformation matrices from z to the involved variables

$$T_{x(k)} = \begin{bmatrix} A & B_\xi & 0 & 0 & Bu(k-1) \end{bmatrix} \tag{7.6a}$$

$$T_{e(k)} = \begin{bmatrix} A & B_\xi & 0 & 0 & Bu(k-1)-\hat{x}(k) \end{bmatrix} \tag{7.6b}$$

$$T_{e(k-1)} = \begin{bmatrix} I & 0 & 0 & 0 & -\hat{x}(k-1) \end{bmatrix} \tag{7.6c}$$

$$T_y = \begin{bmatrix} -CA & -CB_\xi & -D_\eta & -D_\zeta & y(k)-CBu(k-1) \end{bmatrix} \tag{7.6d}$$

$$T_\xi = \begin{bmatrix} 0 & I & 0 & 0 & 0 \end{bmatrix} \tag{7.6e}$$

$$T_\eta = \begin{bmatrix} 0 & 0 & I & 0 & 0 \end{bmatrix} \tag{7.6f}$$

$$T_\zeta = \begin{bmatrix} 0 & 0 & 0 & I & 0 \end{bmatrix} \tag{7.6g}$$

$$T_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.6h}$$

With these matrices, the original variables are reconstructed as

$$\begin{aligned}
x(k) &= T_{x(k)}z && (7.7a)\\
x(k)-\hat{x}(k) &= T_{e(k)}z && (7.7b)\\
x(k-1)-\hat{x}(k-1) &= T_{e(k-1)}z && (7.7c)\\
y(k)-(Cx(k)+D_\eta\eta(k)+D_\zeta\zeta(k)) &= T_y z && (7.7d)\\
\xi(k-1) &= T_\xi z && (7.7e)\\
\eta(k) &= T_\eta z && (7.7f)\\
\zeta(k) &= T_\zeta z && (7.7g)\\
1 &= T_1 z && (7.7h)
\end{aligned}$$

We insert the expressions above into the condition (7.4), the output equation in (7.1) and the disturbance model (7.2). This gives us

$$\begin{aligned}
z^T T_{e(k)}^T P(k) T_{e(k)} z &\leq z^T T_1^T T_1 z\\
&\text{when}\\
z^T T_{e(k-1)}^T P(k-1) T_{e(k-1)} z &\leq z^T T_1^T T_1 z\\
z^T T_y^T T_y z &= 0 \qquad (7.8)\\
z^T T_\xi^T W_\xi T_\xi z &\leq z^T T_1^T T_1 z\\
z^T T_\eta^T W_\eta T_\eta z &\leq z^T T_1^T T_1 z\\
z^T T_\zeta^T W_\zeta T_\zeta z &\leq z^T T_{x(k)}^T W_x T_{x(k)} z
\end{aligned}$$

For easy notation, we introduce the matrices

$$\begin{aligned}
\Lambda &= T_1^T T_1 && (7.9a)\\
S_e &= \Lambda - T_{e(k-1)}^T P(k-1) T_{e(k-1)} && (7.9b)\\
S_y &= T_y^T T_y && (7.9c)\\
S_\xi &= \Lambda - T_\xi^T W_\xi T_\xi && (7.9d)\\
S_\eta &= \Lambda - T_\eta^T W_\eta T_\eta && (7.9e)\\
S_\zeta &= T_{x(k)}^T W_x T_{x(k)} - T_\zeta^T W_\zeta T_\zeta && (7.9f)
\end{aligned}$$

The idea is now to apply the S-procedure to handle the implication, see Theorem 2.1. To do this, we define non-negative scalars τ_ξ, τ_η and τ_ζ. We also introduce the indefinite scalar τ_y (relaxation of the equality constraint in (7.8)). Straightforward application of the S-procedure yields the sufficient condition

$$\Lambda - T_{e(k)}^T P(k) T_{e(k)} \succeq \tau_e S_e + \tau_y S_y + \tau_\xi S_\xi + \tau_\eta S_\eta + \tau_\zeta S_\zeta \qquad (7.10)$$

The unknown variables on the left-hand side are $\hat{x}(k)$ (in $T_{e(k)}$) and $P(k)$. By introducing $Y = P^{1/2}(k)$ and $b = -P^{-1/2}(k)\hat{x}(k)$ we see that

$$P^{1/2}(k)T_{e(k)} = \begin{bmatrix} YA & YB_\xi & 0 & 0 & YBu(k-1)+b \end{bmatrix} = \vartheta(Y,b) \qquad (7.11)$$

With this structure, we can apply a Schur complement to obtain

$$\begin{bmatrix} \Lambda - \tau_e S_e - \tau_y S_y - \tau_\xi S_\xi - \tau_\eta S_\eta - \tau_\zeta S_\zeta & \vartheta^T(Y,b) \\ \vartheta(Y,b) & I \end{bmatrix} \succeq 0 \tag{7.12}$$

Since the volume of the ellipsoid is proportional to $\det(P^{-1/2})$, a suitable way to find $P(k)$ is to minimize $\det(P^{-1/2})$, or equivalently, maximize $\det(P^{1/2}) = \det(Y)$. The optimization problem to find the update of the state estimate can therefore be written as a MAXDET problem

$$\boxed{\begin{array}{rl} \max\limits_{Y,b,\tau} & \det(Y) \\ \text{subject to} & (7.12) \\ & \tau_\xi \geq 0 \\ & \tau_\eta \geq 0 \\ & \tau_\zeta \geq 0 \end{array}} \tag{7.13}$$

As is mentioned in [25], the SDP above can be simplified. To begin with, the indefinite variable τ_y can be eliminated since it is multiplied with a definite matrix (τ_y will be $-\infty$ at the optimum [11]). Furthermore, the optimization of $\hat{x}(k)$ and $P(k)$ can be done independently. The interested reader is referred to [25] for details.

7.3 Minimax MPC Design

It turns out that the derivation of our MPC algorithm is most easily done if we formulate the problem as in Section 3.3.1. To do this, we introduce stacked vectors with future states, control inputs and disturbances (beware the slight difference in the definition of X compared to Equation (3.7))

$$X = \begin{bmatrix} x(k+1) \\ x(k+2) \\ \vdots \\ x(k+N) \end{bmatrix}, \quad U = \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+N-1) \end{bmatrix}, \quad \Xi = \begin{bmatrix} \xi(k) \\ \xi(k+1) \\ \vdots \\ \xi(k+N-1) \end{bmatrix} \tag{7.14}$$

Standard use of the system model (7.1) allows us to write

$$X = Hx(k) + SU + G\Xi \tag{7.15}$$

The minimax problem (7.3) can be written in these variables as

$$\min_U \max_{x(k),\Xi} \quad (Hx(k) + SU + G\Xi)^T \bar{Q}(Hx(k) + SU + G\Xi) + U^T \bar{R} U$$

We reformulate this using epigraph rewriting (recall Section 4.6.1)

$$\begin{array}{rl} \min\limits_{U,t} & t \\ \text{subject to} & \max\limits_{x(k),\Xi} (Hx(k) + SU + G\Xi)^T \bar{Q}(Hx(k) + SU + G\Xi) + U^T \bar{R} U \leq t \end{array}$$

We will now show that this problem can be (conservatively) rewritten and solved, using the same principles as in the state estimation procedure. As a first step, we introduce a basis

$$z = \begin{bmatrix} x(k) \\ \Xi \\ 1 \end{bmatrix} \tag{7.16}$$

and some transformation matrices

$$\begin{aligned} T_x &= \begin{bmatrix} H & G & SU \end{bmatrix} && (7.17\text{a}) \\ T_e &= \begin{bmatrix} I & 0 & -\hat{x}(k) \end{bmatrix} && (7.17\text{b}) \\ T_u &= \begin{bmatrix} 0 & 0 & U \end{bmatrix} && (7.17\text{c}) \\ T_1 &= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} && (7.17\text{d}) \end{aligned}$$

We also introduce the transformation $T_{\xi(j)}$, i.e., $\xi(k+j) = T_{\xi(j)}z$. The constraint in the rewritten minimax formulation, using the bounds on disturbances and estimation error, can now be written as

$$\begin{aligned} z^T T_x^T \bar{Q} T_x z + z^T T_u^T \bar{R} T_u^T z &\leq t z^T T_1^T T_1 z \\ \text{when}& \\ z^T T_e^T P(k) T_e z &\leq z^T T_1^T T_1 z \\ z^T T_{\xi(j)}^T W_\xi T_{\xi(j)} z &\leq z^T T_1^T T_1 z \end{aligned} \tag{7.18}$$

Again, we introduce $\Lambda = T_1^T T_1$ and approximate with an S-procedure

$$t\Lambda - T_x^T \bar{Q} T_x - T_u^T \bar{R} T_u^T \succeq \tau_e(\Lambda - T_e^T P(k) T_e) + \sum_{j=0}^{N-1} \tau_{\xi,j}(\Lambda - T_{\xi(j)}^T W_\xi T_{\xi(j)})$$

To make this constraint linear in U (T_x and T_u), we apply a Schur complement

$$\begin{bmatrix} t\Lambda - \tau_e(\Lambda - T_e^T P(k) T_e) - \sum_{j=0}^{N-1} \tau_{\xi,j}(\Lambda - T_{\xi(j)}^T W_\xi T_{\xi(j)}) & T_x^T & T_u^T \\ T_x & \bar{Q}^{-1} & 0 \\ T_u & 0 & \bar{R}^{-1} \end{bmatrix} \succeq 0 \tag{7.19}$$

Our approximation of the minimax problem (7.3) boils down to the SDP

$$\boxed{\begin{array}{rl} \min\limits_{U,t,\tau} & t \\ \text{subject to} & (7.19) \\ & |U| \leq 1 \\ & \tau_e \geq 0 \\ & \tau_{\xi,j} \geq 0 \end{array}} \tag{7.20}$$

The special case with no disturbances is recovered by removing the corresponding columns in the transformation matrices (7.17), and the matrices in (7.19) related to the disturbances. The case with all states exactly known is obtained equivalently except that SU is changed to $SU + Hx(k)$ in (7.17).

7.3.1 Extensions of the MPC Algorithm

The MPC controller defined by the SDP (7.20) can be extended in various ways.

Robust satisfaction of state constraints

It is easy to extend the MPC controller in order to cope with robust satisfaction of linear constraints. Assume we have an output

$$z(k) = Mx(k) + Du(k) \tag{7.21}$$

for which we have a constraint

$$z(k) \leq 1 \tag{7.22}$$

We introduce a stacked vector with all constrained outputs

$$Z = \begin{bmatrix} z(k) \\ z(k+1) \\ \vdots \\ z(k+N) \end{bmatrix} \tag{7.23}$$

Standard definition of H_z, S_z and G_z using the system model and the definition of $z(k+j)$ gives us the stacked prediction

$$Z = H_z x(k) + S_z U + G_z \Xi \tag{7.24}$$

If we let z_j, h_j, s_j and g_j denote the rows of Z, H_z, S_z and G_z respectively, the jth element of Z can be written as

$$z_j = h_j x(k) + s_j U + \sum_{i=0}^{N-1} g_{ji} \xi(k+i) \tag{7.25}$$

where g_{ji} denotes a partitioning of g_j compatible with the dimension of the disturbance $\xi(k+i)$. With $x(k) \in \mathcal{E}(\hat{x}(k), P(k))$, Lemma 4.1 gives us

$$\max_{x(k),\Xi} z_j = h_j \hat{x}(k) + \sqrt{h_j P^{-1}(k) h_j^T} + s_j U + \sum_{i=0}^{N-1} \sqrt{g_{ji} W_\xi^{-1} g_{ji}^T} \tag{7.26}$$

and the robustified state constraint can be written as

$$h_j \hat{x}(k) + \sqrt{h_j P^{-1}(k) h_j^T} + s_j U + \sum_{i=0}^{N-1} \sqrt{g_{ji} W_\xi^{-1} g_{ji}^T} \leq 1 \tag{7.27}$$

Clearly, this is a linear constraint in the unknown variable U.

Remark 7.1
In the robustified state constraint, the uncertainty in the state estimate reveals itself in the term $\sqrt{h_j P^{-1}(k) h_j^T}$. *This indicates that a perhaps more suitable performance measure in the optimization of* $P(k)$ *is some function* $\phi(h_j P^{-1}(k) h_j^T)$.

Feedback predictions: full state information

In the case of full state information, we can easily adopt the idea of feedback predictions [2, 42, 60, 62]. The idea in feedback predictions (sometimes called closed loop predictions) is to parameterize the control sequence using a fixed feedback matrix L

$$u(k+j) = -Lx(k+j) + v(k+j) \tag{7.28}$$

When there are no uncertainties or disturbances, this parameterization will not influence the solution. With disturbances however, feedback predictions is a simple way to improve performance. If we insert the parameterization into the system model, we obtain

$$x(k+1) = (A - BL)\, x(k) + Bv(k) + B_\xi \xi(k) \tag{7.29}$$

Straightforward use of (7.29) gives us matrices H, S and G to define the stacked predictions

$$X = Hx(k) + SV + G\Xi \tag{7.30}$$

The parameterization (7.28) also allows us to define U

$$U = H_u x(k) + S_u V + G_u \Xi \tag{7.31}$$

The basis z, used in the derivation of (7.20), no longer contains $x(k)$

$$z = \begin{bmatrix} \Xi \\ 1 \end{bmatrix} \tag{7.32}$$

which forces us to define new transformation matrices (T_e is no longer needed)

$$\begin{aligned} T_x &= \begin{bmatrix} G & Hx(k) + SV \end{bmatrix} && (7.33a) \\ T_u &= \begin{bmatrix} G_u & H_u x(k) + S_u V \end{bmatrix} && (7.33b) \\ T_1 &= \begin{bmatrix} 0 & 1 \end{bmatrix} && (7.33c) \end{aligned}$$

From this point, the same derivation as for (7.20) can be used. Notice that the constraints $|u(k+j)| \leq 1$ are mapped into state constraints which are handled using the results in the previous section.

Feedback predictions: measured output

In the case of estimated states, we can parameterize the input as

$$u(k+j) = -L\hat{x}(k+j|k+j) + v(k+j) \tag{7.34}$$

The difficulty with this extension is that we have to have an expression of what our estimate will look like in the future, i.e. $\hat{x}(k+j|k+j)$. We denote these the predicted estimates. In this section, we propose an approach to this in the case when $W_x = W_\zeta = 0$ (the problem is non-convex otherwise). The idea is to assume that a linear estimator will be used in the future

$$\begin{aligned}\hat{x}(k+j|k+j) &= A\hat{x}(k+j-1|k+j-1) + Bu(k+j-1)\\ &\quad + K(y(k+j) - C(A\hat{x}(k+j-1|k+j-1) + Bu(k+j-1)))\end{aligned} \tag{7.35}$$

If we insert the parameterized feedback, $u(k+j) = -L\hat{x}(k+j|k+j) + v(k+j)$, straightforward calculations yields the following linear system for the estimate and true state

$$\begin{aligned}\begin{bmatrix}\hat{x}(k+j|k+j)\\ x(k+j)\end{bmatrix} &= \begin{bmatrix}A-BL-KCA & KCA\\ -BL & A\end{bmatrix}\begin{bmatrix}\hat{x}(k+j-1|k+j-1)\\ x(k+j-1)\end{bmatrix}\\ &\quad + \begin{bmatrix}B\\ B\end{bmatrix}v(k+j-1) + \begin{bmatrix}KCB_\xi\\ B_\xi\end{bmatrix}\xi(k+j-1) + \begin{bmatrix}KD_\eta\\ 0\end{bmatrix}\eta(k+j)\end{aligned}$$

We define the extended state vector

$$\tilde{x}(k+j) = \begin{bmatrix}\hat{x}(k+j|k+j)\\ x(k+j)\end{bmatrix} \tag{7.36}$$

together with matrices $\tilde{A}$, $\tilde{B}$, $\tilde{B}_\xi$ and $\tilde{D}_\eta$ to be able to write

$$\tilde{x}(k+j) = \tilde{A}\tilde{x}(k+j-1) + \tilde{B}v(k+j-1) + \tilde{B}_\xi\xi(k+j-1) + \tilde{D}_\eta\eta(k+j) \tag{7.37}$$

We introduce stacked versions of the augmented state and future measurement disturbances

$$\tilde{X} = \begin{bmatrix}\tilde{x}(k+1)\\ \tilde{x}(k+2)\\ \vdots\\ \tilde{x}(k+N)\end{bmatrix}, \Theta = \begin{bmatrix}\eta(k+1)\\ \eta(k+2)\\ \vdots\\ \eta(k+N)\end{bmatrix} \tag{7.38}$$

Standard definition of the matrices $\tilde{H}$, $\tilde{S}$ and $\tilde{G}$, $\tilde{M}$, V and Ξ yields

$$\tilde{X} = \tilde{H}\tilde{x}(k) + \tilde{S}V + \tilde{G}\Xi + \tilde{M}\Theta \tag{7.39}$$

We now partition the matrix $\tilde{H}$ as $\tilde{H} = \begin{bmatrix}\tilde{H}_x & \tilde{H}_{\hat{x}}\end{bmatrix}$. This gives

$$\tilde{X} = \tilde{H}_x x(k) + \tilde{H}_{\hat{x}}\hat{x}(k|k) + \tilde{S}V + \tilde{G}\Xi + \tilde{M}\Theta \tag{7.40}$$

Introduce transformation matrices that pulls out stacked vectors containing the true states and the predicted estimates

$$\hat{X} = T_{\hat{X}}\tilde{X}, \quad X = T_X\tilde{X} \tag{7.41}$$

At this point, we are almost ready to return to the underlying MPC problem, but first we have to express U in terms of $\hat{x}(k|k)$, V and $\hat{X}$

$$\begin{aligned} U &= \begin{bmatrix} -L\hat{x}(k|k) + v(k) \\ \vdots \\ -L\hat{x}(k+N-1|k+N-1) + v(k+N-1) \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & \dots & 0 \\ -L & 0 & \dots & 0 \\ 0 & -L & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & -L \end{bmatrix} \hat{X} + \begin{bmatrix} -L\hat{x}(k|k) \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + V \end{aligned} \tag{7.42}$$

By inserting (7.41) and (7.39) into (7.42), we obtain H_u, F_u, S_u, G_u and M_u

$$U = H_u x(k) + F_u\hat{x}(k|k) + S_u V + G_u\Xi + M_u\Theta \tag{7.43}$$

In the same way, we use (7.41) and (7.39) to obtain H, F, S, G and M

$$X = Hx(k) + F\hat{x}(k|k) + SV + G\Xi + M\Theta \tag{7.44}$$

The only conceptual difference in the expressions above, compared to the result with full state information, is the explicit influence of the current state estimate $\hat{x}(k|k)$ and future measurement errors Θ.

In order to develop the SDP (7.19) for the MPC algorithm, we introduce a basis

$$z = \begin{bmatrix} x(k) \\ \Xi \\ \Theta \\ 1 \end{bmatrix} \tag{7.45}$$

and the needed transformation matrices

$$T_x = \begin{bmatrix} H & G & M & SV + F\hat{x}(k|k) \end{bmatrix} \tag{7.46a}$$
$$T_e = \begin{bmatrix} I & 0 & 0 & -\hat{x}(k) \end{bmatrix} \tag{7.46b}$$
$$T_u = \begin{bmatrix} H_u & G_u & M_u & S_u V + F_u\hat{x}(k|k) \end{bmatrix} \tag{7.46c}$$
$$T_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.46d}$$

With these expressions, exactly the same derivation with an S-procedure as in the derivation of the LMI (7.19) can be used to formulate a convex optimization problem. The only differences is that additional variables $\tau_\eta \geq 0$ have to be introduced to take care of the uncertain variable Θ, and the control constraints are mapped into state constraints, just as in the algorithm with feedback predictions for full state information. The details are omitted for brevity.

Remark 7.2 (Connections to dynamic programming)
The use of feedback predictions and the introduced concept predicted estimates can be viewed from a dynamic programming [6] point of view. The central idea in dynamic programming is to exploit the fact that we will have a new measurement at $k+1$, and with this new measurement we can calculate the optimal input sequence over the reduced $N-1$ horizon. In the linear Gaussian case, this gives the LQG controller. Feedback predictions can be seen as a cheap trick to mimic the idea that the optimal input will be applied in the future. Instead of optimal, we assume that there will be some feedback at least. The optimal input uses the optimal state estimate, calculated using all the measurements obtained in the future. In our case, we replace this optimal estimate with any estimate.

Let us perform some explicit calculations on feedback predictions

Example 7.1 (Feedback predictions and predicted estimates) *Consider a scalar disturbed unstable system*

$$x(k+1) = 2x(k) + u(k) + \xi(k),\ \ y(k) = x(k) + \eta(k) \tag{7.47}$$

We select $L = 1.5$ and $K = 1.5$ which gives us

$$\tilde{A} = \begin{bmatrix} -2.5 & 3 \\ -1.5 & 2 \end{bmatrix},\ \tilde{B} = \begin{bmatrix} 1 \\ 1 \end{bmatrix},\ \tilde{B}_\xi = \begin{bmatrix} 1.5 \\ 1 \end{bmatrix},\ \tilde{D}_\eta = \begin{bmatrix} 1.5 \\ 0 \end{bmatrix} \tag{7.48}$$

By using the definition of $\tilde{x}(k)$ (7.39) and the system matrices above, we can calculate the state and estimated state two steps ahead

$$\begin{bmatrix} \hat{x}(k+2) \\ x(k+2) \end{bmatrix} = \begin{bmatrix} 1.75 & -1.5 \\ 0.75 & -0.5 \end{bmatrix} \begin{bmatrix} \hat{x}(k) \\ x(k) \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} v(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} v(k+1) + \\ \begin{bmatrix} -0.75 \\ -0.25 \end{bmatrix} \xi(k) + \begin{bmatrix} 1.5 \\ 1 \end{bmatrix} \xi(k+1) + \begin{bmatrix} -3.75 \\ -2.25 \end{bmatrix} \eta(k) + \begin{bmatrix} 1.5 \\ 0 \end{bmatrix} \eta(k+1) \tag{7.49}$$

An alternative is to predict $x(k+2)$ without any feedback predictions.

$$x(k+2) = 4x(k) + 2u(k) + u(k+1) + 2\xi(k) + \xi(k+1) \tag{7.50}$$

If we now insert our current state estimate, $x(k) = \hat{x}(k) + e(k)$, we can gather all the uncertainties in the predictions of $x(k+2)$. With feedback predictions, we have the collected uncertainty

$$-0.5e(k) - 0.25\xi(k) + \xi(k+1) - 2.25\eta(k+1) \tag{7.51}$$

and without

$$4e(k) + 2\xi(k) + \xi(k+1) \tag{7.52}$$

The impact of the current estimation error is reduced, and so is the impact of future disturbances acting on the system. This is at the cost of an uncertainty due to future measurement errors. Notice that the improved certainty of $x(k+2)$ also is at the expense of uncertainty in $u(k+2) = -L\hat{x}(k+2) + v(k+2)$. Clearly, the choice of L and K is intricate and general conclusions can not be drawn, not even for this simple example.

Terminal state constraints

Much work on MPC has been devoted to the development of suitable terminal state constraints and terminal state weights. In the nominal, undisturbed case, the terminal state weight basically accounts for guaranteed asymptotic stability, whereas the terminal state constraints on its own is enough for stability.

Since we have persistent time-varying disturbances in this chapter, asymptotic stability cannot be achieved. However, guaranteed stability would of course be desirable feature. This leads us to the problem of finding suitable terminal state constraints.

The underlying idea in the design of terminal state constraints for nominal systems is to find a domain were a standard linear feedback can be applied, with the restriction that the domain should be positively invariant and all constraints satisfied therein. If we extend these ideas to the disturbed case, the only difference is that the terminal state domain should be robustly positively invariant.

Finding (robustly) invariant sets is a well studied research area, see [9] for a recent survey. Whereas the problem of finding robustly invariant ellipsoidal sets for continuous-time systems with our disturbance model is a classical problem with a known solution, see, e.g., [13] or the aforementioned survey, the same does not seem to hold for discrete time systems. At least, we have not managed to find the result we are about to present in the literature.

Recall the disturbed system (7.1), now assumed to be controlled with the nominal controller $u(k) = -Lx(k)$ (hence, we only consider the case with all states available)

$$x(k+1) = (A - BL)x(k) + B_\xi \xi(k) \tag{7.53}$$

An ellipsoid $\mathcal{E}_W$ is said to be robustly positively invariant for the system if

$$\begin{aligned} x^T(k+1)Wx(k+1) &\leq 1 \\ &\text{when} \\ x^T(k)Wx(k) &\leq 1 \\ \xi^T(k)W_\xi\xi(k) &\leq 1 \end{aligned} \tag{7.54}$$

By inserting the model of $x(k+1)$, we can write this as

$$\begin{bmatrix} x(k) \\ \xi(k) \\ 1 \end{bmatrix}^T \begin{bmatrix} (A-BL)^TW(A-BL) & (A-BL)^TWB_\xi & 0 \\ B_\xi^TW(A-BL) & B_\xi^TWB_\xi & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x(k) \\ \xi(k) \\ 1 \end{bmatrix} \leq 0$$

$$\text{when} \quad \begin{bmatrix} x(k) \\ \xi(k) \\ 1 \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & W_\xi & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x(k) \\ \xi(k) \\ 1 \end{bmatrix} \leq 0$$

$$\begin{bmatrix} x(k) \\ \xi(k) \\ 1 \end{bmatrix}^T \begin{bmatrix} W & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x(k) \\ \xi(k) \\ 1 \end{bmatrix} \leq 0$$

Straightforward application of the S-procedure yields the sufficient condition

$$\begin{bmatrix} (A-BL)^TW(A-BL) & (A-BL)^TWB_\xi & 0 \\ B_\xi^TW(A-BL) & B_\xi^TWB_\xi & 0 \\ 0 & 0 & -1 \end{bmatrix} \preceq \tau_1 \begin{bmatrix} 0 & 0 & 0 \\ 0 & W_\xi & 0 \\ 0 & 0 & -1 \end{bmatrix} + \tau_2 \begin{bmatrix} W & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{7.55}$$

This is a BMI due to the multiplication of τ_2 and W, but by fixating τ_2, an LMI is obtained. Optimization of W, or more precisely some measure of W related to the size of the invariant ellipsoid $\mathcal{E}_W$, can therefore easily be done by a line-search in τ_2. However, the BMI above can be written in a better format. We first notice that we can split the BMI into two constraints

$$\begin{aligned} \tau_1 + \tau_2 &\leq 1 && (7.56) \\ \begin{bmatrix} \tau_2 W & 0 \\ 0 & \tau_1 W_\xi \end{bmatrix} - \begin{bmatrix} (A-BL)^T \\ B_\xi^T \end{bmatrix} W \begin{bmatrix} (A-BL) & B_\xi \end{bmatrix} &\succeq 0 && (7.57) \end{aligned}$$

If we multiply the second constraint from left and right with the positive definite matrix

$$\begin{bmatrix} W^{-1} & 0 \\ 0 & I \end{bmatrix} \tag{7.58}$$

we get the matrix inequality

$$\begin{bmatrix} \tau_2 W^{-1} & 0 \\ 0 & \tau_1 W_\xi \end{bmatrix} - \begin{bmatrix} W^{-1}(A-BL)^T \\ B_\xi^T \end{bmatrix} W \begin{bmatrix} (A-BL)W^{-1} & B_\xi \end{bmatrix} \succeq 0 \tag{7.59}$$

Define $Y = W^{-1}$ and a Schur complement gives us

$$\begin{bmatrix} \tau_2 Y & 0 & YA^T - YL^TB^T \\ 0 & \tau_1 W_\xi & B_\xi^T \\ AY - BLY & B_\xi & Y \end{bmatrix} \succeq 0 \tag{7.60}$$

Obviously, we still have a BMI due to the product $\tau_2 Y$, but in the Y coordinate, two things are gained. To begin with, we can solve a MAXDET problem (for fixed τ_2) to maximize the volume of the robustly invariant ellipsoid. Control and state constraints are easily incorporated using Lemma (4.1), but for brevity we assume only a standard control constraint, $|-Lx(k)| \le 1 \; \forall x(k) \in \mathcal{E}_W$.

$$\boxed{\begin{array}{ll} \max\limits_{Y,\tau_1,\tau_2} & \det(Y) \\ \text{subject to} & \begin{bmatrix} \tau_2 Y & 0 & YA^T - YL^TB^T \\ 0 & \tau_1 W_\xi & B_\xi^T \\ AY - BLY & B_\xi & Y \end{bmatrix} \succeq 0 \\ & \tau_1 + \tau_2 \le 1 \\ & \tau_1 \ge 0 \\ & \tau_2 \ge 0 \\ & L_i Y L_i^T \le 1 \end{array}} \tag{7.61}$$

The second reason for working in the variable Y is that we actually have derived a synthesis method to find L. By defining $Z = LY$, which can be done since Y is invertible, we can find a feedback controller that stabilizes the disturbed system under control constraints as long as the initial condition is in $\mathcal{E}_W$

$$\boxed{\begin{array}{ll} \max\limits_{Y,Z,\tau_1,\tau_2} & \det(Y) \\ \text{subject to} & \begin{bmatrix} \tau_2 Y & 0 & YA^T - Z^TB^T \\ 0 & \tau_1 W_\xi & B_\xi^T \\ AY - BZ & B_\xi & Y \end{bmatrix} \succeq 0 \\ & \tau_1 + \tau_2 \le 1 \\ & \tau_1 \ge 0 \\ & \tau_2 \ge 0 \\ & \begin{bmatrix} 1 & e_i^T Z \\ Z^T e_i & Y \end{bmatrix} \succeq 0 \end{array}} \tag{7.62}$$

To obtain the last constraint above, we introduced the unit vector e_i and inserted the definition of L to obtain $L_iYL_i^T = e_i^T ZY^{-1}YY^{-1}Z^Te_i$. A Schur complement maps the original constraint into an LMI.

So, given a feedback matrix L, either chosen beforehand or optimized and defined as $L = ZY^{-1}$, and the corresponding matrix $W = Y^{-1}$, how are these incorporated into the various versions of the MPC algorithm developed earlier in this chapter? In all versions, a basis z has been defined, and with this basis, a transformation matrix T_x. Adding an ellipsoidal terminal state constraint is easily done within this framework. Recall that the vector X is defined as $T_x z$, hence the state $x(k+N)$ can be written as $\begin{bmatrix} 0 & \dots & 0 & I \end{bmatrix} T_x z$. For simple notation, we write this as $x(k+N) = T_{x(k+N)} z$. The terminal state constraint $x(k+N)Wx(k+N) \le 1$ can be written as

$$z^T T_{x(k+N)}^T W T_{x(k+N)} z \le z^T \Lambda z \tag{7.63}$$

If we use the same approach with an S-procedure in order to guarantee that this constraint holds for all possible disturbances, we obtain an additional constraint to be added to the MPC algorithms (remember that we assume the state to be available)

$$\begin{bmatrix} \Lambda - \sum_{j=0}^{N-1} \tau_{\xi,j}(\Lambda - T_{\xi(j)}^T W_\xi T_{\xi(j)}) & T_{x(k+N)}^T \\ T_{x(k+N)} & W^{-1} \end{bmatrix} \succeq 0 \tag{7.64}$$

Of course, the τ variables in the constraint above does not have to be the same as those used in the relaxation (7.19) of the performance measure.

Notice that the LMI above is linear in the matrix $Y = W^{-1}$. This means that selection of L and Y, defined by the LMI (7.62) with fix τ_2, actually can be incorporated into the MPC algorithm.

Extension of admissible disturbances

The disturbances we have used so far have been bounded by one single ellipsoid centered at the origin. This can easily be extended in order to describe disturbances bounded by several off-centered ellipsoids, hence allowing, e.g., half-spaces and polytopes to be described. It turns out that everything in this chapter can be extended to disturbance bounded by the intersection of a set of ellipsoids

$$\eta(k) \in \cap_{i=1\ldots n_\eta} \{\eta : \begin{bmatrix}\eta\\1\end{bmatrix}^T W_{\eta,i} \begin{bmatrix}\eta\\1\end{bmatrix} \leq 1\} \tag{7.65a}$$

$$\xi(k) \in \cap_{i=1\ldots n_\xi} \{\xi : \begin{bmatrix}\xi\\1\end{bmatrix}^T W_{\xi,i} \begin{bmatrix}\xi\\1\end{bmatrix} \leq 1\} \tag{7.65b}$$

$$\zeta(k) \in \cap_{i=1\ldots n_\zeta} \{\zeta, x : \begin{bmatrix}\zeta\\1\end{bmatrix}^T W_{\zeta,i} \begin{bmatrix}\zeta\\1\end{bmatrix}^T \leq \begin{bmatrix}x\\1\end{bmatrix}^T W_{x,i} \begin{bmatrix}x\\1\end{bmatrix}\} \tag{7.65c}$$

Basically, all that is needed to extend the various results in this chapter is to define more τ variables in order to perform a relaxation of the extra ellipsoidal constraints, using the S-procedure. A motivation for this, initially rather awkward-looking model, is that linear constraints now can be used. As an example , $c^T\xi(k) \leq 1$ can be described as

$$\begin{bmatrix}\xi(k)\\1\end{bmatrix}^T \begin{bmatrix}0 & \frac{1}{2}c\\ \frac{1}{2}c^T & 0\end{bmatrix} \begin{bmatrix}\xi(k)\\1\end{bmatrix} \leq 1 \tag{7.66}$$

7.4 Examples

We conclude this chapter with an example to illustrate some aspects of the proposed MPC controller.

Example 7.2 (Robust output constraint satisfaction) *This example is taken from [2, 3, 4]. It is a second order system with only one state measured.*

$$\begin{aligned} x(k+1) &= \begin{bmatrix} 1.6463 & -0.7866 \\ 1 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(k) + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \xi(k) \\ y(k) &= \begin{bmatrix} 0.1404 & 0 \end{bmatrix} x(k) + \eta(k) \\ z(k) &= \begin{bmatrix} -2.6238 & 2.9045 \end{bmatrix} x(k) \end{aligned}$$

The disturbances are bounded with

$$W_\xi = \begin{bmatrix} 5000 & 0 \\ 0 & 5000 \end{bmatrix}, W_\eta = 400 \tag{7.67}$$

and the initial state is only known to lie in the ellipsoid

$$\hat{x}(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, P(0) = \begin{bmatrix} 8 & 0 \\ 0 & 8 \end{bmatrix} \tag{7.68}$$

The numerical data above are (conservative) ellipsoidal approximations of the sets used in [4] ($||\xi(k)||_\infty \leq 0.01$, $||\eta(k)||_\infty \leq 0.05$ and $||x(0) - \hat{x}(0)||_\infty \leq 0.25$). Of course, we could have applied the extensions mentioned in Section 7.3.1 to obtain the same disturbance model.
The goal is to have the variable $y(k)$ track a constant reference $r(k) = 1$, so the performance weights are chosen as $Q = C^T C$ and $R = 0.01$. The tracking formulation requires a slight modification of the algorithm, see Remark 3.1 and Remark 4.2. The signal $z(k)$ is constrained and has to satisfy $-1 \leq z(k) \leq 3$, and the control input has to satisfy $|u(k)| \leq 2$.
The MPC algorithm is applied to the system from the initial state $x(0) = 0$ with a prediction horizon $N = 10$. Uniformly (over the ellipsoids) distributed disturbances were applied on both the system dynamics and the measurements.
There is a substantial non-minimum phase behavior to the output $z(k)$, so the state constraint turns out to be the limiting factor for performance.
We can see in Figure 7.1 that the state constraint is rather conservatively satisfied when we apply the MPC algorithm. The overall performance seem to be pretty much similar to that reported in [4], except a slightly improved overshoot.
By performing a number of different experiments, it turns out that the initial uncertainty in the state estimate is the most limiting factor (which perhaps not is so surprising since the disturbances are rather small). The approach with feedback predictions for measured outputs was tested, but gave no improvement.
As a second experiment, we assumed the initial state to be exactly known, and instead increased the size of the disturbances $W_\xi = \begin{bmatrix} 312 & 0 \\ 0 & 312 \end{bmatrix}$. Once again, this disturbance is chosen to allow comparison with the results in [4].

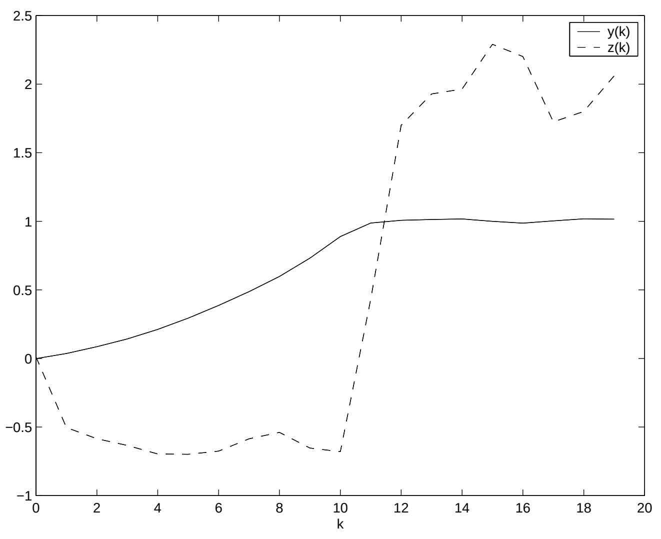

Figure 7.1: Resulting closed loop for first experiment. The state constraint is fulfilled and reasonably good tracking is achieved.

The closed loop behavior was almost the same as in the first experiment (same disturbance realizations). This should be compared to the results reported in [4] where this setup led to a significant increase of the rise-time.

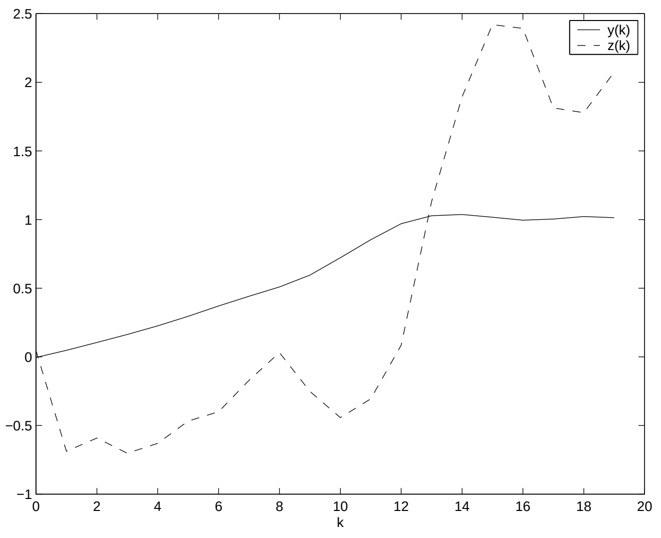

Figure 7.2: Experiment 2; initial state known, but larger disturbances. The performance is basically the same as for the first experiment.

As a final experiment, we study how feedback predictions can improve the approximation of the minimax problem. We assumed full state information, and solved the optimization problem (7.20) in the initial state, with and without feedback predictions. The optimal value of t, which serves as an upper bound of the original cost (7.3), was recorded. This was done for a number of disturbance models, parameterized as $W_\xi = \gamma I$. The feedback matrix L was chosen as an LQ controller calculated using the same performance weights as the MPC controller.

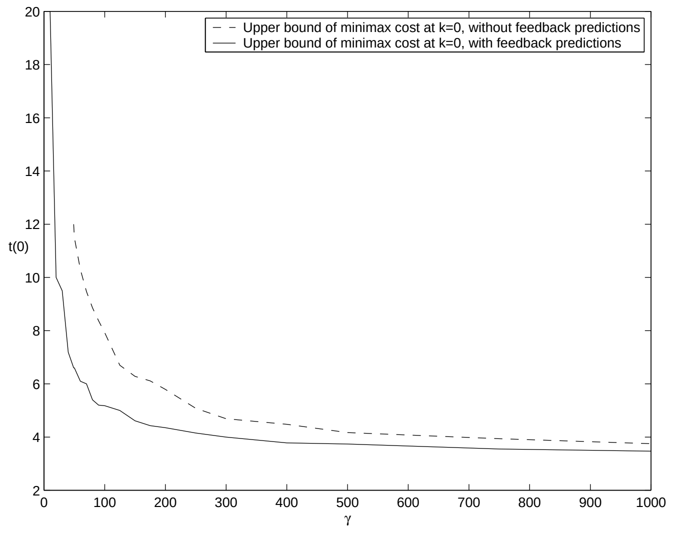

Figure 7.3: Optimal value of the upper bound t in the initial state, evaluated with different disturbance models $W_\xi = \gamma I$.

It was noticed that feasibility was lost at $\gamma = 49$ for the algorithm without feedback predictions, whereas the feasibility was lost at $\gamma = 9$ with feedback predictions. Of course, the choice of L is important and it is easy to select another L so that the upper bound either increases or decreases.
To see whether the upper bound at the initial state actually tells us anything about the actual performance, we select a disturbance model with $\gamma = 100$ and simulate the system with and without feedback predictions.
As we see in Figure 7.4, feedback predictions have indeed improved the performance. Actually, the closed loop behavior is almost unaffected compared to the first experiment we performed.

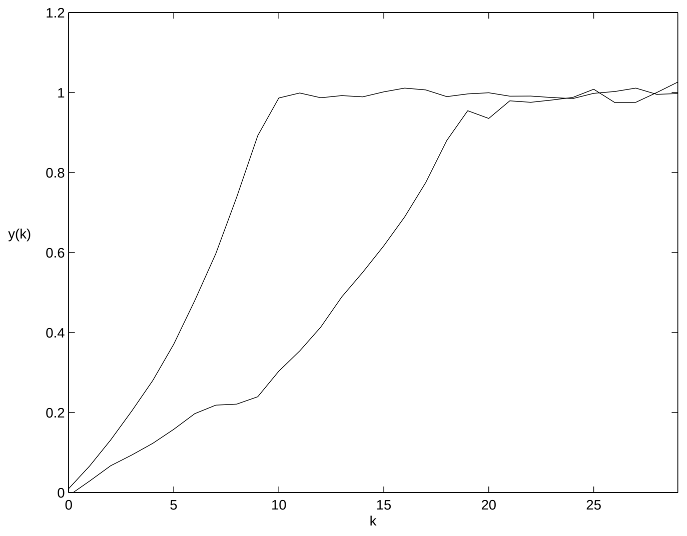

Figure 7.4: Application of feedback predictions. Closed loop performance is almost unaffected with feedback predictions despite the larger disturbances (fast response), while there is a substantial deterioration of the rise-time without feedback predictions (slow response).

7.5 Conclusions

By applying classical methods to perform state estimation for systems with bounded disturbances, and using a convex optimization framework, we managed to develop a minimax MPC algorithm which explicitly incorporates knowledge of possible estimation errors and disturbances, and tries to counteract these. The MPC algorithm turns out to be very flexible and could easily be extended in various ways.

It was recognized in the examples that the closed loop response easily became overly conservative. Although this is a natural behavior since a minimax problem is solved, the level of conservativeness can be a problem. Feedback predictions can solve this to some extent, so this should be used when applicable. Practice has shown that the choice of a suitable feedback gain for the feedback predictions is a complex problem. The choice of the feedback gain and the observer gain when both feedback predictions and predicted estimates are used is an even more complicated problem. Investigation of this, together with extensions of the concept feedback predictions, is currently being performed.

The main drawback of the approach is the computational complexity. Currently, we solve the SDPs with general purpose solvers. It would be interesting to investigate if it is possible to exploit any structure in the problem, in the same way as in, e.g., QP [68], robust QP [31], integral quadratic constraints [30] and SOCPs [45].

Bibliography

[1] C.B. Barber, D.P. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, 1996.

[2] A. Bemporad. Reducing conservativeness in predictive control of constrained systems with disturbances. In *Proceedings of 37th Conference on Decision and Control*, pages 1384–1389, 1998.

[3] A. Bemporad and A. Garulli. Output-feedback predictive control of constrained linear systems via set-membership state estimation. *International Journal of Control*, 73(8).

[4] A. Bemporad and A. Garulli. Predictive control via set-membership state estimation for constrained linear systems with disturbances. In *Proceedings of the 4th European Control Conference*, 1997.

[5] A. Bemporad and E. Mosca. Constrained predictive control with terminal ellipsoid constraint and artificial Lyapunov functions. In *Proceedings of the 36th Conference on Decision & Control*, pages 3218–3219, 1997.

[6] D.P. Bertsekas. *Dynamic programming and optimal control*, volume one. Athena Scientific, 1995.

[7] J. Biannic. Stability Analysis of Multi-Variable Feedback Systems with Input Saturations. In *UKACC International Conference on Control 98*, 1998.

[8] R.R. Bitmead, M. Gevers, and V. Wertz. *Adaptive Optimal Control : The Thinking Man's GPC*. Prentice Hall, 1990.

[9] F. Blanchini. Set invariance in control. *Automatica*, 35:1747–1767, 1999.

[10] H.H.J Bloemen and T.J.J van den Boom. MPC for Wiener systems. In *Proceedings of the* 38^{th} *Conference on Decision & Control*, pages 4595–4600, 1999.

[11] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. SIAM Studies in Applied Mathematics. SIAM, Philadelphia, Pennsylvania, 1994.

[12] S. Boyd and L. Vandenberghe. Introduction to convex optimization with engineering applications. Course Notes, EE364, 1997. Information Systems Laboratory, Stanford University.

[13] M.L. Brockman and M. Corless. Quadratic boundness of nonlinear dynamical systems. In *Proceedings of the* 34^{th} *Conference on Decision & Control*, pages 504–509, 1995.

[14] C. Burgat, S. Tarbouriech, and M. Klai. An algorithm for estimating the stability domain of linear discrete-time systems with saturations. In *IEEE/SMC Conference*, pages 319–324, 1993.

[15] E.F. Camacho and C. Bordons. *Model Predictive Control*. Springer, 1998.

[16] A. Casavola, M. Giannelli, and E. Mosca. Global predictive stabilization of input-saturated linear systems. In *Proceedings of the American Control Conference*, pages 3271–3275, 1998.

[17] A. Casavola, M. Giannelli, and E. Mosca. Switching supervisory predictive control of input saturated plants. In *Proceedings of the 1998 IEEE International Conference on Control Applications*, pages 1155–1159, 1998.

[18] H. Chen. *Stability and Robustness Considerations in Nonlinear Model Predictive Control*. PhD thesis, Universität Stuttgart, 1997.

[19] C.R. Cutler and B.L. Ramaker. Dynamic matrix control - a computer control algorithm. In *Joint Automatic Control Conference*, volume 1, San Fransisco, California, 1980.

[20] J.M. Gomes da Silva and S. Tarbouriech. Local stabilization of discrete-time linear systems with saturating controls: an LMI-based approach. In *Proceedings of the American Control Conference*, pages 92–96, 1998.

[21] J.A. De Doná. *Input Constrained Linear Control.* PhD thesis, The University of Newcastle, Australia, 2000.

[22] C.E. Garcia and A.M. Morshedi. Quadratic programming solution of dynamic matrix control (QDMC). *Chemical Engineering Communications*, 46:73–87, 1986.

[23] C.E. Garcia, D.M. Prett, and M. Morari. Model predictive control: Theory and practice - a survey. *Automatica*, 25(3):335–348, 1989.

[24] The geometry center. Quickhull. `http://www.geom.umn.edu/software/qhull/`.

[25] L. El Ghaoui and G. Calafiore. Worst-Case State Prediction under Structured Uncertainty. In *Proceedings of the American Control Conference*, pages 3402–3406, 1999.

[26] T. Glad and L. Ljung. *Control Theory - Multivariable and Nonlinear Methods.* Taylor & Francis, 2000.

[27] K.C. Goh, M.G. Safonov, and G.P Papavassilopoulos. A global optimization approach for the BMI problem. In *Proceedings of the 33rd Conference on Decision and Control*, pages 2009–2014, 1994.

[28] K.C. Goh, L. Turan, M.G. Safonov, G.P Papavassilopoulos, and J.H. Ly. Biaffine matrix inequality properties and computational methods. In *Proceedings of the American Control Conference*, pages 850–855, 1994.

[29] P.O. Gutman and M. Cwikel. An algorithm to find maximal state constraint sets for discrete-time linear dynamical systems with bounded controls and states. *IEEE Transactions on Automatic Control*, 32(3):251–254, March 1987.

[30] A. Hansson. Efficient solution of linear matrix inequalities for integral quadratic constraints. In *Proceedings of the* 39^{th} *Conference on Decision & Control*, 2000.

[31] A. Hansson. A primal-dual interior-point method for robust optimal control of linear discrete-time systems. *IEEE Transactions on Automatic Control*, 45 (9):1639–1655, September 2000.

[32] H. Hindi and S. Boyd. Analysis of linear systems with saturation using convex optimization. In *Proceedings of 37th Conference on Decision and Control*, pages 903–908, 1998.

[33] T. Hu and Z. Lin. An Analysis and Design Method for Linear Systems Subject to Actuator Saturation and Disturbance. In *Proceedings of the American Control Conference*, pages 725–729, 2000.

[34] M. Johansson. *Piecewise Linear Control Systems.* PhD thesis, Departement of Automatic Control, Lund Institute of Technology, 1999.

[35] S.S. Keerthi and E.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, May 1988.

[36] D.L. Kleinman. Stabilizing a discrete, constant, linear system with application to iterative methods for solving the Riccati equation. *IEEE Transactions on Automatic Control*, 19(3):252–254, 1974.

[37] M.V. Kothare, V. Balakrishnan, and M. Morari. Robust constrained model predictive control using linear matrix inequilities. In *Proceedings of the American Control Conference*, pages 440–444, 1994.

[38] M.V. Kothare, V. Balakrishnan, and M. Morari. Robust constrained model predictive control using linear matrix inequilities. *Automatica*, 32(10):1361–1379, 1996.

[39] A. Kurzhanski and I. Vályi. *Ellipsoidal Calculus for Estimation and Control*. Systems & Control: Foundations and Applications. Birkhäuser, 1997.

[40] E.B. Lee and L. Markus. *Foundations of Optimal Control Theory*. The SIAM Series in Applied Mathematics. John Wiley & Sons, 1968.

[41] J-W. Lee. Exponential stability of constrained receding horizon control with terminal ellipsoid constraints. *IEEE Transactions on Automatic Control*, 45 (1):83–88, 2000.

[42] J.H. Lee. Worst-case formulations of model predictive control for systems with bounded parameters. *Automatica*, 33(5):763–781, 1997.

[43] Y.I. Lee and B. Kouvaritakis. Stabilizable regions of receding horizon predictive control with input constraints. *Systems & Control Letters*, 38:13–20, 1999.

[44] M. S. Lobo, L. Vandenberghe, and S. Boyd. *SOCP : Software for Second-order Cone Programming - User's Guide*. Information Systems Laboratory, Electrical Engineering Departement, Stanford University, beta version edition.

[45] M.S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

[46] J Löfberg. Backstepping with local LQ performance and global approximation of quadratic performance. In *American Control Conference 2000*, Chicago, USA, July 2000.

[47] J. Löfberg. Feasibility analysis of MPC with ellipsoidal terminal state constraints. In *Proceedings Reglermöte*, 2000.

[48] J. Löfberg. A stabilizing MPC algorithm using performance bounds from saturated linear feedback. In *Proceedings of the* 39^{th} *Conference on Decision & Control*, 2000.

[49] J. Löfberg. YALMIP : A Matlab interface to SP, MAXDET and SOCP. Technical Report LiTH-ISY-R-2328, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, January 2001.

[50] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

[51] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming.* SIAM Studies in Applied Mathematics. SIAM, Philadelphia, Pennsylvania, 1993.

[52] C. Pittet, S. Tarbouriech, and C. Burgat. Stability regions for linear systems with saturating controls via circle and popov criteria. In *Proceedings of the 36th Conference on Decision & Control*, pages 4518–4523, 1997.

[53] J.A. Primbs and V. Nevistić. A framework for robustness analysis of constrained finite receding horizon control. In *Proceedings of the American Control Conference*, pages 2718–2722, 1998.

[54] A.I. Propoi. Use of linear programming methods for synthesizing sampled-data automatic systems. *Automation and Remote Control*, 24(7):837–844, 1963.

[55] K.Z. Qi and D. G. Fisher. Robust stability of model predictive control. In *Proceedings of the American Control Conference*, pages 3258–3262, 1994.

[56] J.B. Rawlings. Tutorial: Model predictive control technology. In *Proceedings of the American Control Conference*, pages 662–676, 1999.

[57] J.B. Rawlings and K.R. Muske. The stability of constrained receding horizon control. *IEEE Transactions on Automatic Control*, 38(10):1512–1516, 1993.

[58] J. Richalet, A. Rault, J.L. Testud, and J. Papon. Model predictive heuristic control: Applications to industrial processes. *Automatica*, 14:413–428, 1978.

[59] A. Saberi, A. Stoorvogel, and P. Sannuti. *Control of Linear Systems with Regulation and Input Constraints.* Springer, 2000.

[60] J. Schuurmans and J.A. Rossiter. Robust predictive control using tight sets of predicted sets. *Control Theory and Applications*, 147(1):13–18, 2000.

[61] F.C. Schweppe. Recursive state estimation: Unknown but bounded errors and system states. *IEEE Transactions on Automatic Control*, 13(1):22–28, February 1968.

[62] P.O.M. Scokaert and D.Q. Mayne. Min-max feedback model predictive control for constrained linear systems. *IEEE Transactions on Automatic Control*, 43 (8):1136–1142, 1998.

[63] P.O.M. Scokaert and J.B. Rawlings. Constrained linear quadratic control. *IEEE Transactions on Automatic Control*, 43(8):1163–1169, 1998.

[64] P. Spångéus. Hybrid control using LP and LMI methods - some applications. Licenciate thesis LIU-TEK-LIC-1998:59, Departement of Electrical Engineering, Linköpings universitet, Sweden, 1998.

[65] L. Vandenberghe and S. Boyd. *SP - Software for Semidefinite Programming.* Information Systems Laboratory, Electrical Engineering Departement, Stanford University, version 1.0 edition.

[66] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38: 49–95, March 1996.

[67] G.F. Wredenhagen and P.R Bélanger. Piecewise-linear LQ control for systems with input constraints. *Automatica*, 30(3):403–416, 1994.

[68] S.J. Wright. Applying new optimization algorithms to model predictive control. *AIChE Symposium Series No. 316*, 93:147–155, 1997.

[69] S.P. Wu, L. Vandenberghe, and S. Boyd. *MAXDET -Software for Determinant Maximization Problems - User's Guide.* Information Systems Laboratory, Electrical Engineering Departement, Stanford University, alpha version edition.

[70] A. Zheng. *Robust Control of Systems Subject to Constraints.* PhD thesis, California Institute of Technonlogy, Pasadena, California, 1995.

[71] G. M. Ziegler. *Lectures on Polytopes.* Graduate Texts in Mathematics. Springer-Verlag, New York, 1994.

Index