Doctoral theses at NTNU, 2010:201

Francesco Scibilia

Explicit Model Predictive Control: Solutions Via Computational Geometry

Doctoral Thesis

Francesco Scibilia

ISBN 978-82-471-2381-2 (printed ver.)
ISBN 978-82-471-2382-9 (electronic ver.)
ISSN 1503-8181

Doctoral theses at NTNU, 2010:201

NTNU
Norwegian University of Science and Technology
Thesis for the degree of philosophiae doctor
Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of Science and Technology

NTNU

NTNU
Norwegian University of Science and Technology

Francesco Scibilia

Explicit Model Predictive Control: Solutions Via Computational Geometry

Thesis for the degree of philosophiae doctor

Trondheim, November 2010

Norwegian University of
Science and Technology
Faculty of Information Technology, Mathematics and Electrical
Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of philosophiae doctor

Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Engineering Cybernetics

©Francesco Scibilia

ISBN 978-82-471-2381-2 (printed ver.)
ISBN 978-82-471-2382-9 (electronic ver.)
ISSN 1503-8181

Doctoral Theses at NTNU, 2010:201

Printed by Tapir Uttrykk

Summary

The thesis is mainly focused on issues involved with explicit model predictive control approaches. Conventional model predictive control (MPC) implementation requires at each sampling time the solution of an open-loop optimal control problem with the current state as the initial condition of the optimization. Formulating the MPC problem as a multi-parametric programming problem, the online optimization effort can be moved offline and the optimal control law given as an explicitly defined piecewise affine (PWA) function with dependence on the current state. The domain where the PWA function is defined corresponds to the feasible set which is partitioned into convex regions. This makes explicit MPC solutions into promising approaches to extend the scope of applicability of MPC schemes. The online computation reduces to simple evaluations of a PWA function, allowing implementations on simple hardware and with fast sampling rates. Furthermore, the closed form of the MPC solutions allows offline analysis of the performance, providing additional insight of the controller behavior.

However, explicit MPC implementations may still be prohibitively costly for large optimization problems. The offline computational effort needed to solve the multi-parametric optimization problem may be discouraging, and even the online computation needed to evaluate a complex PWA controller may cause difficulties if low-cost hardware is used.

The first contribution of this thesis is to propose a technique for computing approximate explicit MPC solutions for the cases where optimal explicit MPC solutions are impractical due to the offline computational effort needed and their complexity for online evaluations. This technique is based on computational geometry, a branch of computer science which focuses heavily on computational complexity since the algorithms are intended to be used on large data-sets. The approximate solution is suboptimal only over the subregion of the feasible set where constraints are active. In this subregion, the ineffective optimal explicit MPC solution is replaced by an approximation based on Delaunay tessellations and is computed from a finite number of samples of the exact solution. Finer tessellations can be obtained in order to achieve a desired level of accuracy.

Successively, the thesis presents a twofold contribution concerned with the computation of feasible sets for MPC and their suitable approximations. First, an alternative approach is suggested for computing the feasible set which uses set relations instead of the conventional orthogonal projection. The approach can be implemented incrementally on the length of the MPC prediction horizon, and proves to be computationally less demanding than the standard approach. Thereafter, an algorithm for computing suitable inner approximations of the feasible set is proposed, which constitutes the main contribution. Such approximations are characterized by simpler representations and preserve the essential properties of the feasible set as convexity, positive invariance, inclusion of the set of expected initial states. This contribution is particularly important in the context of finding less complex suboptimal explicit MPC solutions, where the complexity of the feasible set plays a decisive role.
The last part of the thesis is concerned with robustness of nominal explicit MPC solutions to model uncertainty. In the presence of model mismatch, when the controller designed using the nominal model is applied to the real plant, the feasible set may lose its invariance property, and this means violation of constraints. Also, since the PWA control law is designed only over the feasible set, there is the technical problem that the control action is undefined if the state moves outside of this set. To deal with this issue, a tool is proposed to analyze how uncertainty on the model affects the PWA control law computed using the nominal model. Given the linear system describing the plant and the PWA control law, the algorithm presented considers the polytopic model uncertainty and constructs the maximal robust feasible set, i.e. the largest subset of the feasible set which is guaranteed to be feasible for any model in the family of models described by the polytopic uncertainty.
The appendix of the thesis contains two additional contributions which are only marginally related to the main theme of the thesis. MPC approaches are often implemented as state feedback controllers. The state variables are not always measured, and in these cases a state estimation approach has to be adopted to obtain the state from the measurements. The two contributions deal with state estimation in two different applications, but not with the explicit goal of being used in MPC approaches.

Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) for the partial fulfillment of the requirements for the degree of philosophiae doctor.
This doctoral work has been mainly performed at the Department of Engineering Cybernetics, NTNU, Trondheim, Norway, during the period from August 2006 to May 2010, with Professor Morten Hovd as supervisor.
Professor Robert R. Bitmead has been advisor during my six-month visit to the UCSD Jacobs School of Engineering, San Diego, California, USA, from August 2008 to February 2009.
Professor Sorin Olaru has been advisor during my two two-week visits to the Automatic Control Department of SUPELEC Systems Sciences, Paris, France, in April 2008 and in May-April 2009.
The work has been funded by the Research Council of Norway (NFR) through the Enhanced Model Predictive Control project, and partially by the Design of Advanced Controllers for Economic, Robust and Safe Manufacturing Performance (CONNECT) european project.

Acknowledgements

I owe my deepest gratitude to a lot of people, perhaps too many to exhaustively mention, who have contributed with generous friendship, collaboration and support in a number of ways to make possible this journey, an extraordinary work and life experience.
First of all, I would like to thank my supervisor, Professor Morten Hovd. I offer him my sincerest gratitude for having granted me, in the first place, the possibility to pursue a Ph.D. degree, and then for his varied support and encouragement during my studies. He has taken an active interest in my work, being always keen to shares his ideas and knowledge, and offer his insight, constructive suggestions, comments and criticism.
The environment at the Department of Engineering Cybernetics at NTNU has been wonderful and enjoyable. This is thanks to the many good colleagues and great friends, with whom I have had the privilege and the pleasure of working with and doing a lot of extra-activities. I would like to express my profound gratitude to all of them, particularly to Giancarlo Marafioti, my mate in many adventures, Luca Pivano and Ilaria Canova Calori, who have given me invaluable friendship since the beginning, and Johannes Tjønnås, Aksel Andreas Transeth, Alexey Pavlov, Hardy Bonatua Siahaan, Roberto Galeazzi, Mernout Burger, Milan Milovanovic, Esten Grøtli, Christian Holden, Jørgen Spjøtvold, Vidar Gunnerud.
My gratitude goes also to the capable staff at the department for never making all the administrative procedures and paper work never an issue.
I would also like to acknowledge and thank Stewart Clark at NTNU for his editorial assistance on this thesis.

In San Diego, I received the warmest welcome from Professor Robert Bitmead, my advisor at UCSD, and his kind wife, Jan. I heartily thank them for the generous hospitality, all the support and friendship. I am grateful for all the inspiring and invaluable conversations with Robert, who always willingly shared his deep and vast knowledge with me. I feel honored to have worked with him.
I was also fortunate to meet great friends and colleagues there at UCSD, with whom I have shared lot of unforgettable moments. Particularly, I would like to thank

Javier Urzay, Baldomero Alonso-Latorre, Marco Antonio Escobar Acevedo, Sergi Hernandez, Pablo Martinez-Legazpi Aguiló.

My sincerest thanks go to Professor Sorin Olaru, my advisor at SUPELEC, Paris. He has willingly shared with me his knowledge, and has been always helpful and supporting in many ways. I am deeply grateful for all the interesting discussions and exchange of ideas, which have been of great importance for my understanding and work. Beyond the work cooperation, I am grateful we also got to share a good friendship.
During my visits at SUPELEC I met many colleagues who have helped me and have made my stays enjoyable. Particularly, I would like to thank Anamaria Luca.

I would like to thank all my good old friends from Italy, particularly Salvatore Marafioti, Francesco Cristofaro, Davide Cristofaro, Francesco Maragno, Antonio Sanso. My sincerest gratitude also goes to my close friend Anna Macedonio for having listened and been a supporter when I needed it.

Last, but not least, I would like to thank my whole family for having been always my secure harbor, where to recover the consciousness of what really matters in life. Mamma and Papà, this is for you, for all the efforts in the last 30 years!
Of course, I cannot complete these few lines without explicitly express my deepest gratitude to my two little sisters, Filomena and Sofia, and to my brother, Girolamo, for having always believed in me and encouraged. All the time we have chatted via the Internet, distances between us became meaningless, and I was home.

Francesco Scibila
Trondheim, October 2010

Contents

Chapter 1

Introduction

Here the reader is introduced to the thematic content discussed in this thesis. Intentionally, the presentation of ideas and concepts in this chapter is kept at a rather colloquial level, rather than being mathematically rigorous and theoretical. The reader will also notice the absence of any literature references in this chapter, since all the information presented is quite general and easily available, for example, using a simple search on the Internet. Complete references to the literature will be given in the following chapters.

1.1 Model Predictive Control

Imagine yourself as a student in a classroom following a lecture. The professor has just finished teaching, and it is time to head home. To do so, you have to exit the classroom through the exit door on the other side of the classroom. A pretty straightforward task that you would likely do without thinking too much about it. However, this simple goal comprises several issues that you, more or less unconsciously, consider: reasonably you want to walk towards the door minimizing the distance; on the way, you do not want to walk over desks or chairs, neither to crash into your classmates; you take into account the physical limitations of your body, like the fact that in one step you can cover one meter or so. The intrinsic mental mechanism used to do this is "prediction": you predict the behavior of your body moving in the room and based on your observation of the surrounding environment you decide and adjust your actions. In other words, you face an objective which you want to achieve in some sense optimally, while satisfying some constraints and limitations. You achieve this objective by deciding and adjusting your optimal ac-

tion “online” based on your observation. Many circumstances in our lives can be seen similarly, and we solve them naturally without having too much awareness.
Model predictive control is an advanced control approach which offers a framework where the control problem can be formulated and solved in a similar “natural” fashion. A model of the system to be controlled is used to predict the future behavior of the system, the control action is obtained as the optimizer of a suitable objective cost function which is minimized such that certain constraints are satisfied, the control action is adjusted online observing some measured variables.
Suppose now you are still in the classroom and the fire alarm goes off, you need to exit the building quickly (do not panic, it is only a fire drill). There are several exits, but of course you want to go to the nearest emergency exit, following the shortest path. This is a situation where you do not have much time to think about how to obtain your goal optimally. Indeed, it would be helpful to have an emergency plan of the building: once you have located the room you are in, then you can simply look at the optimal way to the nearest emergency exit. In this situation, the emergency plan of the building represents an “explicit” optimal solution to your objective, which has been solved beforehand (“offline”) for all the possible rooms where you could be in the building so that the only online effort is to look it up on a map.
In a similar fashion, explicit model predictive control approaches explicitly provide the optimal solutions to the control problem. The model predictive control optimization problem is formulated parametrically in the measured variables and solved offline. The optimal control action is then given as an explicit function of the measured variables, so that the online effort reduces to just look at the optimal control law for the current measurements and evaluate it.

1.1.1 MPC Idea

Model predictive control (MPC), or receding horizon control, is one of the most advanced control approaches which, in the last few decades, has became the leading industrial control technology for systems with constraints.
MPC approaches determine a sequence of optimal control actions (inputs) over a future time horizon in order to optimize the performance of the controlled system, expressed in terms of a cost function. The optimization is based on an internal mathematical model which, given the current measurements, predict the future behavior of the real system with respect to changes in the control inputs. Once the sequence of optimal control inputs has been determined, only the first element is actually implemented and the optimization is repeated at the next time interval with the new measurements and over the shifted horizon. This feedback mechanism of the MPC compensates for the prediction error due to structural mismatch between

the internal model and the real system as well as for disturbances and measurement noise.
The main advantage which makes MPC industrially desirable is that it can routinely take into account constraints in the control problem. This feature is important for several reasons.

- The possibility to explicitly express constraints in the problem formulation offers a natural way to state complex control objectives.
- Often the best performance, which may correspond to the most efficient or profitable operation, is obtained when the system is made to operate near the constraints.
- In the presence of actuator saturations, a control approach that is aware of the constraints never generates control inputs beyond the saturation values, and this removes the wind-up problem.

In addition, MPC approaches have the advantage of naturally handling multivariable control problems and systems with complex dynamics (like systems with long time delays, for example). MPC approaches are powerful and robust (more than the standard PID control), and their relative ease to configure and tune allows remarkably short pay-back time.

The idea underlying MPC is illustrated in Figure 1.1 where a single-input single-output system is considered. At the current time t, the measured output of the system is $y(t)$. The output is required to follow the reference signal y_r. The figure also gives the previous history of the output trajectory and of the applied input, which is supposed to be subject to a saturation constraint. The prediction horizon length is N. Along the horizon the predicted output is y_k, obtained applying the optimal input sequence u_k^*, $k = t+1, ..., t+N$ to the internal model. Only the first element of the optimal input sequence, i.e. u_{t+1}^*, is applied to the system, and a new optimal input sequence is computed at the next time interval starting from the new measurement $y(t+1)$ and over the shifted horizon $t+N+1$.

1.1.2 Implicit MPC Implementation

The traditional implementation approach for MPC relies on the use of a real-time optimization solver, which is required to compute the updated optimal control input sequence for each new set of measurements. Real-time optimization solvers are in general complex computing machines. Although computational speed and optimization algorithms are continuously improving, traditionally such solvers have

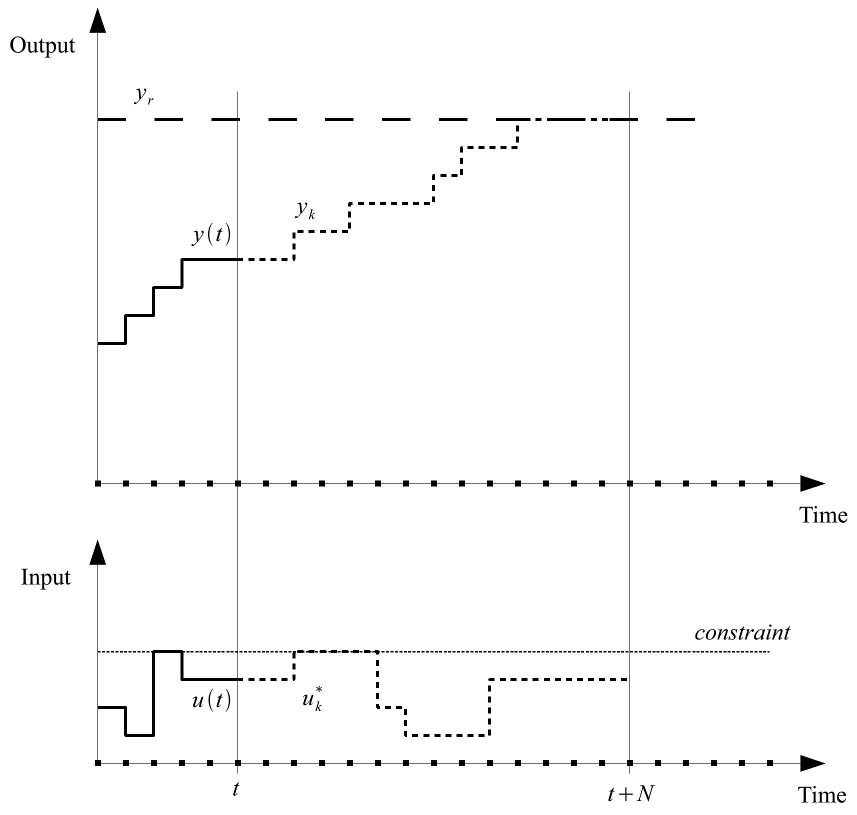


Figure 1.1: Model predictive control idea.

only been able to handle relatively low control input update rates. Therefore, conventional MPC applications have been limited to situations which, in some sense, justify the cost of such hardware and software and which allow a sufficient time span for solving the overall optimization problem. For example, MPC approaches have been very successful in the chemical process industry, where it is common to find complex constrained multivariable control problems with typical time constants of the systems ranging from hours to days, or even weeks.
However, cost and technical reasons prevent the use of real-time optimization solvers in several application areas where it is as well possible to find control problems which could be conveniently formulated as MPC problems. For example, in automotive and aerospace industries often the control strategy has to deal with systems subject to actuator limitations, which need to be sampled and controlled in the range of milli- or microseconds. These short time spans usually limit the complexity of the overall optimization problem (only short horizon and/or simple internal models can be considered) or prohibit the use of MPC approaches at all. Even if modern advances in computing technology allow always faster sampling rates, a critical issue remains the reliability and verifiability of the control algorithm. The main problem here is given by the "implicit" nature of such MPC implementation: the optimal values of the control action are determined as numerical values at each time interval, without any physical or mathematical knowledge of the governing control law. This is undesirable in situations where the possibility to analyze and verify the controller offline is a key issue, like in critical safety applications (for example, the anti-lock braking system, or ABS, in the automotive industry).
Moreover, the implementations via real-time solvers are not well suited for all the situations which require portable and/or embedded control devices.

1.1.3 Explicit MPC Implementation

A different and more recent MPC implementation approach is based on multi-parametric programming. Multi-parametric programming is an optimization technology that allows the optimal solution of an optimization problem to be determined as an explicit function of certain varying parameters. Therefore, multi-parametric programming avoids the need to solve a new optimization problem when the parameter changes, since the optimal solution can readily be updated using the pre-computed function.
In the context of MPC, multi-parametric programming can be used to obtain the optimal control inputs as an explicit function of the measurements, considering these as the parameters of the optimization problem. This allows the online computational effort to be reduced to a series of function evaluations, eliminating the need

of a real-time optimization solver. Commonly, explicit MPC formulations provide the explicit optimal solution as a piecewise function of the system state variables[1]. The typical domain of interest is a subset of the state space, which is partitioned in a finite number of regions (referred to as critical regions). For each critical region, a particular state feedback control law yields the optimal value of the control input. All together these control laws form the piecewise function which represents the explicit optimal MPC solution.
Therefore, for the online implementation such an explicit MPC solution only needs to store the piecewise function and, iteratively for each new measurement, locate which critical region contains the current state vector and evaluate the corresponding feedback control law.
Figure 1.2 illustrates an example of a system with a 2-dimensional state vector and a 1-dimensional control input. The planar domain of the function is partitioned into polygons, each with an associated pre-computed control law.
The concept extends analogously for the general n-dimensional state vector and r-dimensional control input vector (the n-dimensional domain is partitioned in polytopes).

The potential advantages of explicit MPC formulations are extremely promising to extend the scope of applicability of MPC approaches.

- Once the explicit optimal solution has been computed offline, it can be implemented into simple hardware such as a microchip, and can be replicated cheaply for mass production.
- The explicit form of the solution enables MPC to be used on systems which need high control update rates, since function evaluation is usually a very fast operation (compared to solving an optimization problem).
- In contrast to the implicit nature of standard MPC implementations, explicit MPC solutions provide a more accurate and deep intuitive understanding of the control behavior and properties, allowing analysis of performance such as safety verifications.

However, explicit MPC implementation approaches also entail disadvantages. Obtaining the explicit optimal MPC solution amounts to solving (offline) a parametric optimization problem, which is in general a difficult task. Although the problem is tractable and practically solvable for many interesting control applications, the offline computational effort grows fast as the problem size increases. This is the case

[1]State estimation techniques may be used in the situations where the state variables are not fully measured, and explicit MPC solutions that are based on these estimates may be derived.

for long prediction horizon, large number of constraints and high dimensional state and input vectors. Moreover, as the optimization complexity grows, the explicit solution complexity also commonly grows in terms of the number of control laws forming the piecewise function. This means that the storage space needed for the explicit MPC implementation increases and the online function evaluation problem becomes more complex.

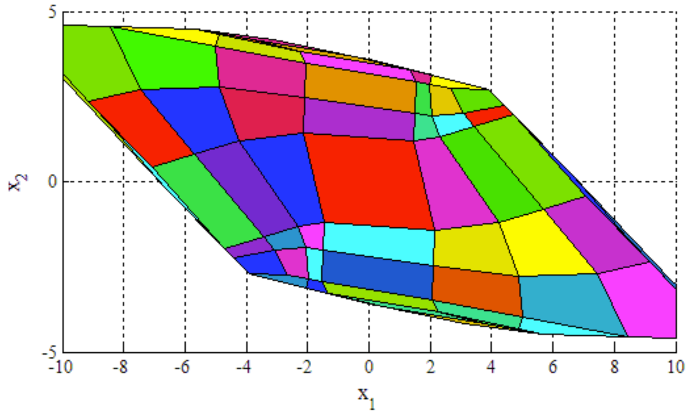
The possible advantages deriving from explicit MPC approaches have attracted a lot the interest in the research community and considerable effort has been put into the development of techniques to deal with the entailed disadvantages.
One research direction is trying to find efficient solutions analyzing the explicit MPC problem from a geometric point of view. This thesis follows this direction.

1.2 Computational Geometry

Many engineering problems embody inherent geometric features, for example distances can be seen as line segments, lands can be represented by polygons, computer networks can be seen as graphs, real objects can be approximated by geometric objects like points, spheres or polyhedra. Therefore, it is often natural to think of solutions from a geometric point of view.
Computational geometry is a subfield of computer science that involves the design and analysis of efficient algorithms for the solution of computational problems which can be stated in terms of geometric objects (points, line segments, polygons, polyhedra, etc.).
Computational geometric techniques are applied in different areas. Some of the application domains where computational geometry has a major impact are briefly described in the following.

- *Computer graphics* – Computer graphics is a wide area which also encompasses computer animation and image processing. This application area is concerned with creating and manipulating images of real or virtual objects, by a computer. Nowadays, computer graphics touch many aspects of daily life, for example on television or on the Internet. Computer graphics is also used to visualize and analyze data, such as underground oil and gas reservoirs for the oil industry.

- *Robotics* – Robotics is concerned with the design, manufacture, application, and structural disposition of robots. As robots can be seen as geometric objects operating in 3-dimensional space (the real world), many related prob-



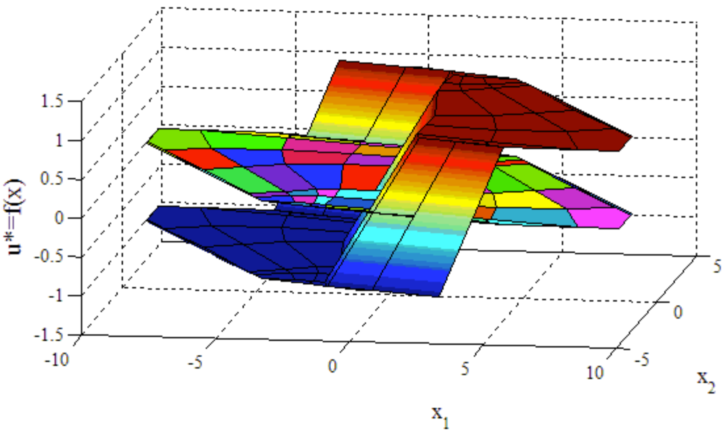


Figure 1.2: Partition of the planar domain into polygons. The piecewise function f(x) defined over this domain comprises different (linear) feedback control laws, one for each polygon of the partition.

lems are naturally seen and solved by a geometric point of view, like for example the motion planning problem, where a robot is asked to find a path in an environment with obstacles.

- *Geographic information systems* – Geographic information systems (GIS) are information systems which store, analyze, and present geographical data. GIS allows users to analyze location information, edit data, maps, and visualize the results of such operations. For example, automotive navigation systems based on the global positioning system (GPS) are nowadays popular applications of GIS.

- *Computer aided design* – Computer aided design (CAD) is concerned with the design of objects by using computer technology. In general, the solutions given by CAD involve more than just shapes, conveying also information such as materials, processes, dimensions, and tolerances. For example, CAD is important in the manufacturing industry where it is also used to assist the manufacturing processes.

Naturally, geometric problems occur in many more application areas. Generally, in all situations where the computational problem possesses a certain geometric nature, geometric algorithms, data structures and techniques from computational geometry may represent the key for efficient and effective solutions.

Explicit model predictive control approaches are concerned with computing a piecewise function with the optimal control input as dependent variables and the system state vector as independent variables. This function is defined over a domain which is partitioned into polygons in the planar case, polyhedra in the 3-dimensional spatial case, and polytopes in the general case. The online implementation effort amounts to locating the polytope of the partition which the current state value lies in, and to evaluating the corresponding feedback law.
A geometric nature is therefore clear.

Devising successful solutions to computational problems with a recognized geometric nature is mostly based on a careful insight into the geometric properties of the problem and on a proper application of geometric concepts, algorithmic techniques and data structures.
To illustrate these principles, three practical problems arising in the application areas mentioned before are discussed in the following. The examples introduce and motivate the importance of fundamental concepts in computational geometry which are particularly relevant for the work in this thesis.

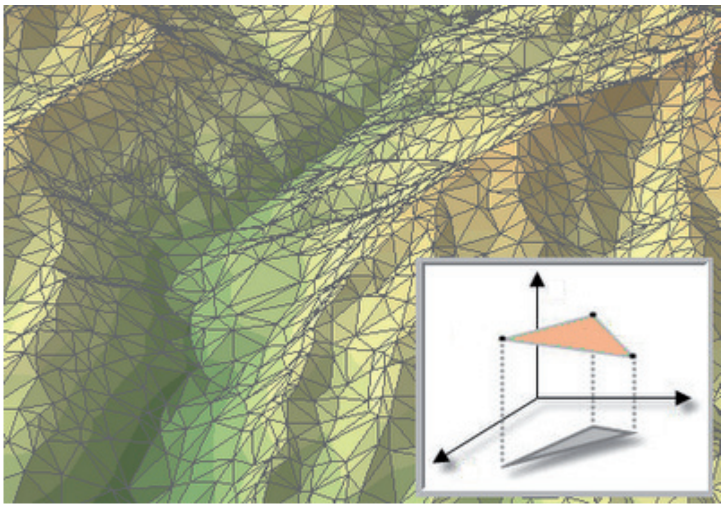


Figure 1.3: A reconstruction of the relief of a terrestrial surface via Delaunay triangulation (www.esri.com).

1.2.1 Surface Reconstruction via Delaunay Triangulation

Consider the common problem in geographic information systems of building a computer model of a terrain (terrestrial surface relief). This problem is important for many reasons: terrains are used to study water flow and distribution; they are also used to analyze the possibility of human settlements and constructions (streets, bridges, buildings) in a region; over large areas, terrains can give insight into weather and climate patterns.

A terrain can be seen as the 3-dimensional plot of a particular function which assigns an elevation for each point on the area of interest (the domain). In practice, we can assume that the value of the function (the elevation) is known only for the limited number of points in the domain where it is measured. Then, the problem is building a model, based only on sampled points, which provides a suitable approximation of the elevation of all the points in the domain.

One of the most successful approaches to the problem comes from computational geometry, and is based on *triangulation*. The planar domain is divided into triangles whose vertices are the sample points[2]. Then, each sample point is lifted to its actual height mapping every triangle in the domain to a triangle in 3-dimensional space. The result is a piecewise (linear) function whose plot approximates the original terrain (Figure 1.3).

[2]It is assumed that the set of sample points is such that can make the triangulation to entirely cover the domain.

Given a set of points, there are several ways to build a triangulation of them. However, in our problem the interest is to establish proper neighborhood relations among the sample points. It is intuitively correct to think that the approximate height of a general point in the domain should be determined by the relatively closest sample points. In this sense, a triangulation with extremely thin triangles (small minimum angle of the triangulation) is likely to lead to a poorer approximation than a triangulation with more regular triangles. A data structure that is extremely well suited for the approximation purpose is the *Delaunay triangulation* of the sample points. One of the main reasons is that the Delaunay triangulation maximizes the minimum angle in the triangulation.
Reconstruction of 3-dimensional terrain is an instance of the wider concept of surface reconstruction via Delaunay structures, which extends to spaces of higher dimension (triangles are generalized by simplices).
The geometric concepts introduced in this section will be considered again in Chapter 3.

1.2.2 Point Location

Consider a dispatch system that is to assign forest rangers to a certain natural park which is partitioned in different regions. The problem considered here is, given a certain point (query point) in the park, which can be, for example, where suspicious smoke has been detected, identify which region of the park contains the point, such that the rangers in that region can be called and sent to check.
The general point location query can be stated as: given a partition of the space into disjoint regions, determine the region where a query point resides. It arises in several application areas like computer graphics, geographic information systems, motion planning, and computer aided design, and often is needed as an ingredient to solve larger geometrical problems.
Suppose here that the surface of the park has been subdivided in triangles (c.f. Section 1.2.1), then the problem is to find the triangle containing the query point. A very simple technique to find such a triangle consists of first selecting a starting triangle and then moving in an iterative way towards the query point. After the starting triangle is given, the technique chooses one of the neighboring triangles in such a way that the distance to the query point decreases. This can be achieved considering the dot products of the normal vectors of the bounding line segments and the vector from the in-circle center of the starting triangle to the query point. Once the new triangle is found, then the procedure is repeated considering this as the starting triangle, until the triangle containing the query point is identified. Figure 1.4 illustrates the idea. This technique is known as *Jump&Walk*, and has the advantages

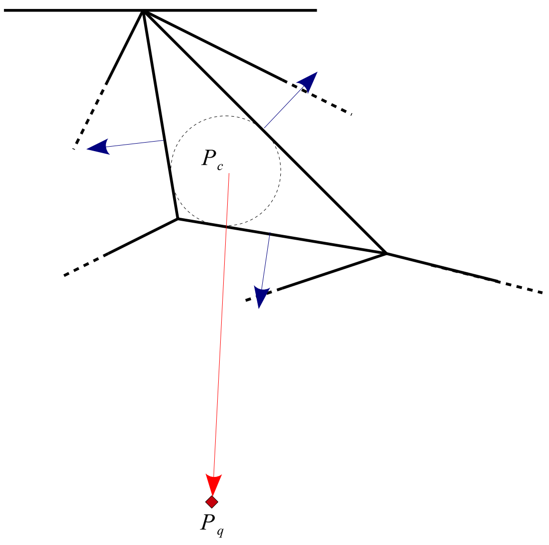


Figure 1.4: Point location algorithm. P_q denotes the query point, P_c is the incircle point of the starting triangle. To determine which neighboring triangle is closest to P_q, the dot products of the vector from P_c to P_q and the normal vectors of all bounding line segments are computed. The line segment whose normal vector results in the largest dot product is chosen, and the corresponding triangle is taken as the next step towards the query point.

of being extremely simple to implement and not requiring any preprocessing of the partition or complex supporting data structures.
The geometric concepts introduced in this section will be considered again in Chapter 3.

1.2.3 Convex Hull

Consider the problem of obtaining a certain color by mixing some basic colors. This is a problem that can arise in image processing or computer-aided design, even if probably the first thought that comes to mind is the less technological scene of a painter mixing his/her colors on a palette.
According to the theory of colors, all the colors can be obtained by mixing different proportions of the three primal spectral colors: red, green and blue. Assume that we have the colors red and green, then the color yellow can be obtained mixing them with an equal proportion. How geometry comes into play becomes clear from

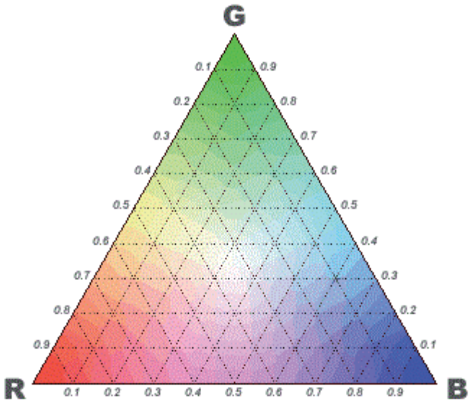


Figure 1.5: Color triangle. The primal spectral colors are red, green and blue. Mixing with different ratios the primary colors all the other colors can be produced.

associating a point in the plane for each primary color, namely R, G and B (vertices of the triangle in Figure 1.5). Then, mixing red and green with a ratio of $1:1$ gives the color yellow represented by the point $Y := 0.5R + 0.5G$ (which is the point on the segment $\overline{RG}$ such that $\mathrm{dist}(R,Y) : \mathrm{dist}(Y,G) = 1:1$, where dist refers to the distance between two points). For different ratios of the component R and G, different colors can be produced (all the colors represented by points on the line segment $\overline{RG}$). Adding to the mixing colors the blue (point B), all the colors in the visible-color spectrum can be produced mixing with different ratios R, G and B, as illustrated in Figure 1.5. The triangle with vertices R, G and B is called the *convex hull* of the points R, G and B.
In general, we can have n base components represented by points $P_1, P_2, ..., P_n$ which can be mixed with ratio $r_1 : r_2 : ... : r_n$. Consider $R = \sum_{i=1}^{n} r_i$ and the coefficients $\lambda_i = r_i/R$, which by construction satisfy $\sum_{i=1}^{n} \lambda_i = 1$ and $\lambda_i \geq 0$ for all i. Then, the mixture produced by a given ratio is represented by $P_d = \sum_{i=1}^{n} \lambda_i P_i$, which is called *convex combination*. The union of all possible convex combinations of a set of points is the convex hull of the points. Therefore, the convex hull represents all the possible mixtures that can be produced by mixing the base components.
An intuitive picture of what constitutes the convex hull of a general set of points can be visualized by thinking of the points like nails sticking out of a planar piece of wood. If an elastic rubber band is held around the nails, and then released, it will snap around minimizing its length. Then, the area enclosed by the rubber band represents the convex hull of the points/nails.
Finding the convex hull of a set of points, in general dimension, is one of the basic

problems in computational geometry.
The geometric concepts introduced in this section will be used in several places in the thesis, and particularly in Chapter 5.

1.3 Thesis Organization and Contributions

The organization of the remaining part of the thesis and the main contributions are stated in the following.
This thesis is written on the basis of a collection of papers. A preliminary chapter provides the common theoretical background, thereafter emphasis has been put on keeping each chapter self-contained. Therefore, some repetition is necessary.

- *Chapter 2* – This chapter provides a common theoretical framework necessary for the arguments in the thesis. It introduces the class of systems descriptions considered, the class of linear systems with constraints, which is probably the most important and widely used one in practice. For the control and analysis of this class of systems, the theory of positive invariant sets has been shown to be particularly important, and the relevant concepts used in the thesis are given here. The chapter thereafter gives a background on polyhedral sets, and in particular polytopic sets, which are the geometrical objects playing the major role in the work in this thesis. Finally, an overview of the MPC problem is given. It is shown how to formulate MPC as a quadratic program and as a multi-parametric quadratic program. Possible extensions of the basic MPC problem considered are also briefly discussed.

- *Chapter 3* – A relevant problem with explicit MPC approaches is that for large dimensional problems, coding and implementing the exact explicit solution may be excessively demanding for the hardware available. In these cases, approximation is the practical way for effective implementations. The main contribution given in this chapter is to propose a technique for computing an approximate piecewise control law which draws its methods from computational geometry. The approximate explicit control law is suboptimal only over the subregion of the feasible set where constraints are active. In this subregion, the problem of computing a suitable suboptimal piecewise control law is related to an important problem in computer aided geometric design, surface reconstruction. Indeed, the technique is based on a fundamental structure in computational geometry theory, Delaunay tessellation, which has been particularly successful in dealing with the surface reconstruction problem. The possible solutions that the approach allows for the point location problem are also discussed.

Most of the material in this chapter has been published or submitted for publication in Scibilia et al. (2010a), Scibilia et al. (2009b) and Hovd et al. (2009).

- *Chapter 4* – In the theoretical framework of MPC, the feasible set plays a decisive role. In fact, explicit MPC approaches provide the optimal solution as a piecewise function over the feasible set. However, computing the feasible set may turn out to be a computationally demanding problem. Moreover, the complexity of representation of the feasible set may constitute a problem in certain situations, particularly, in the context of finding simpler suboptimal explicit solutions. The contributions given by this chapter are in tackling the problems of computing feasible sets and finding suitable approximations thereof. An alternative approach for computing the feasible set is presented, based on set relations and not on the conventional orthogonal projection. The approach can be implemented incrementally on the length of the prediction horizon. This is exploited to design an algorithm to compute suitable inner approximations, which is the main contribution. Such approximations are characterized by simpler representations, and preserve the essential properties of the feasible set such as convexity, positive invariance, inclusion of the set of expected initial states.
  Most of the material in this chapter has been accepted for publication in Scibilia et al. (2010b).

- *Chapter 5* – Piecewise affine feedback control laws represent an important class of controller for linear systems subject to linear constraints, with explicit MPC approaches being probably the most popular techniques to obtain such control laws. In the presence of model mismatch, when the explicit MPC law designed using the nominal model is applied to the real system, the nominal feasible set may lose its invariance property, and this can result in violation of constraints. Moreover, since the controller is designed only over the feasible set, there is the technical problem that the control action is undefined if the state moves outside of the feasible set. The main contribution of this chapter is to present a tool to analyze how uncertainty in the model affects the piecewise affine control law computed using the nominal model. Given the linear system describing the plant and the piecewise affine control law, the algorithm presented considers a polytopic model uncertainty defined by the user and constructs the maximal robust feasible set, i.e. the largest subset of the feasible set which is guaranteed to be feasible for any model in the family of models described by the polytopic uncertainty.
  Most of the material in this chapter has been published in Scibilia et al. (2009a).

- *Chapter 6* – In this chapter conclusions and possible directions for future work are given.

- *Appendix A* – This part contains two particular problems treated during the doctoral studies, but only marginally related to the main theme of the thesis. MPC approaches are often implemented as state feedback controllers. The state variables are not always measured, and in these cases a state estimation approach has to be adopted to obtain the state from the measurements. The two works in this chapter deal with state estimation, but not with the explicit goal to be used in MPC approaches.
  The first work regards the moving horizon state estimation (MHE), which is considered to be the dual problem in the state estimation area of the MPC problem. A practical issue arising with this approach in industrial applications is discussed and a solution is proposed. This work has been published in Scibilia and Hovd (2009).
  The second work considers a particular system in the oil industry: gas-lifted oil well. For control purposes, the state needs to be estimated from the available measurements. The state estimation solution proposed is the use of a nonlinear observer tailored on the particular system under consideration. This work has been published in Scibilia et al. (2008).

1.4 List of Publications

Most of the material presented in this thesis has been published or recently submitted for publication.
The following lists contain the author publications and submitted works produced during the doctoral studies.

Journal papers

a. [Scibilia et al. (2010a)] – **F. Scibilia**, M. Hovd and S. Olaru "An Algorithm for Approximate Explicit Model Predictive Control via Delaunay Tessellations". Submitted to the European Journal of Control, August 2010.

b. [Scibilia et al. (2010b)] – **F. Scibilia**, S. Olaru and M. Hovd "On Feasible Sets for MPC and their Approximations". Automatica, accepted for publication, August 2010.

Conference papers

c. [Scibilia et al. (2008)] – **F. Scibilia**, M. Hovd and R. R. Bitmead “Stabilization of Gas-Lift Oil Wells Using Topside Measurements”. The 17th IFAC World Congress, Seoul, South Korea, July 6-11, 2008.

d. [Scibilia and Hovd (2009)] – **F. Scibilia** and M. Hovd “Multi-Rate Moving Horizon Estimation with Erroneous Infrequent Measurements Recovery”. The 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, Barcelona, Spain, June 30-July 3, 2009.

e. [Scibilia et al. (2009b)] – **F. Scibilia**, S. Olaru and M. Hovd “Approximate Explicit Linear MPC via Delaunay Tessellation”. The 10th European Control Conference, Budapest, Hungary, August 23-26, 2009.

f. [Scibilia et al. (2009a)] – **F. Scibilia**, R. R. Bitmead, S. Olaru and M. Hovd “Maximal Robust Feasible Sets for Constrained Linear Systems Controlled by Piecewise Affine Feedback Laws”. The 7th IEEE International Conference on Control and Automation, Christchurch, New Zealand, December 9-11, 2009.

g. [Hovd et al. (2009)] – M. Hovd, **F. Scibilia**, J. Maciejowski and S. Olaru “Verifying stability of approximate explicit MPC”. The 48th IEEE Conference on Decision and Control, Shanghai, China, December 16-18, 2009.

Publications c. and d. are only marginally related to the main theme of this thesis and thus they are reported as secondary work in the appendix.

Chapter 2

Background

This chapter provides the mathematical background needed for the successive chapters. Most of the definitions and results are well established and can be found in the the literature. Other definitions are slightly adapted for the framework of this thesis.

2.1 Model Setup

This thesis considers systems described by the following discrete-time linear time-invariant model

$$x(t+1) = Ax(t) + Bu(t) \tag{2.1}$$

$$y(t) = Cx(t) \tag{2.2}$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^r$ is the control input and $y \in \mathbb{R}^m$ is the output vector, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times r}$ and $C \in \mathbb{R}^{m\times n}$, and the pair (A, B) is stabilizable.
Full state measurement and no disturbances or model uncertainty are assumed, unless explicitly specified.
The plant is required to satisfy the following output and input constraints

$$y(t) \in \mathcal{Y} \subset \mathbb{R}^m \tag{2.3}$$

$$u(t) \in \mathcal{U} \subset \mathbb{R}^r \tag{2.4}$$

for all times. The sets $\mathcal{Y}, \mathcal{U}$ are considered to be described by linear inequalities on the respective variables. The origin is assumed to be an interior point for both sets.

Remark 1. The state space model (2.1-2.2), commonly used in control theory, often represents only an approximation of the real system, which reasonably behaves in a more complicated nonlinear fashion. Consider the state vector $\mathfrak{x}$ of the real system to evolve according to the following nonlinear differential equation, and the output vector $\mathfrak{y}$ to be determined from the state with the successive static nonlinear equation

$$\frac{d\mathfrak{x}}{dt} = f(\mathfrak{x}, \mathfrak{u}) \tag{2.5}$$

$$\mathfrak{y} = g(\mathfrak{x}) \tag{2.6}$$

where $\mathfrak{u}$ is the input vector. Assume that the system is planned to operate at a certain state $\mathfrak{x} = \mathfrak{x}_0$, with a certain input $\mathfrak{u} = \mathfrak{u}_0$, and output $\mathfrak{y} = \mathfrak{y}_0 = g(\mathfrak{x}_0)$. This is actually a common case in practice, and allows a continuous-time linear model to be obtained by linearization around the operating conditions

$$\frac{d\mathfrak{x}}{dt} = f(\mathfrak{x}_0 + x, \mathfrak{u}_0 + u) \approx f(\mathfrak{x}_0, \mathfrak{u}_0) + \left.\frac{\partial f}{\partial \mathfrak{x}}\right|_{(\mathfrak{x}_0, \mathfrak{u}_0)} x + \left.\frac{\partial f}{\partial \mathfrak{u}}\right|_{(\mathfrak{x}_0, \mathfrak{u}_0)} u \tag{2.7}$$

$$\mathfrak{y} = g(\mathfrak{x}_0 + x) \approx g(\mathfrak{x}_0) + \left.\frac{\partial g}{\partial \mathfrak{x}}\right|_{(\mathfrak{x}_0)} x \tag{2.8}$$

where $\frac{\partial f}{\partial \mathfrak{x}}$, $\frac{\partial f}{\partial \mathfrak{u}}$ and $\frac{\partial g}{\partial \mathfrak{x}}$ are the matrices of partial derivatives (Jacobian). Quadratic and higher-order terms in x and u are neglected.
Since $\mathfrak{x} = \mathfrak{x}_0 + x$ and $\mathfrak{x}_0$ is a particular value of the state, we have $\frac{d\mathfrak{x}}{dt} = \frac{dx}{dt}$, and therefore

$$\frac{dx}{dt} = A_0 x + B_0 u + f(\mathfrak{x}_0, \mathfrak{u}_0) \tag{2.9}$$

$$y = C_0 x \tag{2.10}$$

where the relation $\mathfrak{y} = \mathfrak{y}_0 + y$ is considered, and the Jacobian matrices $\frac{\partial f}{\partial \mathfrak{x}}$, $\frac{\partial f}{\partial \mathfrak{u}}$ and $\frac{\partial g}{\partial \mathfrak{x}}$ are denoted by A_0, B_0 and C_0, respectively. Often the operating point $(\mathfrak{x}_0, \mathfrak{u}_0)$ is an equilibrium point such that the model (2.9-2.10) reduces to the popular continuous-time linear state space model

$$\frac{dx}{dt} = A_0 x + B_0 u \tag{2.11}$$

$$y = C_0 x \tag{2.12}$$

Then, the discrete-time model (2.1-2.2) can be obtained by discretization of the linearized differential equation (2.11-2.12), usually assuming that the input is constant between sampling intervals.
For the cases where the equations (2.5-2.6) are unknown, the state space model (2.1-2.2) can also be obtained via identification of a parameterized black-box model (Ljung (2006), Ikonen and Najim (2002)).

2.2 Positively Invariant Sets

The concept of invariant sets has been shown to play an important role in the control and analysis of constrained systems (Blanchini and Miani (2008), Blanchini (1999), Kerrigan and Maciejowski (2000), Gilbert and Tan (1991)). Given an autonomous dynamic system, a subset of the state space is said to be *positively invariant* if it has the property that if it contains the system state at some time, then it will contain it also at all future times. The presence of constraints on the state variables defines an admissible set in the state space, i.e., the set of states that satisfies the constraints at the present time. Due to the system dynamics not all the trajectories originating from admissible initial states will remain in such a set. Conversely, for any initial condition which belongs to a positively invariant subset of the admissible domain, constraints violations are avoided at future times.

The idea of positive invariance can be formally given considering the autonomous system corresponding to (2.1).

Definition 1 (Positive invariance)**.** The set $\mathbb{S} \subset \mathbb{R}^n$ is said to be *positively invariant* with respect to the autonomous system $x(t+1) = Ax(t)$ if for all $x(0) \in \mathbb{S}$ the system evolution $x(t) \in \mathbb{S}$ for $t > 0$.

The set $\mathbb{S}$ is *maximal* if it also contains all the other positively invariant sets.

In this thesis we are mostly interested in the control implication that such a characterization has. Since model predictive controllers belong to the class of state-feedback controllers, we refer to the following class of control laws

$$u(t) = \Phi(x(t)) \tag{2.13}$$

where $\Phi : \mathbb{R}^n \to \mathbb{R}^r$. As will be clear later on the chapter, model predictive controllers can be expressed in closed form as piecewise affine functions of the state.

The concept of positive invariance extends naturally to systems like (2.1), providing the concept of controlled positive invariance.

Definition 2 (Controlled positive invariance)**.** The set $\mathbb{S} \subset \mathbb{R}^n$ is said to be *controlled positively invariant* with respect to the system (2.1) if there exists a controller of the form (2.13) such that $\mathbb{S}$ is positively invariant for the closed-loop system $x(t+1) = Ax(t) + B\Phi(x(t))$.

Analogously, the set is *maximal* if it also contains all the other controlled positively invariant sets.

The presence of constraints like (2.3-2.4) allows the introduction of a further relevant set characterization.

Definition 3 (Feasible controlled positive invariance)**.** A set $\mathbb{S}_F \subset \mathbb{X}^n$ is said to be *feasibly controlled positively invariant* if it is controlled positively invariant and $u(t) \in \mathcal{U}$ and $y(t) \in \mathcal{Y}$ for all $t \geq 0$.

2.3 Polyhedral Sets

Convex polyhedral sets, and in particular polytopic sets (Blanchini and Miani (2008), Mount (2002), Kvasnica et al. (2006)), are the family of sets which plays the major role in this thesis. The reason is that polyhedral sets arise as natural expressions of physical and/or safety constraints on the system input and output.

Definition 4 (Convex set)**.** A set $\mathbb{S} \subset \mathbb{R}^n$ is said to be *convex* if for all $x_1,\ x_2 \in \mathbb{S}$ then: $\alpha x_1 + (1-\alpha)x_2 \in \mathbb{S}$, for all $0 \leq \alpha \leq 1$.

Definition 5 (Convex hull)**.** The *convex hull* of a set $\mathbb{V} \subset \mathbb{R}^n$ is the smallest convex set containing $\mathbb{V}$. If $\mathbb{V} = V = \left\{v^{(1)}, ..., v^{(n_v)}\right\}$ represents a finite collection of points in $\mathbb{R}^n$, then the convex hull is given by

$$\mathtt{conv}\,(V) = \left\{ \sum_{i=1}^{n_v} \lambda_i v^{(i)} \mid \lambda_i \geq 0,\ \sum_{i=1}^{n_v} \lambda_i = 1 \right\}. \tag{2.14}$$

Definition 6 (Polyhedral set)**.** A convex *polyhedral set* (or polyhedron) is a set which can be expressed as the intersection of a finite number of half-spaces

$$\mathcal{P} = \{x \in \mathbb{R}^n | Dx \leq d\} \tag{2.15}$$

where $D \in \mathbb{R}^{n_{\mathcal{H}} \times n}$ and $d \in \mathbb{R}^{n_{\mathcal{H}}}$, $n_{\mathcal{H}}$ is the number of half-spaces.

Commonly (2.15) is called the $\mathcal{H}$-representation of the set.

We indicate with $D_{(i,j)}$ the (i,j)-element of the matrix D, and $d_{(i)}$ the i-th element of the vector d. Each row $D_{(i,\cdot)}$ and component $d_{(i)}$, $i = 1,..,n_{\mathcal{H}}$, define the half-space $D_{(i,\cdot)}x \leq d_{(i)}$. A half-space $D_{(i,\cdot)}x \leq d_{(i)}$ is said to be *redundant* if the omission of $D_{(i,\cdot)}$ from D and $d_{(i)}$ from d does not change $\mathcal{P}$. A $\mathcal{H}$-representation is *minimal* if there are no redundant half-spaces.
A polyhedral set $\mathcal{P} \subset \mathbb{R}^n$ is said to be *full-dimensional* if there exist $\varepsilon > 0, x_c \in \mathbb{R}^n$ such that $\beta_\varepsilon(x_c) \subset \mathcal{P}$, where $\beta_\varepsilon(x_c)$ is the n-dimensional ε-ball in $\mathbb{R}^n$ centered at x_c: $\beta_\varepsilon(x_c) = \{x \in \mathbb{R}^n | \|x - x_c\| < \varepsilon\}$.
A point $x \in \mathcal{P}$ is called an *interior point* of $\mathcal{P}$ if there exists an $\varepsilon > 0$ such that $\beta_\varepsilon(x)$ is contained in $\mathcal{P}$. The set of all interior points of $\mathcal{P}$ is called the *interior* of $\mathcal{P}$ and is denoted with $\texttt{int}(\mathcal{P})$.

Definition 7 (Partition of a polyhedron)**.** The collection of polyhedral sets $\{\mathcal{R}_i\}_{i=1}^{n_r}$ is called *partition* of a polyhedron $\mathcal{R}$ if $\mathcal{R} = \bigcup_{i=1}^{n_r} \mathcal{R}_i$ and $\texttt{int}(\mathcal{R}_i) \cap \texttt{int}(\mathcal{R}_j) = \emptyset$ for all $i \neq j$. Each $\mathcal{R}_i$ is referred as region of the partition.

Definition 8 (Polytope)**.** A bounded polyhedral set is called a *polytope*.

A full-dimensional polytope $\mathcal{P} \subset \mathbb{R}^n$ characterized by $n_{\mathcal{H}}$ irredundant half-spaces, is bounded by $n_{\mathcal{H}}$ full-dimensional polytopes in $\mathbb{R}^{n-1}$ called *facets*. Each irredundant half-space is associated with a facet, represented as

$$f_i = \left\{x \in \mathcal{P} | D_{(i,\cdot)}x = d_{(i)}\right\} \tag{2.16}$$

where the index i refers to the particular half-space. Facets are polytopes which have facets themselves in $\mathbb{R}^{n-2}$ called *ridges* of $\mathcal{P}$. Every ridge arises as the intersection of two facets (but the intersection of two facets is not in general a ridge). Ridges are once again polytopes whose facets give rise to boundaries of $\mathcal{P}$ in $\mathbb{R}^{n-3}$, and so on. These bounding sub-polytopes are usually referred to as *faces*, specifically k-dimensional faces indicating with k the space $\mathbb{R}^k$ where they are full-dimensional. A 0-dimensional face is called a *vertex*, and consists of a single point in $\mathbb{R}^n$.

The Minkowski-Weyl theorem (Motzkin et al. (1953)) states that every polytope can equivalently be represented as the convex hull of its vertices $V = \left\{v^{(1)}, ..., v^{(n_{\mathcal{V}})}\right\}$

$$\mathcal{P} = \texttt{conv}\,(V) \tag{2.17}$$

where $n_{\mathcal{V}}$ is the number of vertices.
Commonly (2.17) is called the $\mathcal{V}$-representation of the set.
Analogously, a vertex $v^{(i)}$ is said to be *redundant* if the omission of $v^{(i)}$ from V does not change $\mathcal{P}$. A $\mathcal{V}$-representation is *minimal* if there are no redundant vertices.

Any full-dimensional polytope has a *unique* minimal $\mathcal{V}$-representation and a *unique* minimal $\mathcal{H}$-representation.

The *volume* of a polytope $\mathcal{P}$, $\texttt{vol}\,(\mathcal{P}) = \int_{\mathcal{P}} \mathrm{dx}$, indicates the Lebesque measure of $\mathcal{P}$. A full-dimensional polytope in $\mathbb{R}^n$ is characterized by a positive volume (Gritzmann and Klee (1994a)).

Some relevant basic operations with polytopes are defined in the following. Matlab implementations of all the operations mentioned are found in the Multi-Parametric Toolbox (MPT) (Kvasnica et al. (2004)).

Definition 9 (Orthogonal projection)**.** The *orthogonal projection* of a polytope $\mathcal{P} \subset \mathbb{R}^n \times \mathbb{R}^d$ onto $\mathbb{R}^n$ is defined as

$$\Pi_n(\mathcal{P}) = \left\{ x \in \mathbb{R}^n | \exists z \in \mathbb{R}^d,\ \left[x^T\ z^T\right]^T \in \mathcal{P} \right\} \tag{2.18}$$

Definition 10 (Minkowski sum)**.** The *Minkowski sum* of two polytopes $\mathcal{P}$ and $\mathcal{Q}$ is defined as

$$\mathcal{P} \oplus \mathcal{Q} = \{x = p + q |\ p \in \mathcal{P},\ q \in \mathcal{Q}\}\,. \tag{2.19}$$

Definition 11 (Erosion)**.** The *erosion* (or Pontryagin difference) of two polytopes $\mathcal{P}$ and $\mathcal{Q}$ is defined as

$$\mathcal{P} \ominus \mathcal{Q} = \{x |\ x + q \in \mathcal{P},\ \forall q \in \mathcal{Q}\}\,. \tag{2.20}$$

Definition 12 (Set difference)**.** The *set difference* of two polytopes $\mathcal{P}$ and $\mathcal{Q}$ is defined as the set

$$\mathcal{P} \setminus \mathcal{Q} = \{x |\ x \in \mathcal{P}, x \notin \mathcal{Q}\}\,. \tag{2.21}$$

Note that by definition, polytopes are closed sets, i.e. they contain their own boundaries. Rigorously speaking, the set difference of two intersecting polytopes $\mathcal{P}$ and $\mathcal{Q}$ is not a closed set. However, in this work we always refer to the closure of the set difference.

While the orthogonal projection, the Minkowski sum and the erosion are closed operations over the class of convex polyhedral sets, the polyhedral set difference is not. This means that the resulting set might not be a polyhedral region (convex). However, it can always be expressed as a finite union of convex polyhedral regions.

The polyhedral sets considered in this paper result from constraints on the output and input variables expressed as linear inequalities, thus they are naturally given in the $\mathcal{H}$-representation. Finding all the vertices of a given polytope in $\mathcal{H}$-representation is a well-know operation called *vertex enumeration*, and is dual to the convex hull computation (Fukuda (2004)).

Further details and algorithmic implementations of the operations introduced can be found for example in de Berg et al. (2008) or Mount (2002).

Piecewise affine (PWA) functions defined over partitions of polyhedral sets are relevant for this work, particularly PWA functions which are continuous.

Definition 13 (PWA functions over polyhedra)**.** Given a polyhedral set $\mathcal{R} \subseteq \mathbb{R}^{n_x}$ partitioned in polyhedral regions $\mathcal{R} = \bigcup_{i=1}^{n_r} \mathcal{R}_i$, the function $\mu(x) : \mathcal{R} \rightarrow \mathbb{R}^{n_\mu}$ is called *piecewise affine* (PWA) if

$$\mu(x) = L_i x + g_i \; \forall \; x \in \mathcal{R}_i, \tag{2.22}$$

where $L_i \in \mathbb{R}^{n_\mu \times n_x}$, $g_i \in \mathbb{R}^{n_\mu}$ and $i = 1, ..., n_r$.

Definition 14 (Continuous PWA function)**.** The PWA function $\mu(x)$ is called *continuous* if

$$L_i x + g_i = L_j x + g_j \; \forall \; x \in \mathcal{R}_i \cap \mathcal{R}_j, \; i \neq j \tag{2.23}$$

In the following this thesis is restricted to PWA functions defined over polytopes.

2.4 Model Predictive Control

Consider the problem of regulating the discrete-time system (2.1) to the origin. The desired objective is to optimize the performance by minimizing the infinite horizon cost

$$J(x(t), \{u_k\}_{k=0,...,\infty}) = \sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k \tag{2.24}$$

where x_k denotes the predicted state vector at time $t + k$ obtained by applying to (2.1) the k first elements of the input sequence $u_0, ..., u_\infty$, starting from $x_0 = x(t)$. The tuning parameters are the symmetric matrices $Q \succeq 0$ (positive semidefinite) and $R \succ 0$ (positive definite) corresponding to weights on state and input. It is assumed that the pair $\left(\sqrt{Q}, A\right)$ is detectable.

We note that here, as in the following in this thesis, a stage cost $x_k^T Q x_k + u_k^T R u_k$ quadratic in the variables x_k and u_k is considered (2-norm). Stage costs linear in x_k and u_k (1-norm and ∞-norm) have also been investigated in the literature (Rossiter et al. (1995), Jones and Morari (2008), Bemporad et al. (2002a)) as opposed to the quadratic one. This change may lead to some advantages like a reduction in the computational load for the control implementation, however the typical control is not smooth and quadratic cost functions like (2.24) remain the most attractive in practice.

When no constraints are considered, the infinite horizon objective function (2.24) is minimized by the time-invariant state feedback (Naidu (2003))

$$u_k = -Kx_k \tag{2.25}$$

where the matrix K is given by the solution of the discrete-time algebraic Riccati equation (DARE) obtainable by

$$P = (A + BK)^T P (A + BK) + K^T R K + Q \tag{2.26}$$

$$K = \left(R + B^T P B\right)^{-1} B^T P A. \tag{2.27}$$

With the control law (2.25), commonly known as the linear quadratic regulator (LQR), the optimal cost function is given by

$$\sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k = \sum_{k=0}^{\infty} x_k^T (Q + K^T R K) x_k = x_0^T P x_0 \tag{2.28}$$

When constraints on the output and input like (2.3-2.4) are considered, an analytical form of the optimal control law such as (2.27) minimizing the cost function (2.24) does not exist. Therefore, to achieve feedback the (open-loop) minimization of the cost function needs to be performed at each sampling instant when $x(t)$ becomes available, and apply only the first part of the optimal input. The main issue here is that the resulting optimization problem is generally intractable due to the infinite number of optimization variables.
However, it has been shown (Chmielewski and Manousiouthakis (1996), Muske and Rawlings (1993)) that it is possible to optimize the performance over the infinite

horizon with a finite number of optimization variables if the cost function is viewed as composed by two parts:

$$\sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k = \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k + \sum_{k=N}^{\infty} x_k^T Q x_k + u_k^T R u_k \qquad (2.29)$$

where $N < \infty$ corresponds to a chosen horizon. Noticing that after some time the constraints are resolved naturally, and assuming that this happens within the horizon N, the control inputs in the first part are the only optimization variables needed to be considered, since the control inputs in the second part are given by the LQR. Therefore

$$\sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k = \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k + x_N^T P x_N. \qquad (2.30)$$

The term $x_N^T P x_N$ is known as the *terminal cost function*.

The presence of the constraints in the optimization problem results in the concept of *feasible set* (Nocedal and Wright (2006), Boyd and Vandenberghe (2004), Scibilia et al. (2010b)).

Definition 15 (Feasible set)**.** The feasible set, indicated here as $\mathcal{X}_F$, is the largest subset of the state space such that a control action satisfying all the constraints does exist.

Since the constraints considered are expressed by linear inequalities, the feasible set is a polyhedral set.
An optimal input satisfying the constraints is guaranteed to exist for any initial state inside the feasible set.

One of the advantages of having an infinite horizon is that if the detectability assumption on the state is satisfied and if the initial state is chosen inside the feasible set, then nominal closed-loop (exponential) asymptotic stability is ensured (Mayne et al. (2000)). The fact that the infinite horizon cost function can be written as (2.30), allows achieving asymptotic stability by means of a tractable optimization problem also in the presence of constraints, as long as the constraints will not be violated after the end of the chosen horizon N. Therefore, guaranteeing stability is equivalent to guaranteeing that constraints will be respected on the infinite horizon, and not only on the horizon where they are enforced.
One possibility is to select the horizon N that is long enough to guarantee that constraints will not be violated afterwards (Chmielewski and Manousiouthakis (1996)).

However this may result in an excessive computational burden needed to solve the corresponding open-loop optimization problem.
A preferred approach is to modify the open-loop optimization problem introducing a *terminal state constraint*

$$x_N \in \Omega \tag{2.31}$$

where Ω is called the *terminal set*. Generally, the terminal set is chosen to be the maximal output admissible set (MOAS) (Gilbert and Tan (1991)) for the system

$$x(t+1) = (A - BK)x(t). \tag{2.32}$$

Given the nature of the constraints (2.3-2.4), this set is an easily computable polytope, and corresponds to the largest positively invariant set within the state space where the LQR satisfies the constraints.
By means of (2.31), asymptotic stability can be guaranteed for any horizon length.

The introduction of the terminal constraint turns the length of the horizon into a tuning parameter, since this sets the degrees of freedom in the optimization. It follows that the size of the feasible set increases with longer horizons, until the horizon is long enough that the constraints are resolved naturally without the need for the terminal constraint.

We can now formally state the finite horizon MPC optimization problem as follows.

$$\min_{\mathbf{u} \triangleq [\mathbf{u_0},\ldots,\mathbf{u_{N-1}}]} \left\{ J\left(\mathbf{u}, x\left(t\right)\right) = x_N^T P x_N + \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k \right\} \tag{2.33}$$

$$\begin{aligned} \text{s.t.} \quad & x_0 = x\left(t\right), & \\ & x_{k+1} = A x_k + B u_k & \\ & y_k = C x_k & k = 0, 1, ..., N \\ & y_k \in \mathcal{Y}, & k = 1, 2, ..., N \\ & u_k \in \mathcal{U}, & k = 0, 1, ..., N-1 \\ & x_N \in \Omega & \end{aligned} \tag{2.34}$$

For the problem (2.33-2.34), the feasible set is given by

$$\mathcal{X}_F = \left\{ x \in \mathbb{R}^n \middle| \, \exists \, \mathbf{u} \text{ satisfying } (2.34) \right\}. \tag{2.35}$$

The polyhedral set described by (2.35) is actually a polytope due to the nature of the constraints (Scibilia et al. (2010b)).

Since $0 \in \texttt{int}(\mathcal{Y})$ and $0 \in \texttt{int}(\mathcal{U})$, the origin is of course an interior point in the feasible set.

2.4.1 Stability in MPC

MPC stability analysis necessitates the use of Lyapunov theory (Khalil (2000)), since the presence of the constraints makes the closed-loop system nonlinear. As discussed in Mayne et al. (2000), the main idea is to modify the basic MPC concept such that the cost function can be used as Lyapunov function to establish stability. Essentially, the modifications proposed correspond to employing a terminal cost function and/or a terminal constraint (which can be either a terminal equality constraint or a terminal set constraint with an appropriate local stabilizing controller) (Sznaier and Damborg (1987), Keerthi and Gilbert (1988), Rawlings and Muske (1993), Scokaert and Rawlings (1998), Chmielewski and Manousiouthakis (1996), Maciejowski (2002)).
The MPC problem (2.33-2.34) considered in this thesis uses both a terminal cost function and a terminal set constraint, and is the version which attracts most of the attention in the MPC literature. It generally offers better performance when compared with other MPC versions and allows a wider range of control problems to be handled.
Closed-loop stability for the MPC problem (2.33-2.34) is guaranteed by the following theorem.

Theorem 1. *The origin of the system (2.1) in closed-loop with the MPC given by (2.33-2.34) is exponentially stable with domain of attraction the feasible set (2.35).*

Proof. The proof, which can be found in Mayne et al. (2000), demonstrates that the terminal cost function $x_N^T P x_N$, P defined by (2.26), and the terminal constraint $x_N \in \Omega$, Ω chosen as the maximal output admissible set for the system (2.1) in closed-loop with the LQR (2.27) (Gilbert and Tan (1991)), with this LQR as the local stabilizing controller inside Ω, make the cost function a Lyapunov function, establishing exponential stability. $\square$

2.4.2 Quadratic Program Formulation

The cost function in (2.30) can be written as

$$J\left(x(t), \mathbf{u}\right) = \mathbf{x}^T \hat{Q} \mathbf{x} + \mathbf{u}^T \hat{R} \mathbf{u} \tag{2.36}$$

where

$$\mathbf{x} = \begin{bmatrix} x_0 \\ \vdots \\ x_N \end{bmatrix}, \mathbf{u} = \begin{bmatrix} u_0 \\ \vdots \\ u_{N-1} \end{bmatrix}$$

and $\hat{Q} = \texttt{diag}(\texttt{blkdiag}(Q, N), P)$, $\hat{R} = \texttt{blkdiag}(R, N)$.

The function $\texttt{blkdiag}(Q, n)$ indicates the n times block diagonal concatenation of Q, and the function $\texttt{diag}(Q, P)$ indicates the diagonal concatenation of Q and P.

Repeated use of equation (2.1) gives

$$\mathbf{x} = \hat{A}x(t) + \hat{B}\mathbf{u} \tag{2.37}$$

where $\hat{A} \in \mathbb{R}^{n(N+1)\times n}$, $\hat{B} \in \mathbb{R}^{n(N+1)\times rN}$

$$\hat{A} = \begin{bmatrix} I \\ A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}, \hat{B} = \begin{bmatrix} 0 & \cdots & & 0 \\ B & 0 & \cdots & 0 \\ AB & B & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix} \tag{2.38}$$

and I is the $n \times n$ identity matrix and A^N the N-matrix-power of A.
Substituting (2.37) in (2.36) and rearranging gives

$$J(x(t), \mathbf{u}) = \mathbf{u}^T H \mathbf{u} + 2x(t)^T F \mathbf{u} + x(t)^T Y x(t) \tag{2.39}$$

where $H = \hat{B}^T \hat{Q} \hat{B} + \hat{R}$, $F = \hat{A}^T \hat{Q} \hat{B}$ and $Y = \hat{A}^T \hat{Q} \hat{A}$.
Note that $H \succ 0$ since $R \succ 0$.

Consider the polyhedral output and the input constraints sets

$$\mathcal{Y} = \{y \in \mathbb{R}^m | D_{\mathcal{Y}} y \leq d_{\mathcal{Y}}\} \tag{2.40}$$

$$\mathcal{U} = \{u \in \mathbb{R}^r | D_{\mathcal{U}} u \leq d_{\mathcal{U}}\}. \tag{2.41}$$

where $D_{\mathcal{Y}} \in \mathbb{R}^{n_{\mathcal{Y}}\times m}$, $d_{\mathcal{Y}} \in \mathbb{R}^{n_{\mathcal{Y}}}$, $D_{\mathcal{U}} \in \mathbb{R}^{n_{\mathcal{U}}\times r}$, $d_{\mathcal{U}} \in \mathbb{R}^{n_{\mathcal{U}}}$, and $n_{\mathcal{Y}}$, $n_{\mathcal{U}}$ are the numbers of output and input inequalities, respectively.
Using (2.2), the output constraints along the horizon can be expressed as a function of the state

$$\hat{D}_{\mathcal{Y}} \hat{C} \mathbf{x} \leq \hat{d}_{\mathcal{Y}}, \tag{2.42}$$

where $\hat{C} = \texttt{blkdiag}(C, N+1)$,

$$\hat{D}_y = \left[\begin{array}{cc} \mathbb{O}_{(n_y N)\times m} & \texttt{blkdiag}(D_y, N) \end{array} \right], \; \hat{d}_y = \left[\begin{array}{c} d_y \\ \vdots \\ d_y \end{array} \right]$$

and the term $\mathbb{O}_{n\times m}$ is the $n \times m$ zero matrix.

The input constraints along the horizon can be expressed as

$$\hat{D}_u \mathbf{u} \leq \hat{d}_u, \tag{2.43}$$

where $\hat{D}_u = \texttt{blkdiag}(D_u, N)$ and

$$\hat{d}_u = \left[\begin{array}{c} d_u \\ \vdots \\ d_u \end{array} \right].$$

Using (2.37) in (2.42), the output constraints can be written as

$$\hat{D}_y \hat{C} \hat{B} \mathbf{u} \leq \hat{d}_y - \hat{D}_y \hat{C} \hat{A} x(t) \tag{2.44}$$

Combining (2.44) and (2.43) we obtain

$$G' \mathbf{u} \leq w' + E' x\,(t) \tag{2.45}$$

where

$$G' = \left[\begin{array}{c} \hat{D}_y \hat{C} \hat{B} \\ \hat{D}_u \end{array} \right], \; w' = \left[\begin{array}{c} \hat{d}_y \\ \hat{d}_u \end{array} \right], \; E' = \left[\begin{array}{c} -\hat{D}_y \hat{C} \hat{A} \\ \mathbb{O}_{m\times(n_u N)} \end{array} \right].$$

As previously mentioned, the terminal constraint can be obtained considering the closed-loop system

$$x\,(t+1) = A_c x\,(t) \tag{2.46}$$

where we indicate $A_c = A - BK$, K defined in (2.27).
Then, define the set

$$\Omega^{(t)} = \left\{ x \in \mathbb{R}^n \mid D_y C A_c^j x \leq d_y, \; -D_u K A_c^j x \leq d_u \; \forall j = 0, 1, ..., t \right\}. \tag{2.47}$$

This is a polyhedral set, indeed if we chose for example $t = N$ we can use (2.45) to express it as

$$\Omega^{(N)} = \{ x \in \mathbb{R}^n |\, (G'\Lambda - E')\, x \leq w' \} \tag{2.48}$$

where

$$\Lambda = -\begin{bmatrix} K \\ KA_c \\ \vdots \\ KA_c^N \end{bmatrix}.$$

It is known (Gilbert and Tan (1991)) that $\Omega^{(t_2)} \subseteq \Omega^{(t_1)}\ \forall\, 0 < t_1 \leq t_2$.
If $\Omega^{(t)} = \Omega^{(t-1)}$ for some positive t, then $\Omega^{(t)} = \Omega^{(t+1)} = ... = \Omega^{(\infty)} \triangleq \Omega$ is said to be finitely determined.
By arguments in Gilbert and Tan (1991) it can be proven that the asymptotic stability of A_c implies the set Ω to be finitely determined. Therefore, Ω can be determined as described in Algorithm 1.

Algorithm 1: Terminal set

```
  Input: The closed-loop state matrix $A_c$, the polyhedral output and input
         constraints set $\mathcal{Y}$ and $\mathcal{U}$.
  Output: The maximal output admissible set $\Omega$.
1 Set $t = 0$ and express $\Omega^{(t)}$ in polyhedral form as for (2.47);
2 repeat
3     increment $t$ and add the new corresponding constraints;
4 until $\Omega^{(t)} = \Omega^{(t-1)}$ ;
5 set $\Omega = \Omega^{(t-1)}$ and stop.
```

Finite determination guarantees that Algorithm 1 terminates in finite time and gives the polytopic terminal set

$$\Omega = \{x \in \mathbb{R}^n | D_\Omega x \leq d_\Omega\} \tag{2.49}$$

where $D_\Omega \in \mathbb{R}^{n_\Omega \times n}$, $d_\Omega \in \mathbb{R}^{n_\Omega}$, n_Ω is the finite number of half-spaces determining Ω.

Then we can write the terminal constraint as

$$D_\Omega \hat{B}_{[n]}\mathbf{u} \leq d_\Omega - D_\Omega \hat{A}_{[n]} x\,(t) \tag{2.50}$$

where $\hat{A}_{[n]}$ and $\hat{B}_{[n]}$ indicate the last n rows of $\hat{A}$ and $\hat{B}$ respectively.
With (2.50) we can finally write (2.34) in the matrix form

$$G\mathbf{u} \leq w + Ex \tag{2.51}$$

where we indicate $x\,(t)$ as simply x for ease of notation, and

$$G = \begin{bmatrix} G' \\ D_\Omega \hat{B}_{[n]} \end{bmatrix},\ w = \begin{bmatrix} w' \\ d_\Omega \end{bmatrix},\ E = \begin{bmatrix} E' \\ -D_\Omega \hat{A}_{[n]} \end{bmatrix}.$$

From (2.39) and (2.51) we have the MPC quadratic program (QP) formulation:

$$\begin{aligned} V^*(x) = \ & \min_{\mathbf{u}} \mathbf{u}^T H \mathbf{u} + 2x^T F \mathbf{u} \\ & \text{s.t. } G\mathbf{u} \leq w + Ex \end{aligned} \tag{2.52}$$

where the term $x^T Y x$ is removed since it does not influence the optimal argument. The value of the cost function at optimum is simply obtained from (2.52) as $J^*(x) = V^*(x) + x^T Y x$.

2.4.3 Multi-parametric Quadratic Program Formulation

Using the change of variable $\mathbf{z} \triangleq \mathbf{u} + H^{-1}F^T x$, the QP problem (2.52) can be transformed into the equivalent problem

$$\begin{aligned} V_z^*(x) = \ & \min_{\mathbf{z}} \mathbf{z}^T H \mathbf{z} \\ & \text{s.t. } G\mathbf{z} \leq w + Sx \end{aligned} \tag{2.53}$$

where $S \triangleq E + GH^{-1}F^T$, so that the state vector appears only on the right-hand side of the constraints.
Considering the current state vector x as a vector of parameters, the transformed problem (2.53) can be seen as a multi-parametric quadratic problem (mp-QP). In parametric programming, the objective is to obtain the optimal solution $\mathbf{z}^*(x)$ as an explicit function of the parameter x. An explicit solution exists for all the states belonging to the feasible set

$$\mathcal{X}_F = \left\{ x \in \mathbb{R}^n | \exists \mathbf{z} \in \mathbb{R}^{rN} \text{ s.t. } G\mathbf{z} - Sx \leq w \right\} \tag{2.54}$$

To characterize the explicit solution it is instrumental to introduce the concept of active constraint.

Definition 16 (Active constraint)**.** An inequality constraint is said to be an *active constraint* for some x if it holds with equality at the optimum.

From optimization theory, a necessary condition for solution $\mathbf{z}$ to be a global minimizer is for it to satisfy the Karush-Kuhn-Tucker (KKT) conditions. Since $V_z^*(x)$ is

convex, the KKT conditions are also sufficient (Nocedal and Wright (2006), Boyd and Vandenberghe (2004)).
Consider then a state $x_0 \in \mathcal{X}_F$. The optimal solution **z*** of the mp-QP (5.4) corresponding to x_0 will satisfy the KKT conditions

$$\begin{aligned} H\mathbf{z}^* + G^T\lambda &= 0, && (2.55)\\ \lambda\left(G\mathbf{z}^* - w - Sx_0\right) &= 0, && (2.56)\\ \lambda &\geq 0, && (2.57)\\ G\mathbf{z}^* &\leq w + Sx_0, && (2.58)\end{aligned}$$

where $\lambda \in \mathbb{R}^{n_C}$ is the vector of Lagrangian multipliers, $n_C = (N-1)n_{\mathcal{Y}} + Nn_{\mathcal{U}} + n_\Omega$ is the total number of constraints in (2.53)[1].
Solving (2.55) for **z*** we have

$$\mathbf{z}^* = -H^{-1}G^T\lambda \qquad (2.59)$$

that substituted in (2.56) gives

$$\lambda\left(-GH^{-1}G^T\lambda - w - Sx_0\right) = 0. \qquad (2.60)$$

Conditions (2.56) are called the *complementary conditions*; they imply that the Lagrange multiplier associated to a constraint can be strictly positive only when the constraint is active.

Since $H \succ 0$, the optimal solution in x_0 is unique. Let us indicate with $\tilde{G}$, $\tilde{S}$ and $\tilde{w}$ submatrices containing the corresponding rows of G, S and w of active constraints. It follows that

$$\tilde{G}\mathbf{z}^* = \tilde{S}x_0 + \tilde{w}. \qquad (2.61)$$

Assume that the rows of $\tilde{G}$ are linearly independent (linear independence constraint qualification (LICQ) condition). Then, indicating with $\tilde{\lambda}$ the Lagrange multipliers corresponding to the active constraints, and with $\bar{\lambda}$ the Lagrange multipliers corresponding to the inactive constraints, it follows

$$\begin{aligned} \bar{\lambda} &= 0 && (2.62)\\ \tilde{\lambda} &= -\left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1}\left(\tilde{w} + \tilde{S}x_0\right) && (2.63)\end{aligned}$$

Substituting (2.63) into (2.59) we obtain **z*** as an affine function of x_0

$$\mathbf{z}^* = H^{-1}\tilde{G}^T\left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1}\left(\tilde{w} + \tilde{S}x_0\right). \qquad (2.64)$$

[1]Constraints on the current output, y_0, are usually not considered in the MPC optimization problem since the control input cannot influence it anymore. However, in some cases, particularly in explicit MPC, constraints on y_0 may be considered as well, meaning that there are constraints on the initial conditions.

The relation (2.64) remains valid for all the $x \in CR_0$, $CR_0 \subset \mathcal{X}_F$, such that the set of active constraints remains unchanged. The set CR_0, called critical region, can be characterized considering that, if a new state x is considered, $\mathbf{z}^*$ from (2.64) must satisfy the constraints in (2.53) to remain valid, that is

$$GH^{-1}\tilde{G}^T \left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \left(\tilde{w} + \tilde{S}x\right) \leq w + Sx \tag{2.65}$$

and by (2.57), the Lagrange multipliers in (2.63) must remain non-negative

$$-\left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \left(\tilde{w} + \tilde{S}x\right) \geq 0. \tag{2.66}$$

From (2.65) and (2.66), we obtain a compact representation of CR_0 as

$$CR_0 = \{x \in \mathbb{R}^n | (2.65) - (2.66) \text{ are satisfied}\}. \tag{2.67}$$

The set CR_0 is a polytopic set.
In the literature (Bemporad et al. (2002b), Tøndel et al. (2003a)) procedures are described for exploring $\mathcal{X}_F$ and partitioning it into subregions like CR_0, such that $\mathcal{X}_F = \bigcup_{j=1\ldots n_R} CR_j$, where n_R is the number of critical regions generated.
In the same literature also the degenerate case, in which the assumption of linearly independent active constraints is violated, is also discussed.

The following theorem formalizes the explicit MPC-QP optimal solution.

Theorem 2. *The optimal solution $\mathbf{z}^*$ is a continuous and piecewise affine (PWA) function over the feasible set*

$$\mathbf{z}^*(x) = L_j x + g_j,\ \forall x \in CR_j,\ j = 1, ..., n_R \tag{2.68}$$

and $V_z^*(x_k)$ is continuous, convex and piecewise quadratic (PWQ). Each critical region CR_j has an associated set of (linearly independent) active constraints which forms the corresponding sub-matrices $\tilde{G}_j$, $\tilde{S}_j$ and $\tilde{w}_j$ out of the matrices G, S and w respectively, such that

$$L_j = H^{-1}\tilde{G}_j^T \left(\tilde{G}_j H^{-1}\tilde{G}_j^T\right)^{-1} \tilde{S}_j \tag{2.69}$$

$$g_j = H^{-1}\tilde{G}_j^T \left(\tilde{G}_j H^{-1}\tilde{G}_j^T\right)^{-1} \tilde{w}_j \tag{2.70}$$

The critical regions CR_j are polytopes with mutually disjoint interiors and $\mathcal{X}_F = \bigcup_{j=1\ldots n_R} CR_j$.

Proof. The proof can be found in Bemporad et al. (2002b). □

2.4.4 Extensions

The control problem considered so far concerns regulation, that is the system is controlled to the origin. As pointed out in Bemporad et al. (2002b), several extensions to this basic regulation problem can be obtained, which can still be formulated as quadratic programs and as a multi-parametric quadratic programs.
The problem of tracking a reference trajectory in the presence of certain constraints can be tackled by extending the parameter vector to contain a reference trajectory as well.
The feedback nature of the controller must of course handle the unmeasured disturbances. However, measured disturbances can easily be taken into account including those in the prediction model, leading to additional parameters in the parameter vector.
Softening constraints via slack variables is a common way to deal with the feasibility problems which can arise in MPC. The slack variables can be added to the optimization variables giving an explicit solution which is still piecewise affine.
The constraints on the control inputs and states may vary depending on the operating conditions in the system. This can be taken into account by making the constraints dependent on some external parameters added to the parameter vector.
These issues have received considerable attention from the research community, and discussions about them can be found, for example, in Rawlings and Mayne (2009), Mayne et al. (2000) and Maciejowski (2002) and references therein.

Chapter 3

Approximate Explicit MPC via Delaunay Tessellations

Explicit Model Predictive Control formulations aim to extend the scope of applicability of MPC to situations which cannot be covered effectively with existing standard MPC schemes. A relevant problem with explicit MPC is that, for large dimensional problems, coding and implementing the exact explicit solution may be excessively demanding for the hardware available. In these cases, approximation is the practical way to make effective implementations. In this chapter a technique is proposed to compute an approximate PWA control law that is suboptimal only over the subregion of the feasible set where constraints are active. In this subregion, the problem of computing a suitable suboptimal PWA control law is related to an important problem in computer aided geometric design, surface reconstruction. In fact the technique is based on a fundamental structure in computational geometry theory, Delaunay tessellation, which has been particularly successful in dealing with the surface reconstruction problem.

3.1 Introduction

Over the last few decades, Model Predictive Control (MPC) has established itself as the leading industrial control technology for systems with constraints (Rawlings and Mayne (2009), Mayne et al. (2000), Qin and Badgwell (2003), Maciejowski (2002)).
Conventional MPC methodology uses a linear model of the plant in order to determine the optimal control action with respect to a cost function (typically quadratic),

satisfying certain linear constraints. The standard implementation requires the solution of an open-loop optimal control problem over a finite horizon at each sampling time, with the current state as the initial condition of the optimization. The first element in the optimal control sequence is applied to the system. At the next time step, the computation is repeated starting from the new state and over the shifted horizon. This means that the implementation of the MPC strategy requires a quadratic programming (QP) solver for the online optimization (assuming a quadratic cost function). Although efficient QP solvers have been developed, computing the optimal control sequence at each and every time step may still require prohibitive online computational efforts, and, moreover, reduces the reliability and verifiability of the control algorithm.

Explicit MPC formulations (Bemporad et al. (2002b), Tøndel et al. (2003a), Grancharova and Johansen (2005)), based on multi-parametric programming (Bemporad and Filippi (2006), Kvasnica et al. (2006)), allow moving the optimization effort offline and obtain the optimal control as an explicitly defined piecewise affine (PWA) function with dependence on the current state vector. The domain of the PWA function is the feasible set, which is partitioned into convex regions. Thus, the online computation reduces to the simple evaluation of the piecewise affine function (point location problem). Therefore, explicit MPC represents a promising approach to extend the scope of applicability of MPC to situations where the computations required for the online optimization are restrictive, and/or where insight into the control behavior is necessary for performance analysis (like safety verifications). These are common situations, for example, in the automotive and aerospace industries (Johansen et al. (2005), Pistikopoulos (2009)).

However, explicit MPC implementations may still be prohibitively costly for large optimization problems, in fact the offline computation effort needed to solve the multi-parametric optimization problem increases fast with the dimensions of state and input, the number of constraints involved in the optimization problem and the length of the prediction horizon. As the optimization complexity increases, also the number of linear gains associated with the PWA control law increase enormously, which may cause serious difficulties on low-cost hardware.

Several approaches have been proposed to address these problems and the following just mentions a few. The approach used in Geyer et al. (2004) is to post-process the feasible set partition with the goal of merging regions characterized by the same feedback law and thereby reduce the complexity of the PWA function. In approaches like Tøndel et al. (2003b) and Christophersen (2007) the explicit PWA solution is post-processed to generate search trees that allow efficient online evaluation. In Spjøtvold et al. (2006), reachability analysis is utilized to build a structure that improves the average time for the online evaluation of PWA controllers. In Bemporad and Filippi (2003), small slacks in the optimality conditions and modifi-

cations on the explicit MPC algorithm lead to the solution of a relaxed problem that gives a simpler approximate suboptimal solution. In Rossiter and Grieder (2005) the authors simply remove several regions from the feasible set partition, and use two interpolations to obtain an online approximated control action for the missing regions. In Johansen and Grancharova (2003) the feasible set is partitioned into orthogonal hypercubes, approximate explicit control laws are obtained for each hypercube, and organized in an orthogonal search tree to allow fast real-time evaluation[1]. In Jones and Morari (2008) the authors propose a suboptimal solution based on barycentric interpolation when a linear cost function is considered. An approximate controller can be obtained using the algorithm in Bemporad and Filippi (2006), where approximate solutions of multi-parametric optimization problems are expressed as PWA functions over simplicial partitions of subsets of the feasible sets, and organized to ensure efficient evaluation. Simplices are also used in Grieder et al. (2004) to reduce the online computational load for standard MPC implementations. In the approximate explicit MPC approach presented in this chapter (Scibilia et al. (2009b), Hovd et al. (2009), Scibilia et al. (2010a)) part of the feasible set is partitioned using a particular simplicial tessellation known as Delaunay tessellation. The feasible set is considered composed by two regions: the *unconstrained* region, where no constraints are active and, thus, the optimal MPC coincides simply with the linear quadratic regulator (LQR); the *constrained* region, where constraints are active and the prohibitive optimal explicit MPC solution is replaced by a suitable approximation computed from a finite number of samples of the exact solution. The constrained region is processed and partitioned into simplices with a procedure based on Delaunay tessellation. Inside each simplex, the approximate controller is an affine state feedback law whose gains are given by linear interpolation of the exact solution at the vertices.
Delaunay tessellations (which in the plane corresponds to triangulations) are fundamental structures in computational geometry theory and have many applications in science and engineering (Hjelle and Dæhlen (2006), de Berg et al. (2008), Mount (2002), Bern and Plassmann (2000), Gallier (2000)). Delaunay tessellations characterize natural neighbor relations among sets of points distributed in the Euclidean space, and thus they have been particularly successful in the surface reconstruction problem (Cazals and Giesen (2006), Dyer et al. (2009), Bern and Plassmann (2000)), where a model of an unknown surface has to be computed given only a finite set of samples.
The basic idea behind our approach is then clear: the optimal explicit PWA controller over the state space is considered as the (partially) unknown surface to be reconstructed. The initial set of samples are the vertices of the unconstrained re-

[1]It is worth mentioning that a similar approach was used in Johansen (2004) to obtain effective approximate explicit MPC solutions when nonlinear models are considered in the MPC formulation.

gion. New samples are accurately added, and then a finer Delaunay tessellation obtained, if necessary to achieve desired performance. The entire feasible set is covered, and the whole semi-approximate solution obtained is PWA continuous.
The use of such data structures is motivated by several advantages. Delaunay tessellations are easy to define and construct (Su and Drysdale (1995), Cignoni et al. (1998), Boissonnat et al. (2009)). An additional important motivation for using Delaunay tessellation is that they allow fast online implementation of the approximate PWA control law without any additional post-processing and need of additional memory to store support structures (like search trees). The point location problem occurs quite frequently in surface reconstruction and mesh generation applications, and thus numerous high performance algorithms have been proposed in the computational geometry literature (Devroye et al. (2004), Devroye et al. (1999)) which can be straightforwardly applied in our framework. In particular, a successful simple algorithm for Delaunay tessellations which needs no pre-processing time and no additional storage is based on the Jump&Walk technique (Mücke et al. (1999)).
Delaunay tessellations are also used in the approach presented in Bemporad and Filippi (2006) which, however, differs from the approach presented here for two main characteristics: in Bemporad and Filippi (2006) the authors propose to tessellate the entire feasible set, not distinguishing between unconstrained and constrained regions; the final tessellation derived according to the approach in Bemporad and Filippi (2006) is not, in general, a Delaunay tessellation, which restricts the applicability of point location algorithms based on Delaunay properties like, for example, Jump&Walk algorithms.

3.2 Problem Formulation

In this section, the MPC problem introduced in Chapter 2 is recalled.
Consider the problem of regulating the following discrete-time linear time-invariant system to the origin

$$x\,(t+1) = Ax\,(t) + Bu\,(t) \tag{3.1}$$

$$y(t) = Cx(t) \tag{3.2}$$

while satisfying the output and input constraints

$$y(t) \in \mathcal{Y} \tag{3.3}$$

$$u(t) \in \mathcal{U} \tag{3.4}$$

for all time instants $t \geq 0$, where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^r$ is the input vector and $y \in \mathbb{R}^m$ is the output vector, $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times r}$, $\mathcal{Y} \subset \mathbb{R}^m$ and

$\mathcal{U} \subset \mathbb{R}^r$ are polyhedral sets given by linear inequalities

$$\mathcal{Y} = \left\{ y \in \mathbb{R}^m | \; D_{\mathcal{Y}} y \leq d_{\mathcal{Y}} \right\} \tag{3.5}$$

$$\mathcal{U} = \left\{ u \in \mathbb{R}^r | \; D_{\mathcal{U}} u \leq d_{\mathcal{U}} \right\} \tag{3.6}$$

where $D_{\mathcal{Y}} \in \mathbb{R}^{n_{\mathcal{Y}} \times m}$, $D_{\mathcal{U}} \in \mathbb{R}^{n_{\mathcal{U}} \times r}$, $d_{\mathcal{Y}} \in \mathbb{R}^{n_{\mathcal{Y}}}$, $d_{\mathcal{U}} \in \mathbb{R}^{n_{\mathcal{U}}}$, and $n_{\mathcal{Y}}$ and $n_{\mathcal{U}}$ are the number of inequalities on the output and on the input respectively.
It is assumed that the pair (A, B) is stabilizable.
Provided that the state $x\,(t)$ is available from the measurements, the finite horizon MPC optimization problem is

$$\min_{\mathbf{u} \triangleq [\mathbf{u_0}, \ldots, \mathbf{u_{N-1}}]} \left\{ J\left(\mathbf{u}, x\,(t)\right) = x_N^T P x_N + \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k \right\} \tag{3.7}$$

$$\begin{array}{lll} \text{s.t.} & x_0 = x\,(t)\,, & \\ & x_{k+1} = A x_k + B u_k & \\ & y_k = C x_k & k = 0, 1, \ldots, N \\ & y_k \in \mathcal{Y}, & k = 1, 2, \ldots, N \\ & u_k \in \mathcal{U}, & k = 0, 1, \ldots, N-1 \\ & x_N \in \Omega & \end{array} \tag{3.8}$$

where x_k denotes the predicted state vector at time $t + k$ obtained by applying the k first elements of the input sequence $u_0, \ldots, u_{N-1}$; N is the prediction horizon; $Q \succeq 0$ (positive semidefinite) and $R \succ 0$ (positive definite) are symmetric matrices corresponding to weights on state and input. It is assumed that $0 \in \mathtt{int}(\mathcal{Y})$, $0 \in \mathtt{int}(\mathcal{U})$ such that the origin is an interior point in the feasible set:

$$\mathcal{X}_F = \left\{ x \in \mathbb{R}^n | \; \exists\, \mathbf{u} \text{ satisfying}(3.8) \right\} \tag{3.9}$$

The terminal cost matrix P and the terminal constraint $x_N \in \Omega$ are defined to guarantee stability of the closed-loop.
Note that $\mathcal{X}_F$ is a polytope due to the nature of the constraints (Scibilia et al. (2010b)).
The MPC optimization problem (3.7-3.8) can be formulated as the following QP

$$\begin{array}{rl} J^*(x) = & \min_{\mathbf{u}} \mathbf{u}^T H \mathbf{u} + 2 x^T F \mathbf{u} + x^T Y x \\ & \text{s.t. } G\mathbf{u} \leq w + Ex \end{array} \tag{3.10}$$

where the matrices H, F, Y, G, w and E are as defined in Chapter 2. Note that the term $x^T Y x$ can be removed from the optimization since it does not influence the optimal argument.

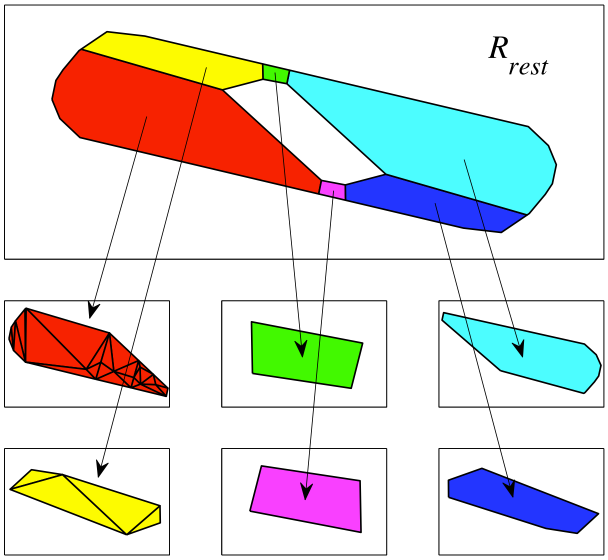


Figure 3.1: An example of partition of a non-convex region R_{rest} in convex subregions. For some of the subregions the Delaunay triangulation of the points needed to achieve a given tolerance is also shown. The empty region within R_{rest} belongs to the terminal set Ω.

3.3 Approximate Explicit MPC

The feasible set $\mathcal{X}_F$ is considered to be composed of two regions:

- Ω, the region where no constraints are active and thus the MPC reduces to the unconstrained LQR;
- $R_{rest} = \mathcal{X}_F \backslash \Omega$, the region containing the rest of the feasible set.

The first region of our partition is Ω, and the associated controller is the LQR. The set R_{rest} is processed and partitioned into simplices with a procedure based on Delaunay tessellations. Note that the operation is not trivial since R_{rest} is non-convex (Figure 3.1). First, R_{rest} is appropriately divided into convex subregions. For each subregion, the Delaunay tessellation of the vertices is computed. Inside each simplex, the approximate controller is the affine state feedback law whose gains are obtained simply by linear interpolation of the exact optimal controller at the vertices of the simplex. If the desired accuracy is not achieved, more points are opportunely added and a finer Delaunay tessellation of the subregion is derived.
In the following sections the details of the algorithm are given.

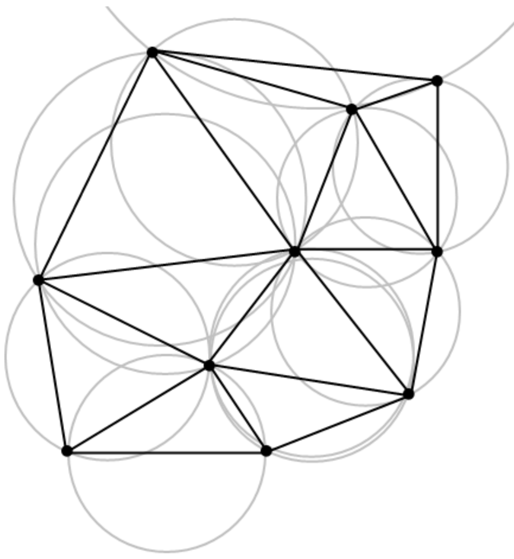


Figure 3.2: Delaunay triangulation of a set of points.

3.4 Delaunay Tessellation

The importance of Delaunay tessellations (DT) is in their ability to characterize natural neighbor relations among points distributed in Euclidean space.
Considering the plane, the Delaunay triangulation of a general set of (non-collinear) points is formally defined as follows by the *empty circle condition*.

Definition 17. (Delaunay triangulation) The Delaunay triangulation of a set P of points in $\mathbb{R}^2$ is a triangulation (DT(P)) such that no point in P is inside the circumcircle of any triangle in DT(P) (Figure 3.2).

The concept can be generalized to the general d-dimensional space as follows.

Definition 18. (Delaunay tessellation) The Delaunay tessellation of a set P of points in $\mathbb{R}^d$ (DT(P)) is a simplicial tessellation such that no vertex in P is inside the circum-hypersphere of any simplex in DT(P).

Simplicial tessellations/triangulations are a common approach to discretizing surfaces or, in general, objects. Taken a suitable set of samples P, these samples are connected by linear elements (line segments, triangles, tetrahedra, etc.) to form a covering of the object that provides a convenient framework for interpolation and numerical computations (Hjelle and Dæhlen (2006), de Berg et al. (2008), Mount

(2002)).
Delaunay tessellations are structures characterized by several important properties which constitute the reason for their many applications in science and engineering. In the plane, DT minimizes the maximum circumradius of the triangles. In the general d-dimensional space, DT minimizes the radius of the maximum smallest enclosing hypersphere. Such properties generally mean triangulations/tessellations composed by “well-shaped" simplices, where the quality of a simplex is here measured as the ratio of the shortest edge to the circumradius: the larger this number is, the higher the simplex quality is. Another relevant property is that the union of all the simplices in the DT tessellation is the convex hull of the points. These are all desired properties particularly in the context of surface approximation or mesh generation. DTs also enjoy other features which make these structures particularly suitable for surface approximation: topological consistency, pointwise approximation and normal approximation. These features mean that, given a set P of samples of a surface S, the approximation DT(P) and S are homeomorphic[2], and that the approximation improves with increased sampling density (Cazals and Giesen (2006), Dyer et al. (2009), Bern and Plassmann (2000)).
The problem of computing the DT of a set of points in the general d-dimensional space is strictly related to another fundamental problem in computational geometry: the computation of the convex hull of a set of points. Consider the set of points P in d-dimension, then DT(P) can be computed by giving each point $p \in P$ an extra coordinate equal to $||p||^2$ (the points are lifted onto a paraboloid in $d+1$-dimensional space), and then taking the projection of the downward-facing facet of the convex hull of these higher dimensional points (Gallier (2000)). The fastest algorithms for computing the DT of a set of points use the *divide&conquer* approach. In the plane, the Delaunay triangulation of n_p points can be constructed in $O\left(n_p \log n_p\right)$ runtime; in the general d-dimensional space, the worst case complexity for constructing the DT of n_p points is $O\left(n_p \log n_p + n_p^{\lceil d/2 \rceil}\right)$. The simplest algorithms are based on the *incremental* approach, where points are added one by one, iteratively updating the existing DT (Cignoni et al. (1998), Su and Drysdale (1995), Boissonnat et al. (2009)).

3.5 Approximate Controller

The controller inside each simplex is given by the following theorem (Bemporad and Filippi (2006)).

[2]Roughly speaking, the homeomorphism between two geometric objects is a continuous stretching and bending of one object into the shape of the other.

Theorem 3. *Given a simplex $\mathcal{S} \subseteq \mathcal{X}_F$ whose vertices are $\left\{v^{(1)}, ..., v^{(n+1)}\right\}$ and the optimal input sequences at the vertices $\mathbf{u}^*\left(v^{(i)}\right)$, $i = 1, ..., n+1$, then the affine function*

$$\tilde{\mathbf{u}}_{\mathcal{S}}(x) = \tilde{L}_{\mathcal{S}}^T x + \tilde{g}_{\mathcal{S}} \tag{3.11}$$

where $\tilde{L}_{\mathcal{S}} \in \mathbb{R}^{n \times rN}$ and $\tilde{g}_{\mathcal{S}} \in \mathbb{R}^{rN}$ are obtained by linear interpolation

$$\begin{bmatrix} {v^{(1)}}^T & 1 \\ \vdots & \vdots \\ {v^{(n+1)}}^T & 1 \end{bmatrix} \begin{bmatrix} \tilde{L}_{\mathcal{S}} \\ \\ \tilde{g}_{\mathcal{S}}^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}^*\left(v^{(1)}\right)^T \\ \vdots \\ \boldsymbol{u}^*\left(v^{(n+1)}\right)^T \end{bmatrix}, \tag{3.12}$$

gives a feasible input sequence for all $x \in \mathcal{S}$.

Proof. It is simple to prove that $(G\tilde{L}_{\mathcal{S}}^T - E)x \leq w - G\tilde{g}_{\mathcal{S}}\ \forall\, x \in \mathcal{S}$. By construction, it is $(G\tilde{L}_{\mathcal{S}}^T - E)v^{(i)} \leq w - G\tilde{g}_{\mathcal{S}}, i = 1, .., n+1$. Thus, since every $x \in \mathcal{S}$ can be expressed as linear convex combination $x = \sum_{k=1}^{n+1} \lambda_k v^{(k)}$ where $\sum_{k=1}^{n+1} \lambda_k = 1$ and $\lambda_k \geq 0\ \forall k$, feasibility follows by joint convexity of the constraints and $\mathcal{S}$. □

3.6 Approximation Error

Consider a simplex $\mathcal{S} = \{x \in \mathcal{X}_F |\ D_{\mathcal{S}} x \leq d_{\mathcal{S}}\}$ and the approximate controller (3.11) defined therein. The approximate cost function is given by

$$\tilde{J}_{\mathcal{S}}(x) = x^T \tilde{H}_{\mathcal{S}} x + \tilde{F}_{\mathcal{S}}^T x + \tilde{Y}_{\mathcal{S}}\ \forall x \in \mathcal{S} \tag{3.13}$$

where $\tilde{H}_{\mathcal{S}} = Y + \tilde{L}_{\mathcal{S}} H \tilde{L}_{\mathcal{S}}^T + 2F\tilde{L}_{\mathcal{S}}^T$, $\tilde{F}_{\mathcal{S}} = 2\tilde{L}_{\mathcal{S}} H \tilde{g}_{\mathcal{S}} + 2F\tilde{g}_{\mathcal{S}}$ and $\tilde{Y}_{\mathcal{S}} = \tilde{g}_{\mathcal{S}}^T H \tilde{g}_{\mathcal{S}}$.
The quadratic function $\tilde{J}_{\mathcal{S}}(x)$ is our approximation inside $\mathcal{S}$ of the optimal cost function $J^*(x)$ (Figure 3.4). Note that $\tilde{J}(x)$ and $J^*(x)$ are both convex and $\tilde{J}(x) \geq J^*(x)\ \forall x \in \mathcal{S}$ (Bemporad and Filippi (2006)).
As a measure of the level of accuracy achieved, the maximum absolute error introduced by the approximation is considered

$$\epsilon_{max}(\mathcal{S}) \triangleq \max_{x \in \mathcal{S}} \left\{\tilde{J}_{\mathcal{S}}(x) - J^*(x)\right\}. \tag{3.14}$$

The above optimization problem maximizes the difference of two convex functions, which in general is a hard problem to solve (Bemporad and Filippi (2006)). In this case it results in an indefinite QP programming problem, since the corresponding Hessian matrix can be indefinite. However, for problems of modest dimension this may nevertheless be tractable. Moreover, for indefinite QPs defined over simplices,

in addition to the piecewise linear lower bounds often used in Branch&Bound based optimization for indefinite QPs, there are also more accurate convex lower bounds based on semidefinite programming (Hovd et al. (2009)).
Here we consider easily computable estimates of the maximum error introduced, which are based on similar ideas as in Bemporad and Filippi (2006). These estimates are upper limits of the exact maximum error (3.14).

3.6.1 Approximation error estimated via a lower bound on the optimal cost function

Let $V_{\mathcal{S}} = \left\{v^{(1)}, ..., v^{(n+1)}\right\}$ be the vertices of the simplex $\mathcal{S}$.
Assume that $J^*(v)$ is differentiable for all $v \in V_{\mathcal{S}}$. Let $(\mathbf{u}_i^*, \lambda_i^*)$, $i = 1, ..., n+1$ be the solution of the KKT conditions at the vertices. Then, using the optimal Lagrange multipliers, the optimal input sequence at each vertex can be expressed as affine function of the vertex where it is computed (cf. Chapter 2)

$$\mathbf{u}_i^* = L_i v^{(i)} + g_i \;\; i = 1, ..., n+1 \tag{3.15}$$

Using (3.15) in the cost function of (3.10) and differentiating with respect to x, a subgradient of $J^*(x)$ in each vertex is given by

$$\nabla J_i^* = 2F_{\nabla i} v^{(i)} + Y_{\nabla i} \tag{3.16}$$

where $F_{\nabla i} = L_i^T H L_i + 2FL_i + Y$, $Y_{\nabla i} = 2L_i^T H g_i + 2F g_i$.
From convexity of $J^*(x)$ it follows that

$$J^*(x) \geq J^*(v^{(i)}) + \nabla J_i^{*T}(x - v^{(i)}) \tag{3.17}$$

for all $i \in \{1, 2, ..., n+1\}$. Then, the piecewise linear function

$$\underline{J}_{\mathcal{S}}(x) = \max_{i=1,...,n+1} \left(J^*(v^{(i)}) + \nabla J_i^{*T}(x - v^{(i)})\right) \tag{3.18}$$

defined for all $x \in \mathcal{S}$ is a lower bound of $J^*(x)$ inside $\mathcal{S}$ (Figures 3.3, 3.5).
By construction it follows that

$$\tilde{J}_{\mathcal{S}}(x) \geq J^*(x) \geq \underline{J}_{\mathcal{S}}(x) \; \forall x \in \mathcal{S} \tag{3.19}$$

where the three functions coincide at the vertices of $\mathcal{S}$.
Therefore, it follows that

$$\underline{\epsilon}_{max}(\mathcal{S}) \triangleq \max_{x \in \mathcal{S}} \left\{\tilde{J}_{\mathcal{S}}(x) - \underline{J}_{\mathcal{S}}(x)\right\} \tag{3.20}$$

is an upper bound of the maximum error introduced within $\mathcal{S}$ (Johansen and Grancharova (2003), Bemporad and Filippi (2006)).
The point $x_{\underline{\epsilon}} = \arg\max_{x\in\mathcal{S}} \left(\tilde{J}_{\mathcal{S}}(x) - \underline{J}_{\mathcal{S}}(x)\right)$ is the state corresponding to $\underline{\epsilon}_{max}(\mathcal{S})$.

Note that the problem of finding $x_{\underline{\epsilon}}$ and $\underline{\epsilon}_{max}(\mathcal{S})$ within the simplex $\mathcal{S}$ is made quite simple by the nature of the lower bounding function $\underline{J}_{\mathcal{S}}$ and by the function $\tilde{J}_{\mathcal{S}}$ being convex quadratic within $\mathcal{S}$.
The piecewise linear function $\underline{J}_{\mathcal{S}}$ can be defined by geometric methods. For each vertex $v^{(i)} \in V_{\mathcal{S}}$, the associated subgradient characterizes a half-plane in $n+1$-dimension

$$h_i = J^*(v^{(i)}) + \nabla J_i^{*T}(x - v^{(i)}) \tag{3.21}$$

Projecting the intersections of these half-planes on the state-space produces the partition of the simplex which defines $\underline{J}_{\mathcal{S}}$ (Figure 3.5): each piece of the partition has associated the corresponding linear function (3.21).

Remark 2. In the definitions of the subgradients (3.16) it has been implicitly assumed that the function $J^*(x)$ is differentiable at the vertices. This assumption, however, may be violated at some vertex, since the function J^* is (continuous) piecewise quadratic over the feasible set, and therefore points exist where it is not differentiable. This corresponds to the degenerate case described in Bemporad et al. (2002b). The optimal solution of the QP (3.10) may result in a linearly dependent set of active constraints for some of the vertices of the simplex considered. This is the case, for instance, when a vertex is located on a boundary between two or more critical regions. Therefore, for the degenerate vertex more than one subgradient of J^* will exist, particularly one for each critical region containing the vertex. Given the definition (3.18) of $\underline{J}_{\mathcal{S}}$, considering all the subgradients associated to a vertex will deal with the problem. However, this makes more involved defining geometrically $\underline{J}_{\mathcal{S}}$. For each vertex, several half-planes (3.21) can be determined (some of those may not be relevant within the simplex under consideration) and the mutual intersections also need to be considered.

3.6.2 Approximation error estimated via an upper bound on the suboptimal cost function

Let $V_{\mathcal{S}} = \left\{v^{(1)}, ..., v^{(n+1)}\right\}$ be the vertices of $\mathcal{S}$. Define

$$\bar{J}_{\mathcal{S}}(x(t)) = \bar{F}_{\mathcal{S}}^T x(t) + \bar{Y}_{\mathcal{S}} \tag{3.22}$$

where $\bar{F}_{\mathcal{S}}$ and $\bar{Y}_{\mathcal{S}}$ are obtained by

$$\begin{bmatrix} v^{(1)^T} & 1 \\ \vdots & \vdots \\ v^{(n+1)^T} & 1 \end{bmatrix} \begin{bmatrix} \bar{F}_{\mathcal{S}} \\ \\ \bar{Y}_{\mathcal{S}} \end{bmatrix} = \begin{bmatrix} J^*\left(v^{(1)}\right) \\ \vdots \\ J^*\left(v^{(n+1)}\right) \end{bmatrix}. \tag{3.23}$$

From convexity of $\tilde{J}_{\mathcal{S}}(x)$ and $J^*(x)$ it is immediate to show that (Bemporad and Filippi (2006)):

$$\bar{J}_{\mathcal{S}}(x) \geq \tilde{J}_{\mathcal{S}}(x) \geq J^*(x)\ \forall x \in \mathcal{S} \tag{3.24}$$

(Figure 3.4).
Then it follows that

$$\bar{\epsilon}_{max}(\mathcal{S}) \triangleq \max_{x \in \mathcal{S}} \left\{ \bar{J}_{\mathcal{S}}(x) - J^*(x) \right\} \tag{3.25}$$

is an upper bound of the maximum error introduced within $\mathcal{S}$. Considering the initial state x as an optimization variable, we can define

$$\theta \triangleq \begin{bmatrix} x \\ \mathbf{u} \end{bmatrix}$$

and rewrite the cost function of (3.10) as

$$J(\theta) = \theta^T H_b \theta \tag{3.26}$$

where $H_b = \left[\hat{A}\ \hat{B}\right]^T \hat{Q} \left[\hat{A}\ \hat{B}\right] + \mathtt{diag}\left(\mathbb{0}_{n\times n}, \hat{R}\right)$, and $\hat{A}$, $\hat{B}$, $\hat{Q}$, $\hat{R}$ as defined in Chapter 2. The constraints can also be rewritten accordingly as

$$G_b \theta \leq w \tag{3.27}$$

where $G_b = [-E \ \ G]$.

Theorem 4. *The upper bound $\bar{\epsilon}_{max}(\mathcal{S})$ on the maximum absolute error inside $\mathcal{S}$ is given by the optimal value of the QP*

$$\begin{aligned} \bar{\epsilon}_{max}(\mathcal{S}) = \ & -\min_\theta\ \theta^T H_{\bar{\epsilon}} \theta + F_{\bar{\epsilon}}^T \theta + Y_{\bar{\epsilon}} \\ & \text{s.t. } G_b \theta \leq w,\ \left[D_{\mathcal{S}}\ \mathbb{0}_{(n+1)\times(rN)}\right] \theta \leq d_{\mathcal{S}} \end{aligned} \tag{3.28}$$

where

$$H_{\bar{\epsilon}} = H_b,\ F_{\bar{\epsilon}} = \begin{bmatrix} -\bar{F}_{\mathcal{S}} \\ 0 \end{bmatrix},\ Y_{\bar{\epsilon}} = -\bar{Y}_{\mathcal{S}}.$$

Moreover, if

$$\theta^* = \begin{bmatrix} x^* \\ \mathbf{u}^* \end{bmatrix}$$

is the optimal solution of (3.28), then $\mathbf{u}^*$ is the optimal solution of (3.10) at x^*.

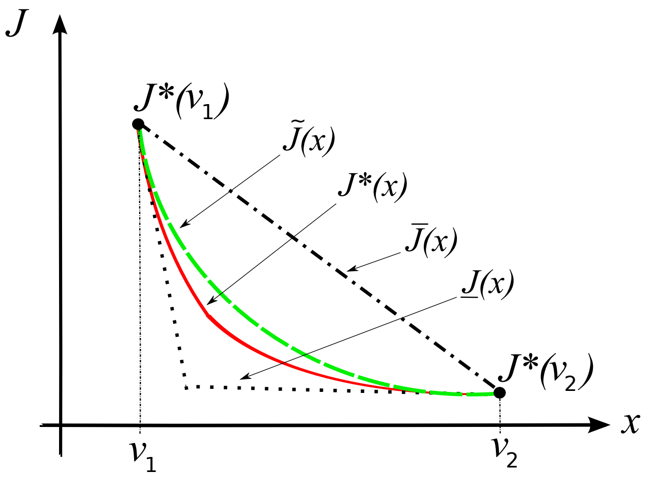


Figure 3.3: Approximation of the value cost function in convex parametric programming: the scalar case.

Proof. From (3.22) and (3.26) we can see that

$$\theta^T H_{\bar{\epsilon}}\theta + F_{\bar{\epsilon}}^T\theta + Y_{\bar{\epsilon}} = J(\theta) - \bar{J}_{\mathcal{S}}(x). \tag{3.29}$$

Then by (3.25) we have

$$\begin{aligned} \bar{\epsilon}_{max}(\mathcal{S}) &= \max_{x\in\mathcal{S}}\left\{\bar{J}_{\mathcal{S}}(x) - J^*(x)\right\} \\ &= -\min_{x\in\mathcal{S}}\left\{-\bar{J}_{\mathcal{S}}(x) + J^*(x)\right\} \\ &= -\min_{x\in\mathcal{S}}\left\{-\bar{J}_{\mathcal{S}}(x) + \min_{\mathbf{u}}\left\{J(x,\mathbf{u}) : G\mathbf{u} \le w + Ex\right\}\right\} \\ &= -\min_{x,\mathbf{u}}\left\{-\bar{J}_{\mathcal{S}}(x) + J(x,\mathbf{u}) : G\mathbf{u} \le w + Ex, x\in\mathcal{S}\right\} \end{aligned} \tag{3.30}$$

and (3.28) follows noting that

$$\begin{bmatrix} x \\ \mathbf{u} \end{bmatrix} = \theta.$$

The last statement of the theorem can be simply proven by contradiction. □

Note that $H_{\bar{\epsilon}}$ is a positive semidefinite matrix because of the assumptions on Q and R.

3.7 The Algorithm

Before presenting the algorithm, the concepts of *virtual constraints* and *virtual vertices* need to be introduced, which are instrumental to overcome the problem of

partitioning a non-convex region into convex subregions (Figure 3.1).
Let $\mathcal{P}_{in}$ and $\mathcal{P}_{ext}$ be two polytopes such that $\mathcal{P}_{in} \subset \mathcal{P}_{ext}$. Consider the half-space representation of $\mathcal{P}_{in}$

$$\mathcal{P}_{in} = \{x \in \mathbb{R}^n | Dx \leq d\}. \tag{3.31}$$

The facet i is represented as

$$f_i = \mathcal{P}_{in} \cap \left\{x \in \mathbb{R}^n | D_{(i,\cdot)}x = d_{(i)}\right\} \tag{3.32}$$

where $D_{(i,\cdot)}$ indicates the ith row of D and $d_{(i)}$ the ith element of d. Given two facets of a polyhedral set, they are said to be *contiguous* if they have a ridge in common. Denote with $F_c(i)$ the index set of all the facets of P_{in} contiguous to f_i.

Definition 19 (Virtual constraint)**.** For all $j \in F_c(i)$ we define the half-spaces

$$D_{ij}x \geq d_{ij} \tag{3.33}$$

where

$$D_{ij} = \begin{bmatrix} \frac{D_{(i,1)}}{\rho_i} - \frac{D_{(j,1)}}{\rho_j} \\ \vdots \\ \frac{D_{(i,n)}}{\rho_i} - \frac{D_{(j,n)}}{\rho_j} \end{bmatrix}, \; d_{ij} = \frac{d_{(i)}}{\rho_i} - \frac{d_{(j)}}{\rho_j}$$

and $\rho_i = \sqrt{\sum_{k=1}^n D_{(i,k)}^2}$, $\rho_j = \sqrt{\sum_{k=1}^n D_{(j,k)}^2}$.

We call (3.33) the *virtual constraints* relative to f_i and its contiguous f_j.

Definition 20 (Visible vertices from a facet)**.** Consider the polytope

$$\mathcal{P}_{sect}(i) = \left\{x \in \mathcal{P}_{ext} | D_{(i,\cdot)}x \geq d_{(i)}, D_{ij}x \geq d_{ij} \; \forall j \in F_c(i)\right\}. \tag{3.34}$$

The vertices of $\mathcal{P}_{ext}$ *visible* from f_i are all the vertices of $\mathcal{P}_{sect}(i)$ except the common vertices with $\mathcal{P}_{in}$.

The definition just given of "vertices of $\mathcal{P}_{ext}$ visible from f_i" also includes vertices that are not real vertices of $\mathcal{P}_{ext}$. These vertices are called *virtual vertices*.
A geometric interpretation can be given in the 2-dimensional state-spaces, where a facet is a line and the virtual constraint relative to two facets coincides with the bisector of the angle between the two lines (Figure 3.1).

Remark 3. Note that this approach of partitioning the non-convex region R_{rest} into convex subregions maintains a natural neighbor relationship of the set of points distributed among the convex subsets.

The procedure for computing the approximate explicit MPC law can be summarized in Algorithm 2.
We indicate with $\mathfrak{E}_{max}(\mathcal{S})$ a function which measures the level of accuracy achieved in each simplex $\mathcal{S}$ in terms of the maximum absolute error introduced by the corresponding suboptimal controller. Without any particular complication, we can assume that the function provides also the state where this error occurs. This function can be either exactly the maximum error (3.14) or an approximation of it by means of the upper bounds (3.20) or (3.25).
For each simplex, the suboptimal controller is readily given by (3.11).

Algorithm 2: Approximate explicit MPC

```
Input: The feasible set to be partitioned $\mathcal{P}_{ext} = \mathcal{X}_F$. The set where the
       optimal solution is the LQR $\mathcal{P}_{in} = \Omega$. The prescribed tolerance
       $\tau_\epsilon > 0$ for the cost function error.
Output: The suboptimal PWA controller over the corresponding partition
        $\mathcal{D}_P = \{S_1, ..., S_{n_\mathcal{D}}\}$ such that $\mathcal{P}_{ext} = \bigcup_{j=1...n_\mathcal{D}} S_j$.
1  Set the first partition of $\mathcal{D}_\mathcal{P}$ to be $\mathcal{P}_{in}$. The controller is the LQR;
2  compute the optimum at the vertices and at the virtual vertices of $\mathcal{P}_{ext}$;
3  foreach facet $f_i$ of $\mathcal{P}_{in}$ do
4      initialize the set $V_{tess}$ with the vertices of the facet $f_i$ and all the vertices
       of $\mathcal{P}_{ext}$ visible from $f_i$;
5      compute the initial DT$(V_{tess})$;
6      foreach simplex $\mathcal{S} \in \mathrm{DT}(V_{tess})$ do
7          compute the maximum error $\mathfrak{E}_{max}(\mathcal{S})$;
8      end
9      while $\max_{\mathcal{S}\in\mathrm{DT}(V_{tess})} \{\mathfrak{E}_{max}(\mathcal{S}) - \tau_\epsilon\} > 0$ do
10         add to $V_{tess}$ the state $x_\epsilon$ where $\max\{\mathfrak{E}_{max}(\mathcal{S}) - \tau_\epsilon\}$ is achieved;
11         update DT$(V_{tess})$;
12         compute $\mathfrak{E}_{max}(\mathcal{S})$ for the new simplices;
13     end
14     add DT$(V_{tess})$ to $\mathcal{D}_P$;
15 end
```

With respect to the MPC optimization problem (3.7-3.8), we have the following theorem.

Theorem 5. *Algorithm 4.1 terminates in finite time providing a suboptimal PWA control law over $\mathcal{X}_F$ which is continuous and feasible.*

Proof. The number of facets of Ω is finite, thus the for-loop at step 3 will iterate for a finite number of times. Since V_{tess} is a finite set of points, DT(V_{tess}) contains a finite number of simplices, and also the for-loop at step 6 will terminate in finite time. Then, we have to prove that the while-loop at step 9 will terminate in finite time too. At each iteration of the loop the algorithm adds to the present DT(V_{tess}) the state which currently corresponds to the cost function error that most severely violates the maximum allowed tolerance. This means that at each iteration the finer DT obtained is characterized by a lower maximum cost function error, thus providing a monotonic decrease. Considering that the maximum cost function error converges towards zero (possibly for an infinite number of points) and that the tolerance is a value strictly greater than zero, eventually the estimate of the maximum cost function error will be lower than the tolerance, and the while-loop will terminate. Continuity and feasibility of the PWA solution follows by Theorem 3. □

Remark 4. Note that in Algorithm 2, for each facet of $\mathcal{P}_{in}$ the Delaunay tessellation is first computed from the corresponding vertices of $\mathcal{P}_{ext}$ and $\mathcal{P}_{in}$, and then successively refined adding one by one more points if needed to improve the approximation. With this way of computing the DT, an efficient computation strategy would be to use at the first stage a fast algorithm based on branch&bound approach, and then refine the tessellation with a simple incremental approach algorithm.

3.8 Stability

Using a similar approach as in Bemporad and Filippi (2003) and in Johansen and Grancharova (2003), it can be proven that under a certain condition on the maximum tolerance allowed, the exact cost function J^* is a Lyapunov function for the system (3.1) in closed-loop with the approximate controller given by Algorithm 2, and thus the origin is asymptotically stable (Mayne et al. (2000)).

Theorem 6. *Let $\beta > 0$ be the largest number for which the ellipsoid*

$$E = \left\{x \in \mathbb{R}^n | \; x^T Q x \leq \beta\right\} \tag{3.35}$$

is contained in the terminal constraint set Ω, where Q is as in (3.7). If the tolerance is such that $\tau_\epsilon \leq \beta$, then the approximate explicit MPC given by Algorithm 2 asymptotically stabilizes the system (3.1-3.2) while fulfilling the constraints (3.3-3.4).

Proof. Let $\tilde{\mathbf{u}} = \left[\tilde{u}_0^T, ..., \tilde{u}_{T-1}^T\right]^T$ be the approximate controller for an arbitrary $x(t) \in \mathcal{X}_F$ at time t. Applying the input sequence $\tilde{\mathbf{u}}$ to (3.1) with initial state $x(t)$, we obtain the state x_N, which is within Ω since $\tilde{\mathbf{u}}$ is feasible. By the MOAS properties of Ω, the input $u = -Kx_N$ is feasible, which makes the the input sequence $\tilde{\mathbf{u}}' = \left[\tilde{u}_1^T, ..., \tilde{u}_{T-1}^T, (-Kx_N)^T\right]^T$ feasible for the time $t+1$. Since $\tilde{J}(x)$ is an upper bound on $J^*(x)$, as in Bemporad and Filippi (2003) standard arguments from dynamic programming can be used to show that along the trajectories of the closed-loop system with the approximate controller we have

$$\begin{aligned} J^*(x(t+1)) - J^*(x(t)) \leq \;& \tilde{J}(x(t+1)) - J^*(x(t)) \\ &= \tilde{J}(x(t)) - x(t)^T Q x(t) - \tilde{u}_0^T R \tilde{u}_0 - J^*(x(t)) \end{aligned} \tag{3.36}$$

For all $x(t) \in \Omega$ we have $\tilde{J}(x(t)) = J^*(x(t))$ and therefore

$$J^*(x(t+1)) - J^*(x(t)) < 0, \; \forall \, x(t) \in \Omega \setminus \{0\} \tag{3.37}$$

Consider now the case $x(t) \in R_{rest}$. Because of the while-loop at step 9 in Algorithm 2, we have that

$$\tilde{J}(x(t)) - J^*(x(t)) \leq \tau_\epsilon. \tag{3.38}$$

Then, since $x(t)^T Q x(t) > \beta$ for all $x(t) \notin \Omega$, requiring that $\tau_\epsilon \leq \beta$ means

$$J^*(x(t+1)) - J^*(x(t)) \leq \tilde{J}(x(t)) - J^*(x(t)) - x(t)^T Q x(t) - \tilde{u}_0^T R \tilde{u}_0 < \; 0 \tag{3.39}$$

Since J^* is positive definite with $J^*(0) = 0$, and radially unbounded, it is a valid candidate Lyapunov function. From the invariance principle of LaSalle (Khalil (2000)), $x(t) \to \Omega$ as $t \to \infty$, and the origin is asymptotically stable with a region of attraction equal to $\mathcal{X}_F$. $\square$

Theorem 6 gives a condition on the allowed tolerance τ_ϵ. The upper bound β can be computed a priori from the control specifications, but may be quite conservative.
Less conservative bounds on the maximum allowed tolerance can be given if we particularize the condition inside each simplex, as explained in the following theorem.
Given a simplex $\mathcal{S}$ with approximate input sequence $\tilde{\mathbf{u}}_{\mathcal{S}}(x) = \tilde{L}_{\mathcal{S}}^T x + \tilde{g}_{\mathcal{S}} \; \forall x \in \mathcal{S}$, let us indicate the control law as $\tilde{u}_{\mathcal{S}0}(x) = \tilde{L}_{\mathcal{S}(\cdot,1)}^T x + \tilde{g}_{\mathcal{S}(1)}$, where $\tilde{L}_{\mathcal{S}(\cdot,1)}$ is the first column of $\tilde{L}_{\mathcal{S}}$ and $\tilde{g}_{\mathcal{S}(1)}$ is the first element of $\tilde{g}_{\mathcal{S}}$.

Theorem 7. *For each simplex $\mathcal{S}$ and relative approximate controller $\tilde{u}_{\mathcal{S}0}(x) \; \forall x \in \mathcal{S}$, if the tolerance is such that $\tau_\epsilon \leq \beta(\mathcal{S})$, where*

$$\beta(\mathcal{S}) = \min_{x \in \mathcal{S}} x^T H_\beta x + F_\beta^T x + Y_\beta \tag{3.40}$$

$H_\beta = Q + \tilde{L}_{\mathcal{S}(\cdot,1)} R \tilde{L}^T_{\mathcal{S}(\cdot,1)}$, $L_\beta = 2\tilde{L}_{\mathcal{S}(\cdot,1)} R \tilde{g}_{\mathcal{S}(1)}$ and $Y_\beta = \tilde{g}^T_{\mathcal{S}(1)} R \tilde{g}_{\mathcal{S}(1)}$, then the approximate explicit MPC given by Algorithm 2 asymptotically stabilizes the system (3.1-3.2) while fulfilling the constraints (3.3-3.4).

Proof. Noticing that

$$x^T H_\beta x + F_\beta^T x + Y_\beta = x(t)^T Q x(t) + \tilde{u}^T_{\mathcal{S}0} R \tilde{u}_{\mathcal{S}0}, \tag{3.41}$$

then analogously to the proof for Theorem 6 it is possible to show that the origin is asymptotically stable with a region of attraction equal to $\mathcal{X}_F$. □

Note that to check the condition given by Theorem 7, Algorithm 2 needs to be slightly modified at step 9 in the while-loop condition as following

$$\max\{\mathfrak{E}_{max}(\mathcal{S}) - \hat{\tau}_\epsilon\} > 0, \text{ where } \hat{\tau}_\epsilon = \min\{\tau_\epsilon, \beta(\mathcal{S})\} \tag{3.42}$$

Remark 5. Closed-loop stability with the approximate controller is guaranteed by ensuring that the approximate cost function is within a certain bound around the optimal cost function, such that the optimal cost function is a Lyapunov function. Most of the existing approaches for designing approximate explicit MPC bind the stability issue analogously with a maximum allowed loss of performance. Although this is nevertheless reasonable, a different approach is presented in Hovd et al. (2009), where the quality of the approximation is gradually increased by refining the Delaunay tessellation until it can be shown that the approximate cost function is itself a Lyapunov function. Thus, a stable approximate explicit MPC law can be obtained removing the need for estimating the loss relative to the optimal cost function. However, the approach results in the solution of indefinite QP problems which are in general computationally demanding. A discussion of this issue, and more details about the approach can be found in Hovd et al. (2009).

3.9 Complexity

Usually, with the complexity of explicit MPC one refers to the number of pieces forming the corresponding PWA controller. A higher number of pieces means in general higher computational complexity for online implementation. However, the offline computational complexity to obtain the PWA controller and, possibly, to compute suitable support structure for efficient online implementation (like search trees) needs to be considered for effectiveness assessments. Issues with the complexity of explicit MPC approaches may come before the concern of how to deal with a complex PWA controller. This controller may not be available at all because of prohibitive offline computations. The approximate explicit MPC proposed in this chapter aims to address the complexity problem considering both offline and online computations. When the offline computation needed for computing the optimal explicit MPC solution is prohibitive, an approximate solution computed using just samples of the optimal one may be an effective way to deal with the problem. The Delaunay tessellation structures give properties to this approximate PWA controller which allow fast online implementation without the need for any additional support structure (this is further discussed in the next section). Though the number of pieces of the approximate PWA controller cannot be guaranteed to be smaller than the number of pieces of the optimal PWA controller (especially when many samples are added to achieve a certain tolerance), it is believed that this geometric approach may be nevertheless successful in several situations. In fact, the approach takes advantage of existing solutions to relevant problems in computational geometry. This is a branch of computer science which focuses heavily on computational complexity since the algorithms are intended to be used on large data-sets containing millions of geometrical objects (e.g. points, half-spaces, polyhedra). Effective and efficient algorithms are present in the literature to generate Delaunay tessellations, store the relative data and point locate a query point.

An approach to reduce the number of pieces of the approximate PWA solution may be to reduce the initial set of vertices to be tessellated. The number of simplices in the DT of a set of points depends on the number of points considered. In the worst case, it is known that the DT of n_p points in the d-dimensional space contains $O\left(n_p^{\lceil d/2 \rceil}\right)$ simplices.

The partition $\mathcal{D}_P$ comprises a number of simplices which depends on the number of vertices of the feasible set (together with the virtual vertices), the number of vertices of the terminal set and the number of possible points added to improve the approximation and achieve the desired accuracy. It follows that a way to obtain a smaller number of simplices in $\mathcal{D}_P$ is to consider suitable (inner) approximations of the feasible set characterized by a reduced number of vertices.

The problem of computing such approximations of the feasible set is considered in Scibilia et al. (2010b) and is one of the issues addressed in Chapter 4. The proposed solution is based on an existing approach for the computation of polytope approximations: given a polytope $\mathcal{P}$, the approximating polytope $\tilde{\mathcal{P}}$ is obtained by consecutively removing chosen vertices from $\mathcal{P}$.

The number of simplices in $\mathcal{D}_P$ can be further reduced if a simpler terminal set is also considered such that it results in a reduced number of virtual constraints introduced (which implies less virtual vertices in the feasible set and less vertices in the terminal set). This can be obtained considering an inner approximating polytope $\tilde{\Omega} \subset \Omega$, $0 \in \tilde{\Omega}$, characterized by fewer half-spaces. However, this reduces the region where the approximate controller coincides with the optimal controller, thus a suitable compromise between the simplicity of the approximating polytope and the accuracy of the approximation must be considered (Bronstein (2008)).

3.10 Online Implementation

As previously observed, explicit MPC approaches result in piecewise affine control laws, thus the online computational effort reduces to the evaluation of these piecewise affine functions. Naturally, the rate at which this problem can be solved determines the minimal sampling time of the system. Many approaches have been proposed to efficiently solve this problem, and in general they require post-processing the PWA solution to generate suitable support structures (like search trees) for fast evaluation, which also means the use of additional memory to store the corresponding data (Tøndel et al. (2003b), Spjøtvold et al. (2006), Christophersen (2007)).
The problem of evaluating piecewise affine function corresponds to a fundamental problem in computational geometry, the point location problem: given a partition of the space into non-overlapping regions, determine the region where a query point lies (de Berg et al. (2008)). In particular, the point location problem has received considerable attention in the context of Delaunay tessellations, motivated by the implications it has in many applications like computer graphics, geographic information systems, motion planning, and computer aided design. Several methods which use simple data structures have been proposed to efficiently locate a point in DTs (Devroye et al. (2004), Devroye et al. (1999)).
One of the most interesting approaches for point location in DTs is the *Jump&Walk*. This approach uses a very simple algorithm to find the simplex containing the query point. The algorithm is structured in two parts. The first part selects a starting point in the DT (*jump*). The second part locates the simplex containing the query point by traversing all simplices intersected by the line segment from the starting point to the query point (*walk*). Given the simplex containing the starting point, the procedure chooses one of the $n + 1$ neighboring simplices computing the dot products of the normal vectors of the bounding facets and the vector from the starting point to the query point. This is repeated until the simplex containing the query point is located (see also Figure 1.4 in Chapter 1).
The great advantage of this approach is that it does not require any pre-processing of the partition and no additional storage memory. The only underlying assumption is that the DT is given by an internal representation such that constant time access

between neighboring simplices is possible. This can be achieved, for example, by using edge-facet data structures for storing the tessellations (Brisson (1993), Nienhuys and van der Stappen (2003), Celes et al. (2005), Blandford et al. (2005)).
The performance of Jump&Walk depends on the selection of the starting point. Several algorithms to select such an element were proposed. A simple approach is to chose at random a number of starting points in the DT and all the time select the one characterized by the shortest distance to the query point. Other approaches exploit the possible relations between the locations. In our case an algorithm based on the latter approach[3] reasonably seems to be the most efficient way to choose starting points. In fact, at each time step relevant information about the current state location in the partition $\mathcal{D}_P$ can be obtained by the pervious state location: one possibility could be to choose the last state location as the current starting point; another possibility could be to use the last state value to compute the one-step-ahead state prediction and use this as the current starting point.
In the literature the performance of Jump&Walk algorithms on a DT composed by n randomly distributed points in the d-dimensional space is given as being proportional to $n^{1/d}$. Because of its exceptional simplicity, Jump&Walk algorithms have been successfully used in several popular software packages like Triangle, QHULL, CGAL and X3D Grid Generation System (Devroye et al. (2004), Mücke et al. (1999), Zhu (2006)).

3.11 Numerical Illustrations

In this section we consider the double integrator with input and state constraints to illustrate the results presented in the previous sections.
The model of the double integrator is one of the most important in control applications, representing single degree-of-freedom translational and rotational motion. Thus it can be used to model for instance low-friction, free rigid-body motion, such as single-axis spacecraft rotation and rotary crane motion (Rao and Bernstein (2001)).

The double integrator is given by the continuous-time linear system

$$\dot{x} = Ax + Bu \tag{3.43}$$

$$y = Cx \tag{3.44}$$

[3]We assume that such an algorithm handles that $\mathcal{D}_P$, in general, is not itself a DT of points but is formed by subregions which are individually DTs of points (cf. Section 3.7). It is easy to see that this does not represent a real issue.

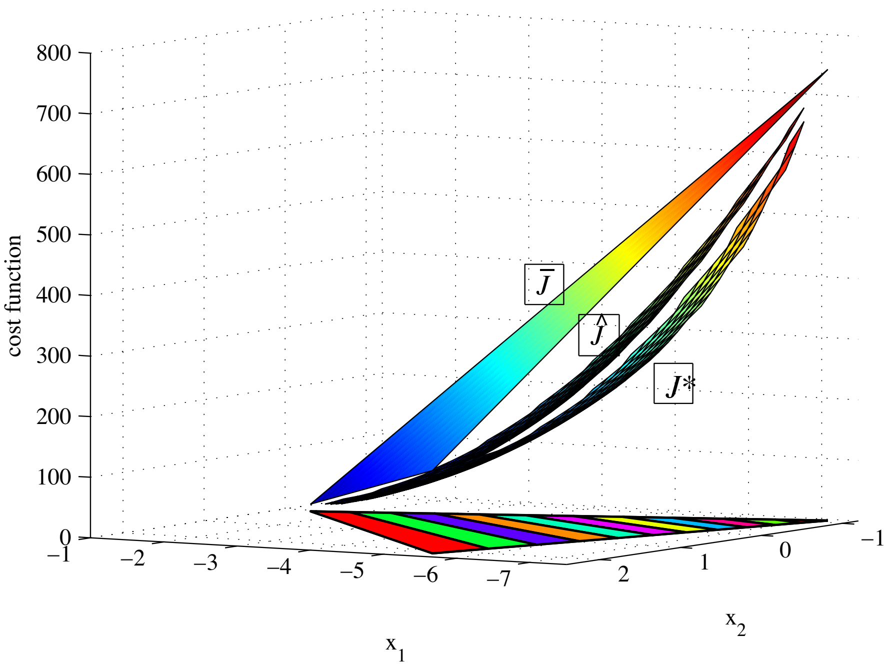


Figure 3.4: Optimal cost function J^*, approximate cost function $\tilde{J}$ and linear upper bound on the cost function $\bar{J}$ over a simplex.

where $x \equiv y \in \mathbb{R}^2$, $u \in \mathbb{R}$,

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix},\ B = \begin{bmatrix} 0 \\ 1 \end{bmatrix},\ C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.45}$$

The state components x_1 and x_2 can represent for instance the position and velocity, respectively, of a body having mass 1. Discretizing the system (3.43-3.44) with sampling time 0.3 the following discrete-time double integrator system matrices are obtained

$$A = \begin{bmatrix} 1 & 0.3 \\ 0 & 1 \end{bmatrix},\ B = \begin{bmatrix} 0.04 \\ 0.3 \end{bmatrix},\ C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{3.46}$$

The system is subject to the input constraints $-1 \leq u \leq 1$, and to the velocity constraints $-3 \leq x_2 \leq 3$. The MPC problem is tuned with weight matrices $Q = I$, $R = 1$ and horizon $N = 11$.

Figure 3.6 shows the partitions of the feasible set for the optimal PWA control law. The same figure also depicts the partitions for two suboptimal PWA control laws for

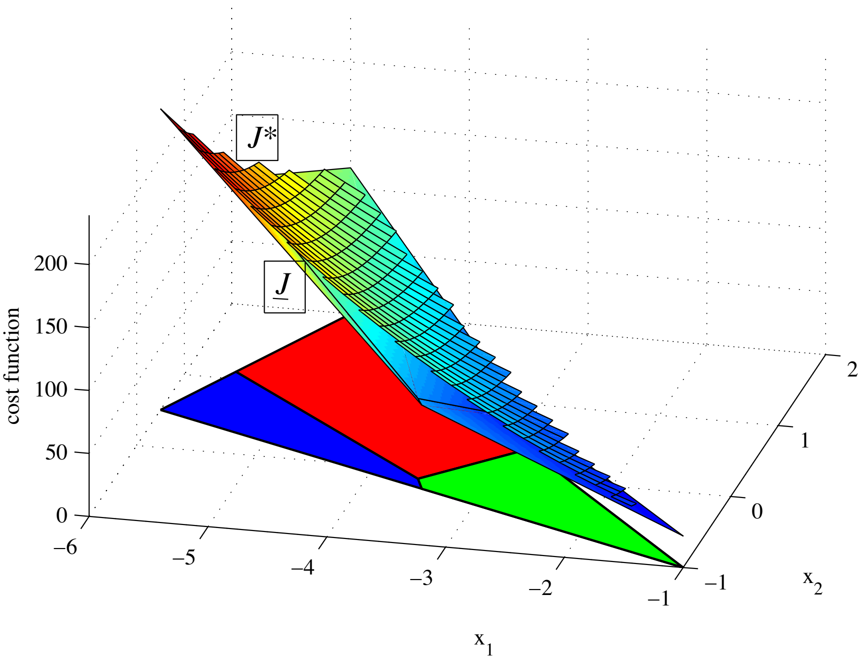


Figure 3.5: Optimal cost function J^* and piecewise linear lower bound $\underline{J}$ over a simplex.

different tolerance settings . The tolerance is measured in percentile difference and is set to 30% in one case and 100% in the other, i.e. the suboptimal cost function must not be greater than 30% and 100% of the optimal cost function, respectively. The upper bound (3.25) on the cost function error is used (Figure 3.4). Note that unnecessary vertices may have been added due to the conservative estimate (3.22). The partition obtained by considering only the vertices of the feasible set (together with the virtual vertices) and the vertices of the terminal set is also depicted in Figure 3.6. This corresponds to the partition before any extra point is added for improving the approximation, and comprises the minimum number of regions (simplices and terminal set) needed by Algorithm 2 to cover the feasible set.
Table 3.2 lists the number of regions generated by Algorithm 2 as function of the tolerance, for different tolerance settings. It also indicates the number of regions needed for theoretically guaranteed stability according to Theorem 7. However, Theorem 7 is quite conservative. Indeed, post-processing the approximate solutions according with the methods described in Hovd and Olaru (2010), it is simple to prove stability also for the approximate controller with 63 regions.
Note that the conservativeness of Theorem 7 is increased by the conservativeness of the estimate (3.22).
The last row in the table states the minimum number of regions needed to cover the feasible set (obtained by ignoring the tolerance).

Table 3.1: Number of simplices generated as function of the tolerance

Tolerance	Number of regions
100%	63
50%	107
30%	171
10%	341
for stability	369
not considered	51

Figures 3.7 illustrates the optimal PWA control law and the suboptimal PWA control law characterized by 63 regions.
The performance of the simple approximate controller characterized by 63 regions is illustrated in Figures 3.8 and 3.9, where starting from position/velocity $x_0 = [10\ 0]^T$ the corresponding optimal and suboptimal state trajectories and control inputs are shown.

The approach has been tested in numerical simulations for several systems with different state (n) and input (r) dimensions. The systems are random continuous-time linear systems all discretized with sampling time 0.1. The common MPC setting used are: horizon $N = 5$, $Q = I$ and $R = I$, where I is the identity matrix; state

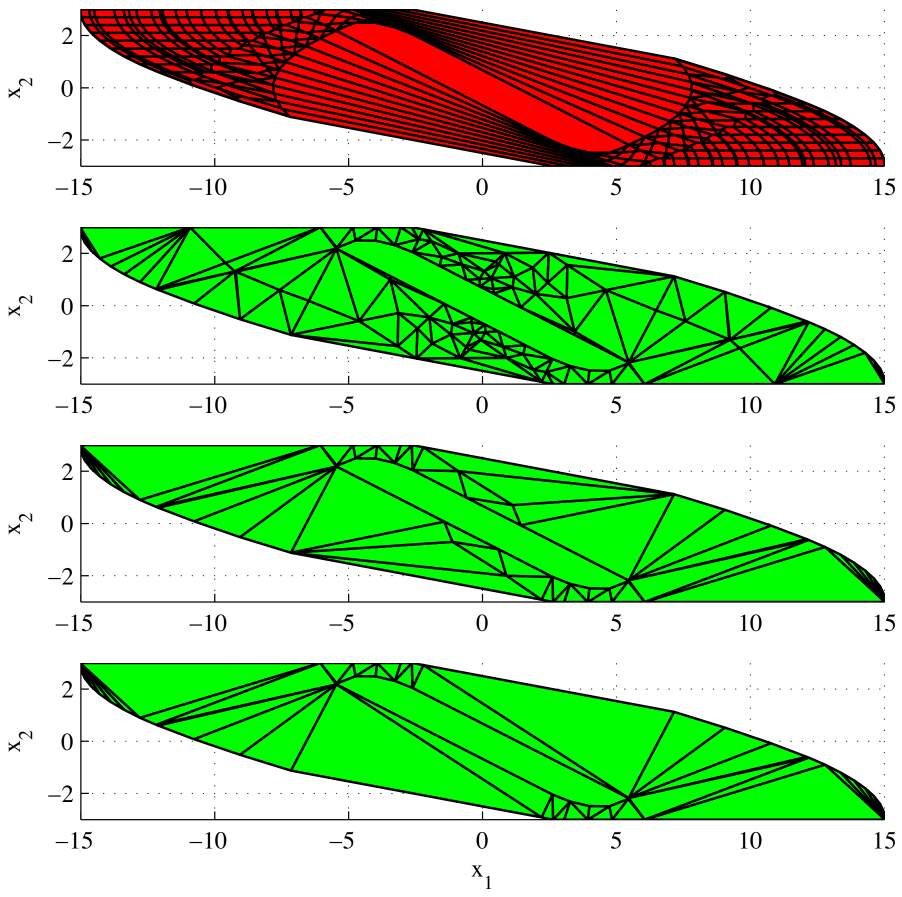
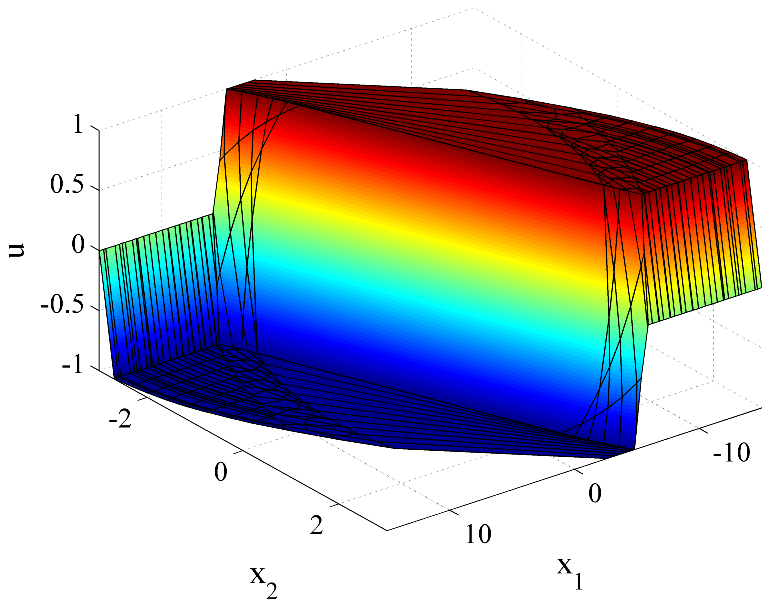


Figure 3.6: From the top, partition of the feasible set for the optimal PWA control law, number of regions: 551; partitions of the feasible set for the suboptimal PWA control laws generated by Algorithm 2, number of regions (maximum tolerance allowed): 171 (30%), 63 (100%), 51 (tolerance not considered), respectively.



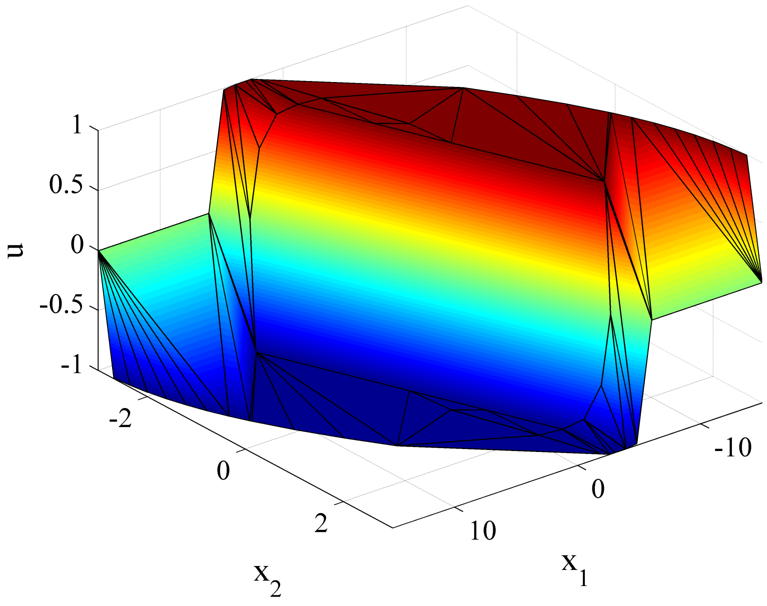


Figure 3.7: The figure on top depicts the optimal PWA control law over the partition of the feasible set. The figure below depicts the approximate PWA control law over the partition of the feasible set characterized by 63 regions.

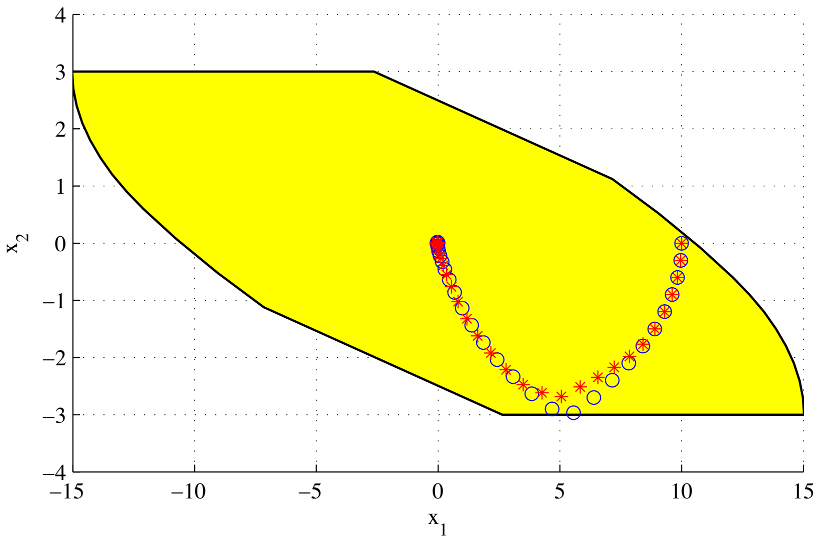


Figure 3.8: The paths of the circles (blue) and stars (red) represent the state trajectories corresponding to the exact and the approximate controller (approximate controller in Table 3.2 characterized by 63 regions), respectively, starting from position/velocity $x_0 = [10\ 0]^T$.

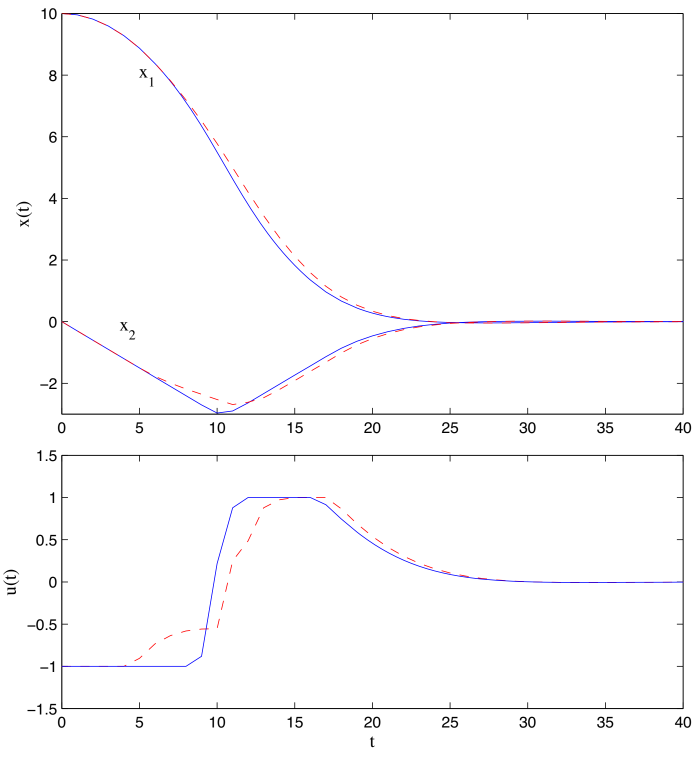


Figure 3.9: On the top graph, the solid (blue) and dashed (red) lines show the state trajectories corresponding to the exact and the approximate controller (approximate controller in Table 3.2 characterized by 63 regions), respectively. Below, the optimal control input is represented by the solid (blue) line and the approximate control input by the dashed (red) line.

constraints $-10 * \mathbb{1} \leq x \leq 10 * \mathbb{1}$, input constraints $-\mathbb{1} \leq u \leq \mathbb{1}$, where $\mathbb{1}$ is a suitably dimensioned vector with all elements equal to 1.
The approximate controllers do not consider extra points so to achieve a given tolerance. The conservativeness of the bounds (3.20) and (3.25) (and of Theorem 7) could have led to misleading results about the potential of the approach. In fact, the simulations carried out have showed often that the partition did not need any further refinement in order to achieve stability and good performance (Figure 3.10). Of course a means to guarantee stability is always desirable, but this may be achieved also without the need for accurately approximating the exact objective function (for instance using PWQ Lyapunov functions).
Table 3.2 lists some of the results obtained in the simulations. As expected from the discussion in Section 3.9, the approximate controller is not always characterized by fewer regions than the optimal controller. It is interesting to note that the simulations seem to suggest that the approach is more effective when multiple inputs are considered. The table also lists an extreme case where the computations for obtaining the optimal controller had to be stopped because the number of regions was bursting (17000 is the number of regions computed when the simulation was stopped). Contrarily, for the same system obtaining the approximate controller was not an issue.

Table 3.2: Optimal PWA controller vs approximate PWA controller for several random systems. The case marked with (*) is illustrated in Figure 3.10.

Rand. sys. (n, r)		Optimal PWA controller number of regions	Approximate PWA controller number of regions
$(3, 1)$		449	447
		487	691
		615	453
$(3, 2)$		309	155
		597	437
	(*)	3391	1995
$(3, 3)$		8217	2411
		4105	553
		> 17000	4447
$(4, 2)$		7738	5476
		4445	6591

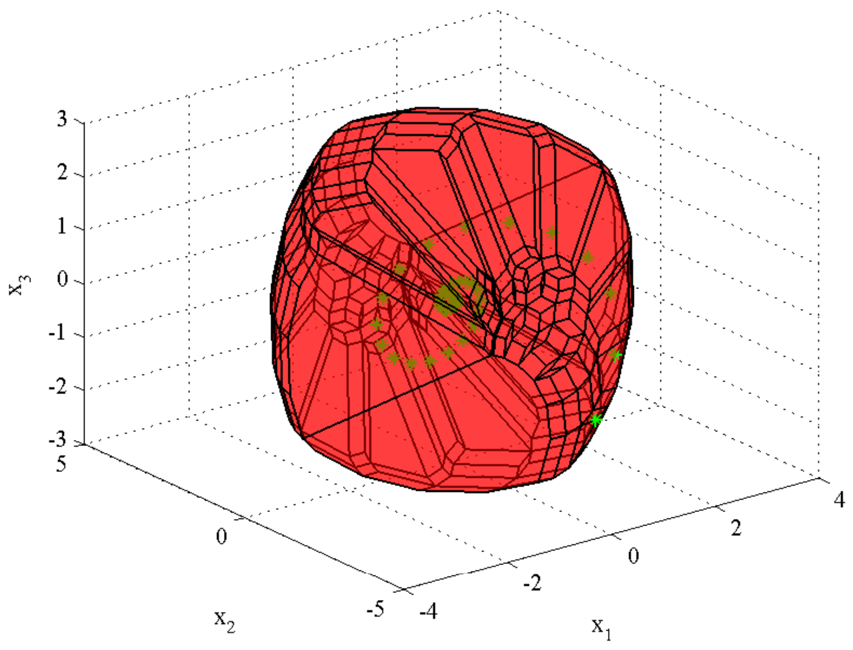


Figure 3.10: Feasible set and state trajectory corresponding to the approximate controller starting from the vertex $v = [0.24 \ -2.79 \ -2.62]^T$ for one of the random systems in Table 3.2.

3.12 Conclusion

The chapter has presented a geometric approach for deriving an approximate explicit solution to linear constrained MPC problems. The solution is optimal for the portion of the feasible set where constraints are not active, on the remaining part of the feasible set the prohibitive optimal explicit MPC solution is replaced by an approximation based on Delaunay tessellations and computed from a finite number of samples of the exact solution. Finer tessellations can be obtained so as to achieve a desired tolerance on the cost function approximation error.
The proposed approach suggests an answer to the cases where explicit MPC implementations are desired but impractical both due to the offline computational effort needed to compute the explicit optimal solution and due to their complexity for online evaluations. The approach is based on computational geometry, a branch of computer science which focuses heavily on computational complexity since the algorithms are intended to be used on large data-sets containing millions of geometrical objects. Effective and efficient algorithms are present in the literature to generate Delaunay tessellations, store the relative data and point locate a query point.
The approach uses the cost function approximation error to decide whether a finer tessellation is needed or not. Since computing this error is in general computationally demanding, the approach proposes the use of easily computable upper bounds as estimates. However, these bounds may be quite conservative, resulting in finer tessellations even if they are not really needed. A future research direction would be to find easily computable less conservative bounds.
Rigorous stability proofs were given when the maximum cost function approximation error is less than a certain tolerance. However, the conditions given are quite conservative. In fact, stability was proven post-analyzing approximate PWA controllers with cost function values that did not satisfy the theoretical maximum allowed tolerance. A further research direction would be to derive less conservative stability conditions to use in the algorithm for generating guaranteed stabilizing suboptimal PWA controllers.

Chapter 4

Feasible Sets for MPC and their Approximations

This chapter considers the problem of computing inner approximations for the feasible set for linear Model Predictive Control techniques. An alternative approach for computing the feasible set is presented, based on set relations instead of the conventional orthogonal projection. The approach can be implemented incrementally on the length of the prediction horizon. This is exploited to design an algorithm to compute suitable inner approximations. Such approximations are characterized by simpler representations and preserve the essential properties of the feasible set such as convexity, positive invariance and inclusion of the set of expected initial states. This is important when in order to avoid the online optimization, the optimal MPC solution is precomputed offline in an explicit form as a piecewise affine state feedback control law over the feasible set. Particularly in the context of finding simpler suboptimal explicit solutions the complexity of the feasible set plays a decisive role.

4.1 Introduction

Within the theoretical framework for MPC, a key role is played by the so-called *feasible set*, i.e. the largest subset of the state space such that there exists a control action satisfying all the constraints. The feasible set is closely related to the prediction horizon considered. Generally longer horizons result in larger feasible sets, but this is at the cost of a larger MPC optimization problem. Provided that the MPC optimization problem is formulated so that closed-loop stability is ensured (Mayne et al. (2000)), an optimal control action is guaranteed to exist at each sam-

pling time, for any initial state chosen in the feasible set. This also means that when explicit MPC formulations (Bemporad et al. (2002b), Tøndel et al. (2003a)) are employed, the feasible set is the domain where the optimal piecewise affine control function is defined. A well-known problem in the explicit MPC area is that finding and deploying the optimal explicit solution may be impractical in several relevant situations. This problem has been extensively tackled by the research community, which has proposed many approaches to approximate explicit MPC (see the literature overview in Chapter 3). The availability of the feasible set and furthermore its shape description play key roles in the effectiveness of many of these approaches, particularly for the ones based on feasible set discretizations (Scibilia et al. (2009b) and Scibilia et al. (2010a), Nam et al. (2010), Bemporad and Filippi (2006), Johansen and Grancharova (2003), Jones and Morari (2009)).
An important characteristic of the feasible set is the *convexity*, in fact the feasible set is completely described by the linear constraints involved in the MPC optimization problem, which places it in the specific class of convex sets called polyhedra (more precisely, polytopes). The standard approach to compute the feasible set uses an important operation in polyhedral set theory, the orthogonal projection (Burger et al. (1996), Jones et al. (2004), Mount (2002)). However, the orthogonal projection often turns out to be a computationally demanding operation in high spatial dimensions (Jones et al. (2008)). This is the case, for example, when the feasible set is computed for MPC with long prediction horizons.
The crucial property of the feasible set in the MPC context is the *positive invariance* with respect to the closed-loop system, i.e. for any initial state contained in the feasible set, the state evolution of the closed-loop system is also contained in the feasible set for all future times. In general, polyhedral sets represent an important family of candidate positively invariant sets and have been particularly successful in the solution of many control engineering problems thanks to their flexibility (Blanchini (1999), Kerrigan and Maciejowski (2000), Gilbert and Tan (1991)). However, the appurtenant disadvantage of this flexibility is the complexity of representation which may be extremely high since it is not fixed by the space dimension considered (Blanchini and Miani (2008)).
Approximating polytopes by simpler sets is a well-known problem in many research areas related to optimization, system identification and control. With any simpler representation a certain loss of information is associated in principle. Thus, in general, the ideal solution is always a right balance between simplicity and accuracy (Dabbene et al. (2003), Bronstein (2008), Gritzmann and Klee (1994b)).
This chapter, based on results in Scibilia et al. (2010b), proposes two contributions: first it suggests an alternative approach for computing the feasible set which uses set relations instead of orthogonal projection. Set relations of similar nature have also been used in Kolmanovsky and Gilbert (1995). The proposed approach

can be implemented incrementally over the length of the horizon, and proves to be computationally less demanding than the standard approach. Thereafter a solution is proposed to the problem of finding (inner) approximations of the feasible set characterized by simpler representations, which constitutes the main contribution of the chapter. When approximation approaches of polytopes are considered, while convexity is easily maintained by the family of sets we are dealing with, positive invariance is generally lost. Thus the primary issue is to compute approximations while preserving both convexity and positive invariance. Furthermore, at the design stage, one of the requirements of the controller is that it has to be able to regulate the system for a given set of initial states representing the expected initial operation conditions. Assuming that the MPC fulfills the design specifications, this set, here called the *operating set*, is contained within the feasible set. To maintain the effectiveness of the MPC, an additional issue is then to compute approximations that do not result in a loss of information which will prevent the MPC from performing acceptably for states in the operating set.

4.2 Model Predictive Control

Consider the following discrete-time linear time-invariant system:

$$x(t+1) = Ax(t) + Bu(t) \tag{4.1}$$

$$y(t) = Cx(t) \tag{4.2}$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^r$ is the control input and $y \in \mathbb{R}^m$ is the output vector, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times r}$ and $C \in \mathbb{R}^{m\times n}$, and the pair (A, B) is stabilizable. Full state measurement and no disturbances or model uncertainty are assumed.
The system is assumed to be subject to the following state and input constraints:

$$x(t) \in \mathcal{X} \subset \mathbb{R}^n \tag{4.3}$$

$$u(t) \in \mathcal{U} \subset \mathbb{R}^r \tag{4.4}$$

for all future times. The sets $\mathcal{X}, \mathcal{U}$ are considered to be described by linear inequalities on the respective variables. The origin is assumed to be an interior point for both sets.
Consider the problem of regulating the system (4.1) to the origin, such that constraints like (4.3-4.4) are satisfied.

Note that considering equation (4.2), the case of output regulation would be an immediate extension. Constraints on the output may be easily cast in terms of constraints on the state and taken into account in (4.3).

The regulation problem is solved by the finite horizon MPC

$$\min_{\mathbf{u}} \left\{ J\left(\mathbf{u}, x(t)\right) = x_N^T P x_N + \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k \right\} \tag{4.5}$$

$$\begin{array}{llll} \text{s.t.} & x_0 = x\left(t\right), & & \text{(a)} \\ & x_{k+1} = Ax_k + Bu_k, & k = 0, 1, ..., N-1, & \text{(b)} \\ & x_k \in \mathcal{X}, & k = 1, 2, ..., N-1, & \text{(c)} \\ & u_k \in \mathcal{U}, & k = 0, 1, ..., N-1, & \text{(d)} \\ & x_N \in \Omega, & & \text{(e)} \end{array} \tag{4.6}$$

where x_k denotes the predicted state vector at time $t+k$ obtained by applying the k first elements of the input sequence $\mathbf{u} \triangleq [u_0, ..., u_{N-1}]$; N is the prediction horizon; $Q \succeq 0$ (positive semidefinite) and $R \succ 0$ (positive definite) are symmetric matrices corresponding to weights on state and input; P is the terminal cost matrix and $x_N \in \Omega$ the terminal constraint, which are defined to guarantee stability. The matrix $P \succ 0$ is the solution of the algebraic Riccati equation resulting from the corresponding unconstrained LQR problem. The terminal set Ω is chosen to be feasible and positively invariant for the closed-loop system with this LQR (see Chapter 2 for details).
The MPC optimization problem (4.5-4.6) can be formulated as the following QP

$$\begin{array}{l} \min_{\mathbf{u}} \mathbf{u}^T H \mathbf{u} + 2x^T F \mathbf{u} + x^T Y x \\ \text{s.t. } G\mathbf{u} \leq w + Ex \end{array} \tag{4.7}$$

where the matrices H, F, Y, G, w and E are as defined in Chapter 2. Note that the term $x^T Y x$ can be removed from the optimization since it does not influence the optimal argument.

4.3 The Feasible Set

The MPC regulates the state to the origin for all the initial conditions contained in the feasible set. The feasible set is defined as

$$\mathcal{X}_F = \{x \in \mathbb{R}^n | \; \exists\, \mathbf{u} \text{ satisfying } (4.6)\} \tag{4.8}$$

and can be interpreted as the maximal controlled invariant set by means of the MPC with prediction horizon N and terminal set Ω.
When explicit solutions are considered, the feasible set is the domain where the piecewise affine controller is defined.

4.3.1 Computing the Feasible Set: Standard Approach

The feasible set can be completely characterized by the constraints involved in the optimization problem. As seen in the QP formulation (4.7), the constraints (4.6) can be expressed in terms of the input sequence **u** and the initial state $x(t)$:

$$G\mathbf{u} - Ex(t) \leq w \tag{4.9}$$

where G and E are matrices and w a vector of suitable dimensions (cf. Chapter 2). The linear inequalities (4.9) define a polytope in the space $\mathbb{R}^{n+rN}$

$$\mathcal{Q} = \left\{ \left[x(t)^T \; \mathbf{u}^T\right]^T \in \mathbb{R}^{n+rN} \mid G\mathbf{u} - Ex(t) \leq w \right\} \tag{4.10}$$

Then, the feasible set $\mathcal{X}_F$ is given as orthogonal projection of $\mathcal{Q}$ onto the state coordinates

$$\mathcal{X}_F = \Pi_n(\mathcal{Q}) \tag{4.11}$$

With this approach the computation of the feasible set relies essentially on the efficiency of projection algorithms. However, the orthogonal projection is intrinsically a computationally demanding operation (NP-hard), becoming increasingly prohibitive as the dimension of the projecting polytope increases (Tiwary (2008a)). This affects the computation of feasible sets for MPC, especially when long prediction horizons are considered.

4.3.2 Computing the Feasible Set: Alternative Approach

This section considers a different approach for computing the feasible set, which will be also useful for the results in the following sections and provides an alternative for the case of long prediction horizons.
Consider the optimization problem (4.5) subject to the constraints (4.6- a, b, d, e), i.e. ignoring the state constraints $x_k \in \mathcal{X}$, $k = 1, 2, ..., N-1$.
For this relaxed optimization problem, indicate with $\tilde{\mathcal{X}}_F$ the corresponding relaxed feasible set.
Note that the only constraints on the state are the equality constraints (4.6- a, b), and the terminal constraint (4.6- e): the terminal state must be contained in the terminal

set.
Using the equality constraints, the terminal state equation can be written as:

$$x_N = A^N x(t) + \hat{B}\mathbf{u} \tag{4.12}$$

where A^N is the N-matrix-power of A and $\hat{B} = \left[A^{N-1}B \; A^{N-2}B \; ... \; B\right]$.
Equation (4.12) suggests the existence of a relation in terms of the sets involved in the relaxed MPC optimization problem considered. Set relations of similar nature have been also used in Kolmanovsky and Gilbert (1995) in the context of finding admissible sets for discrete-time systems subject to bounded input disturbances.
Before formally stating the set relation, we need to introduce the set of admissible input sequences:

$$\mathcal{U}^{(N)} = \left\{\mathbf{u} \in \mathbb{R}^{rN} \mid u_j \in \mathcal{U}, j = 0, ..., N-1\right\}. \tag{4.13}$$

Theorem 8. *Consider the optimization problem (4.5) subject to the constraints (4.6-a, b, d, e). Then the terminal set, Ω, the corresponding feasible set, $\tilde{\mathcal{X}}_F$, and the set of admissible sequence input, $\mathcal{U}^{(N)}$, satisfy the following set relation*

$$\Omega = A^N \tilde{\mathcal{X}}_F \ominus \hat{B}(-\mathcal{U}^{(N)}) \tag{4.14}$$

where $A^N : \mathbb{R}^n \mapsto \mathbb{R}^n$ and $\hat{B} : \mathbb{R}^{rN} \mapsto \mathbb{R}^n$ represent linear maps applied respectively to $\tilde{\mathcal{X}}_F$ and to $\mathcal{U}^{(N)}$.

Proof. According to (4.8), the relaxed feasible set can be written as:

$$\tilde{\mathcal{X}}_F = \left\{x \in \mathbb{R}^n | \exists \mathbf{u} \in \mathcal{U}^{(N)} : \; A^N x + \hat{B}\mathbf{u} \in \Omega\right\} \tag{4.15}$$

Using the linear maps A^N and $\hat{B}$ we can define the sets:

$$A^N \tilde{\mathcal{X}}_F = \left\{x_e \in \mathbb{R}^n | x_e = A^N x, \; x \in \tilde{\mathcal{X}}_F\right\} \tag{4.16}$$

$$\hat{B}\mathcal{U}^{(N)} = \left\{x_u \in \mathbb{R}^n | x_u = \hat{B}\mathbf{u}, \mathbf{u} \in \mathcal{U}^{(N)}\right\} \tag{4.17}$$

From (4.15) we can write the equivalence:

$$A^N \tilde{\mathcal{X}}_F = \left\{x_e | \exists x_u \in \hat{B}\mathcal{U}^{(N)} : \; x_e + x_u \in \Omega\right\} \tag{4.18}$$

which subsequently implies that $\tilde{\mathcal{X}}_F$ is the collection of all the states that can be obtained as a combination of points in Ω and $\hat{B}(-\mathcal{U}^{(N)})$. This leads to the equivalence:

$$\Omega = \left\{x_\Omega | \; x_\Omega - x_u \in A^N \tilde{\mathcal{X}}_F, \; \forall x_u \in \hat{B}\mathcal{U}^{(N)}\right\} \tag{4.19}$$

By the definition of the erosion operator (cf. Chapter 2), (4.19) corresponds to (4.14). □

In the following we assume that the matrix A of (4.1) is invertible. Notice that zero eigenvalues of A mean that there are modes which are pure delays of the inputs. This is clear by taking the Jordan form of A, which also gives linearly transformed constraint sets $\mathcal{X}'$ and $\mathcal{U}'$ (the constraints on the state corresponding to delayed inputs must then be compatible with $\mathcal{U}'$). Then, the assumption is motivated by considering that the Jordan blocks of A with zero eigenvalue can then be excluded, meaning that constraints involving linear combinations of past inputs and current states corresponding to the remaining Jordan blocks are not allowed. However, we also note that the assumption is always satisfied by discretized (finite dimensional) continuous-time systems.
Therefore, by (4.14) the relaxed feasible set can be computed as:

$$\tilde{\mathcal{X}}_F := \left(A^N\right)^{-1}\left[\Omega \oplus \hat{B}(-\mathcal{U}^{(N)})\right] \tag{4.20}$$

Note that the computation of the feasible set via the formula (4.20) basically costs a Minkowski sum in $\mathbb{R}^n$, given the polytopes Ω and $\mathcal{U}^{(N)}$, which is an operation that can be done in polynomial time (Gritzmann and Sturmfels (1993)). This is generally more convenient than using (4.11) which requires handling polytopes in higher dimensions, $\mathbb{R}^{n+rN}$.

Inspecting (4.14), an incremental approach for computing $\tilde{\mathcal{X}}_F$ can be derived, which with a simple modification is extendable for also computing $\mathcal{X}_F$.
Let us explicitly express the dependence of the feasible set from the length of the horizon as $\mathcal{X}_F^{(k)}$, which indicates the feasible set for a horizon of length k. According to this notation, the feasible set we are interested in is $\mathcal{X}_F = \mathcal{X}_F^{(N)}$.
For the case $k = 1$, relation (4.14) becomes

$$\Omega = A\tilde{\mathcal{X}_F}^{(1)} \ominus B(-\mathcal{U}) \tag{4.21}$$

which leads to the set[1] of all the initial states that in one time-step move inside Ω

$$\tilde{\mathcal{X}}_F^{(1)} = (A)^{-1}\left[\Omega \oplus B(-\mathcal{U})\right] \tag{4.22}$$

At this point, introducing the constraint on the state is straightforward, and thus the feasible set $\mathcal{X}_F^{(1)}$ is simply computed from $\tilde{\mathcal{X}}_F^{(1)}$ as

$$\mathcal{X}_F^{(1)} = \tilde{\mathcal{X}}_F^{(1)} \cap \mathcal{X} \tag{4.23}$$

The feasible set $\tilde{\mathcal{X}}_F^{(2)}$ is determined by $\tilde{\mathcal{X}}_F^{(1)}$ considering an analogous relations to (4.21):

$$\tilde{\mathcal{X}}_F^{(1)} = A\tilde{\mathcal{X}}_F^{(2)} \ominus B(-\mathcal{U}) \tag{4.24}$$

[1]This set is also called the one-step controllability set to Ω (Blanchini (1994)).

which gives

$$\tilde{\mathcal{X}}_F^{(2)} = (A)^{-1}\left[\tilde{\mathcal{X}}_F^{(1)} \oplus B(-\mathcal{U})\right] \tag{4.25}$$

and thus also $\mathcal{X}_F^{(2)}$ can be determined analogously to (4.23).
In general, the feasible set with horizon k can be computed in this incremental fashion from the feasible set with horizon $k-1$. This leads to Algorithm 3 for computing the feasible set for the MPC (4.5-4.6).

Algorithm 3: Feasible set

```
  Input: The system state and input matrices $A$ and $B$. The terminal set
         $\Omega$. The state and input constraints sets $\mathcal{X}$ and $\mathcal{U}$. The length of
         the horizon $N$.
  Output: The feasible set $\mathcal{X}_F$
1 Initialize the set $\mathcal{T} = \Omega$;
2 for $k = 1$ to $N$ do
3     Compute $\tilde{\mathcal{X}}_F^{(k)} = A^{-1}[\mathcal{T} \oplus B(-\mathcal{U})]$;
4     Compute $\mathcal{X}_F^{(k)} = \tilde{\mathcal{X}}_F^{(k)} \cap \mathcal{X}$;
5     Set $\mathcal{T} = \mathcal{X}_F^{(k)}$
6 end
7 Set $\mathcal{X}_F = \mathcal{X}_F^{(k)}$.
```

In addition to giving the possibility to include the state constraints, the advantage of the incremental approach is that it avoids the necessity of handling the polytope $\mathcal{U}^{(N)}$ which, especially for long horizons, may be undesirable.

4.3.3 The Operating Set

Reasonably, we can assume that the MPC (4.5)-(4.6) is designed to regulate the system for a given set of initial states which represents the expected initial operating conditions. Without any particular restriction, we can consider this set expressed as linear inequalities, and thus represented by a polytope $\mathcal{X}_o$. We call $\mathcal{X}_o$ the *operating set*, and assuming the MPC meets the design specifications, we have $\mathcal{X}_o \subset \mathcal{X}_F$. Naturally, the origin is an interior point of the operating set.

4.4 Approximation of Feasible Sets

The feasible set is represented by a polytope in the state space. The problem of finding polytope approximations by means of simpler convex bodies arises in many research areas related to optimization, system identification and control. In general, the solution is a balance between the *simplicity* of the representation and the *accuracy* of the approximation, where the accuracy can be measured using different metrics, depending on the particular approach used to solve the problem. An example is the work in Dabbene et al. (2003), where the authors provide algorithms for computing inner approximations in terms of the largest ellipsoids inscribed. However, more often it is required that the approximating convex body is itself a polytope. In this direction, different approaches have been proposed in the literature, and reference is made to Bronstein (2008) and Gritzmann and Klee (1994b) (and references therein) for surveys on the subject.
The common representation complexity indexes of a polytope are the number of half-spaces (or facets) for the $\mathcal{H}$-representation, and the number of vertices for the $\mathcal{V}$-representation. Since a polytope is characterized by unique minimal $\mathcal{H}$- and $\mathcal{V}$-representations, any lower complexity representation must correspond either to an inner or to an outer approximation of the polytope.
In this section, the scope is to approximate the feasible set by means of a simpler polytope. Since the feasible set corresponds to the maximal feasible controlled invariant set by means of the MPC, no feasible outer approximations can exist, and therefore attention is restricted only to the search for inner approximations. Note that the task is more involved than finding simpler representations maintaining a prescribed accuracy in the approximation. For control purposes it is of prime importance that the approximating polytope preserves the positive invariance property and contains the operating set.
A natural approach for computing inner approximations is based on the fact that, for any polytope, the omission of any of the vertices from the $\mathcal{V}$-representation changes the polytope by reducing it[2]. Typically (but not necessarily), the approximations thus obtained also result in lower complexity $\mathcal{H}$-representations, as will be discussed later. Furthermore, several situations can be recognized where a simpler feasible set characterized by fewer vertices would provide immediate improvements. This is the case for example in approaches to explicit MPC such as Scibilia et al. (2009b), Hovd et al. (2009), Nam et al. (2010) and Jones and Morari (2009), in approaches to multi-parametric convex programming such as Bemporad and Filippi (2006)), or also in control approaches as Gutman and Cwikel (1986), where the solution depends strictly on the complexity of the feasible set in terms of the

[2]Dually, the omission of any of the half-spaces from the $\mathcal{H}$-representation changes the polytope by enlarging it.

number of vertices.
Therefore, interest is focused on finding appropriate inner approximations characterized by a reduced number of vertices.

An algorithm for computing inner approximations of polytopes based on the removal of vertices is proposed in Reisner et al. (2001) (in Lopez and Reisner (2002) if only 3D polytopes are considered). The fundamental result is the following.

Proposition 1. *Given a polytope $\mathcal{P} \in \mathbb{R}^n$ characterized by $n_\mathcal{V}$ vertices $V_\mathcal{P} = \{v^{(1)}, ..., v^{(n_\mathcal{V})}\}$, $\mathcal{P} = \texttt{conv}(V_\mathcal{P})$, there exists a vertex $v \in V_\mathcal{P}$ such that the polytope $\mathcal{Q} = \texttt{conv}(V_\mathcal{P} \setminus \{v\})$ satisfies*

$$\frac{\texttt{vol}(\mathcal{P}) - \texttt{vol}(\mathcal{Q})}{\texttt{vol}(\mathcal{P})} \leq \alpha(n) n_\mathcal{V}^{-\frac{n+1}{n-1}}. \tag{4.26}$$

The factor $\alpha(n)$ is a constant depending only on the space dimension (details about how to estimate this constant can be found in Reisner et al. (2001) and Lopez and Reisner (2002)).
This result is the best possible in general, for the dependence on the numbers of vertices of the approximating polytope.
The main idea of the algorithm is thus the consecutive removal of the chosen vertices. Taking an appropriate number $k < n_\mathcal{V}$, it can be identified a successive minimizing choice of vertices of $\mathcal{P}$, i.e. a sequence $\{v^{(r_1)}, ..., v^{(r_{n_\mathcal{V}-k})}\}$ of different vertices in $V_\mathcal{P}$ such that for all $i = 1, ..., n_\mathcal{V} - k$

$$\texttt{vol}\left(\texttt{conv}\left(V_\mathcal{P} \setminus \{v^{(r_1)}, ..., v^{(r_{i-1})}\}\right)\right) - \texttt{vol}\left(\texttt{conv}\left(V_\mathcal{P} \setminus \{v^{(r_1)}, ..., v^{(r_i)}\}\right)\right) \tag{4.27}$$

is minimal over all choices of $v^{(r_i)} \in V_\mathcal{P} \setminus \{v^{(r_1)}, ..., v^{(r_{i-1})}\}$. The polytope

$$\mathcal{Q} = \texttt{conv}(V_\mathcal{P} \setminus \{v^{(r_1)}, ..., v^{(r_{n_\mathcal{V}-k})}\}), \tag{4.28}$$

characterized by k vertices, is an inner approximation of $\mathcal{P}$. The accuracy of the approximation obtained is measured by the difference of volume between $\mathcal{P}$ and $\mathcal{Q}$ and, in general, it is the best possible obtainable by any polytope with k vertices (up to the dimension dependent constants involved).
More details about the implementation of the algorithm can be found in Reisner et al. (2001) and Lopez and Reisner (2002).
The following presents an approach to extend algorithms like the one found in Reisner et al. (2001) in order to meet the primary objective of maintaining the fundamental properties of the feasible set.

Given a vertex v of $\mathcal{P}$, indicate with $\texttt{adj}(v)$ all the vertices adjacent to v, i.e. all the vertices of $\mathcal{P}$ which share a facet with v.

Proposition 2. *The region excluded from $\mathcal{P}$ when the vertex v is omitted from its $\mathcal{V}$-representation is given by*

$$\mathcal{L}_v = \texttt{conv}(\{v, \texttt{adj}(v)\}) \setminus \texttt{conv}(\{\texttt{adj}(v)\}) . \tag{4.29}$$

Proof. Naturally, $\mathcal{L}_v$ is characterized only by the facets of $\mathcal{P}$ incident in v. These facets comprise a region given by the convex hull of v and all its adjacent vertices. Since the adjacent vertices of v still remain vertices of $\mathcal{P}$, the prospective region identified solely by these vertices needs to be removed from the description of $\mathcal{L}_v$. The remaining vertices of $\mathcal{P}$, i.e. the vertices of the polytope $\mathcal{P} \setminus \texttt{conv}(\{v, \texttt{adj}(v)\})$, are not affected by the omission of v. □

In general, $\mathcal{L}_v$ is non-convex, but can be represented as a finite collection of polytopes.
Let us now consider the polytope $\mathcal{P}$ as our feasible set $\mathcal{X}_F$, i.e. $\mathcal{P} \equiv \mathcal{X}_F$. Since the convexity property is simply maintained by removing a vertex, attention is turned to the problems of how to preserve positive invariance and how to maintain the states comprising the operating set.

4.4.1 Preserving Positive Invariance

The difficulty in preserving positive invariance comes from the fact that we have to take into consideration the nonlinear dynamics of the closed-loop system with the MPC.
The *next time-step* feasible set $\mathcal{X}_F^+$ is defined as follows

$$\mathcal{X}_F^+ = \left\{x^+ | x^+ = Ax + Bu_0^*,\ x \in \mathcal{X}_F\right\} \tag{4.30}$$

where u_0^* is the first element of the MPC optimal control sequence at x.
The asymptotic (exponential) stability of the MPC guarantees that $\mathcal{X}_F^+ \subset \mathcal{X}_F$ and that $\mathcal{X}_F^+$ is positively invariant for the closed-loop system. We can now define the set $\mathcal{X}_\mathcal{N} = \mathcal{X}_F \setminus \mathcal{X}_F^+$, which has the interesting property to contain only points of the feasible set that are exclusively initial states of state evolutions starting inside the feasible set. In other words, considering any possible state evolution in $\mathcal{X}_F$, each state in $\mathcal{X}_\mathcal{N}$ can only be a starting point of it.

Theorem 9. *Any inner approximation of the feasible set obtained as convex hull of vertices inside $\mathcal{X}_{\mathcal{N}}$, such that also all the facets are inside $\mathcal{X}_{\mathcal{N}}$, preserves the positive invariance.*

Proof. Consider a set of points $V_{\mathcal{N}}$ inside $\mathcal{X}_{\mathcal{N}}$, such that $\texttt{conv}(V_{\mathcal{N}})$ has all the facets within $\mathcal{X}_{\mathcal{N}}$. It follows that $\mathcal{X}_F^+ \subset \texttt{conv}(V_{\mathcal{N}})$. Therefore, for any starting point inside such a polytope, the state evolution either moves in one step inside $\mathcal{X}_F^+$ or is already inside $\mathcal{X}_F^+$, which shows positively invariance of $\texttt{conv}(V_{\mathcal{N}})$. □

The property of positive invariance could then be preserved if for every vertex $v^{(r)}$ removed from $\mathcal{X}_F$, the condition $\mathcal{L}_{v^{(r)}} \subset \mathcal{X}_{\mathcal{N}}$ is satisfied. In fact, this ensures that the remaining vertices satisfy the requirements of Theorem 9.
However, because of the set $\mathcal{X}_F^+$, computing the set $\mathcal{X}_{\mathcal{N}}$ in general involves a substantial computational effort, which drastically reduces the applicability of the approach. Indeed, note from (4.30) that the definition of the next time-step feasible set implies the knowledge of the optimal input for the states in the feasible set. In particular, since the feasible set is a convex set, only the knowledge of the optimal control input on the border of $\mathcal{X}_F$ is needed for the computation of the next time-step feasible set. Nevertheless, this still comports a computational burden which may compromise the effectiveness of the overall approach. A further undesirable aspect of using the next time-step feasible set is that $\mathcal{X}_F^+$ is, in general, non-convex (Blanchini (1994))(Figure 4.2). This results in more difficulties in the computation of $\mathcal{X}_N$ as the intersection between non-convex sets is more involved than the intersection between polytopes, even if the possible resulting non-convex set can still be expressed as a collection of polytopes.
This issue can be easily overcome considering the following relation

$$\mathcal{X}_F^+ \subseteq \mathcal{X}_F^{(N-1)} \subset \mathcal{X}_F \tag{4.31}$$

Note that $\mathcal{X}_F^{(N-1)}$ is easily available using an incremental procedure such as Algorithm 3 for computing the feasible set. Moreover it is always convex and it is positively invariant for the closed-loop system. Then, the proposed solution is to use $\mathcal{X}_F^{(N-1)}$ in place of $\mathcal{X}_F^+$. The conservativeness introduced is not severe for the purpose here considered, $\mathcal{X}_F(N-1)$ being a tight outer approximation of $\mathcal{X}_F^+$ (Blanchini (1994)). Defining the set $\bar{\mathcal{X}}_{\mathcal{N}} = \mathcal{X}_F \setminus \mathcal{X}_F^{(N-1)}$, the property of positive invariance is preserved if for every vertex $v^{(r)}$ removed from $\mathcal{X}_F$, the following condition is satisfied

$$\mathcal{L}_{v^{(r)}} \subset \bar{\mathcal{X}}_{\mathcal{N}}. \tag{4.32}$$

In fact, condition (4.32) ensures that the remaining vertices satisfy the requirements of Theorem 9, whose results are valid if $\bar{\mathcal{X}}_{\mathcal{N}}$ is considered instead of $\mathcal{X}_{\mathcal{N}}$.

4.4.2 The Operating Set Condition

The goal of algorithms like the one developed in Reisner et al. (2001) is to find the polytope $\mathcal{Q}$ characterized by k vertices that best approximate the polytope $\mathcal{P}$ (characterized by $n_{\mathcal{V}} > k$ vertices). The accuracy of the approximation is given by the difference of volume between $\mathcal{P}$ and $\mathcal{Q}$. The introduction of condition (4.32) (invariance) changes the degrees of freedom in the minimization of the difference of volume. Generally, not all the vertices in the successive minimizing choice of vertices of $\mathcal{P}$ satisfy the necessary condition (4.32) and, therefore, not all can be removed.
When the focus is on feasible sets, the loss of volume may not necessarily be a critical issue in itself, since practically it would be more of interest that the approximating feasible set still contains the operating set $\mathcal{X}_o$. This objective can be achieved simply by checking that every time a vertex $v^{(r)}$ is removed from $\mathcal{X}_F$, the corresponding excluded region does not comprise any part of the operating set,

$$\mathcal{X}_o \cap \mathcal{L}_{v^{(r)}} = \emptyset. \tag{4.33}$$

If a certain vertex does not satisfy (4.33), then it is not removed and the next vertex is considered.

4.4.3 Removing Vertices

Suppose that the interest is to remove as many vertices as possible as long as conditions (4.32) (invariance) and (4.33) (operating set inclusion) are satisfied. This can be done iteratively: at each iteration, among all the current vertices which makes (4.32) and (4.33) satisfied, remove the one that results in the lowest loss in terms of volume.
Note that for any vertex $v^{(r)}$, conditions (4.32) and (4.33) and the volume loss can be evaluated locally, i.e. only $v^{(r)}$ and $\texttt{adj}(v^{(r)})$ are involved in the computation of $\mathcal{L}_{v^{(r)}}$, as can be seen from Proposition 2.
An efficient way to implement the algorithm is to use structures similar to pointers. Given the list of vertices $V_{\mathcal{X}_F}$ characterizing the feasible set, where each element on the list is identified by the position number, two structures can be defined:

1. `Index`, a list containing numbers referring to vertices in $V_{\mathcal{X}_F}$ (list of pointers).

2. `Adj`, a structure containing the adjacency information. $\texttt{Adj}(\texttt{Index}(i))$ gives the list of pointers to the vertices in $V_{\mathcal{X}_F}$ adjacent to the vertex with pointer $\texttt{Index}(i)$.

Then the operation of removing a vertex $v^{(r)} \in V_{\mathcal{X}_F}$ with pointer, say, $\texttt{Index}(i)$, can be done removing the i-th element from `Index`, after having removed the $\texttt{Index}(i)$-th element from `Adj` and updated the elements $\texttt{Adj}(j)$, for all $j \in \texttt{Adj}(\texttt{Index}(i))$. The update is done as follows. Each vertex j is also vertex of the polytope $\mathcal{R} = \texttt{conv}(\texttt{Adj}(\texttt{Index}(i)))$, then for each list $\texttt{Adj}(j)$ the reference $\texttt{Index}(i)$ is removed and the adjacencies resulting from $\mathcal{R}$ are added.
The advantage of using the pointer structures is to allow each iteration to simply update only the data affected by the current vertex removal.

4.4.4 Discussion on the Complexity Indexes

In general, the number of half-spaces may be much higher than the number of vertices, and vice versa. Thus, a natural question would be how will the reduction of the complexity in the $\mathcal{V}$-representation affect the complexity in the $\mathcal{H}$-representation. While in 2 and 3 dimensions there exist simple relations between the two complexity indexes, in higher dimension analytical relations are very difficult to define (Matousek (2002)). Thus, giving an exact answer to the question is a hard problem. However, a well-known achievement in the theory of convex polytopes allows giving an answer in terms of upper bounds: a polytope in the n-dimensional space with $n_{\mathcal{V}}$ vertices has at most $2\binom{n_{\mathcal{V}}}{\lfloor n/2 \rfloor}$ half-spaces. Thus, for a fixed space dimension n the number of half-spaces has an order of magnitude of $n_{\mathcal{V}}^{\lfloor n/2 \rfloor}$ ("upper bound" theorem Matousek (2002)). The upper bound theorem refers to worst case scenarios. Certainly, not all polytopes exhibit this extreme behavior, for example it is known that if n_p points are chosen uniformly at random in the unit n-dimensional ball, then the expected number of half-spaces of their convex hull is only of order of magnitude of n_p (Matousek (2002)). Thus, even if there exist cases where the omission of a vertex causes an increase in the number of half-spaces, it is reasonable to expect that typically a certain reduction of complexity in terms of number of vertices also provides a fairly relevant reduction of the complexity in terms of number of half-spaces, in the sense that the upper bound on the number of half-spaces decreases.

4.5 Discussion on Computational Complexity

Both Algorithm 3 for computing feasible sets and the approach proposed in Section 4.4 for computing simplifications of feasible sets are based on basic geometric

operations on polytopes[3]: Minkowski sum, intersection, convex hull and volume computation. Therefore, it is interesting to discuss some aspects connected with the computational complexity of these operations so to provide with additional insight into the algorithmic behavior of the approaches presented. It should be noted, though, that the scope of this section is not to give a comprehensive discussion on the computational complexity of each operation.
Every polytope admits two equivalent representation forms (cf. Chapter 2): the $\mathcal{V}$-representation (using vertices) and the $\mathcal{H}$-representation (using half-spaces). For polytopes, the representation conversion from $\mathcal{H}$- to $\mathcal{V}$-representation (vertex enumeration) and the conversion from $\mathcal{V}$- to $\mathcal{H}$-representation (facet enumeration or convex hull computation) are computationally equivalent and, in general, are difficult operations (NP-hard). The computational complexity increases fast with the number of half-spaces and/or the number of vertices involved (Khachiyan et al. (2008), Fukuda (2004)).
Often, operations on polytopes that are easy to perform in one representation become difficult if the polytopes are instead in the other representation. The Minkowski sum of two polytopes is a computationally easy operation (polynomial time) when the two polytopes are in $\mathcal{V}$-representation while becomes a difficult operation (NP-hard) when they are in $\mathcal{H}$-representation (Gritzmann and Sturmfels (1993), Tiwary (2008b)). This means that the known Minkowski sum algorithms operating on polytopes in the $\mathcal{H}$-representation show a computational complexity which increases fast with the number of half-spaces of the operands.
On the other hand, the intersection of two polytopes given in the $\mathcal{H}$-representation is an easy operation while becomes a difficult operation (NP-hard) with polytopes in the $\mathcal{V}$-representation (Tiwary (2008b)).
Resorting to a representation conversion is often not a solution to reduce complexity since the operation is itself hard.

Let us consider Algorithm 3. At each iteration the computational complexity is basically determined by a Minkowski sum and an intersection operation. Let us assume that the interest is to obtain a feasible set in the $\mathcal{V}$-representation (which here may be motivated by the procedure for computing simplified feasible sets discussed in Section 4.4). Therefore, the computational complexity of Algorithm 3 would increase fast with the number of vertices considered due to the intersection operation. Note that if the interest is in a feasible set in the $\mathcal{H}$-representation, then the Minkowski sum would be computationally costly. This suggests that computing the feasible set is intrinsically a hard problem. Given that polytopes, in general, are far more complex in terms of number of vertices and facets in higher dimensions

[3]A polytope is a bounded polyhedral set. Strictly speaking, here the operations are on general polyhedral sets defined by state and input constraints. However, in practice these sets are always bounded.

(cf. Section 4.4.4), it is reasonable to expect that the computational complexity for computing the feasible set increases fast with the dimension of the state space, n. The computational advantage of the proposed approach in respect to the traditional one is that at each iteration the operands are polytopes of dimension n. Instead, the traditional approach requires the projection of a polytope of dimension $n + Nr$ (cf. Section 4.3.1) which is easily a prohibitive operation even for small n since the polytope dimension depends also from the horizon length N and the input dimension r. Also, when the standard approach is implemented incrementally on the horizon length, at each iteration the projection of a polytope of dimension $n + r$ is required, which may still be prohibitive.
This is illustrated in the next section, where the results from several numerical tests are reported. The proposed approach started to require considerable computations for $n > 4$, while the standard approach (implemented incrementally) started to be prohibitive already for $n > 3$ (and $r > 1$).

Analogous considerations can be made for the approach proposed in Section 4.4 for computing simplifications of feasible sets. Assume that the interest is to remove as many vertices as possible as long as conditions (4.32) (invariance) and (4.33) (operating set inclusion) are satisfied (cf. Section 4.4.3). The algorithm requires initially the computation of the volume loss associated with each vertex removal. The polytope volume computation, either in the $\mathcal{V}$- or the $\mathcal{H}$-representation, is not a difficult operation (polynomial time) (Gritzmann and Klee (1994a)). Then, at each iteration the intersection operation is used to check the conditions (4.32) and (4.33): the conditions are first inspected on the vertex which currently means the lowest loss of volume, continuing with the vertex causing the second lowest loss if the former does not satisfy the conditions, and so on until a suitable vertex is identified for removal (or none, in which case the algorithm terminates). The use of pointer structures allows easily updating just the volumes affected by the current vertex removal. At each iteration the intersections are the most expensive operations to perform, and although they are done on relatively simple polytopes, these may require relevant computation especially when n increases (since this typically means high number of vertices with a complex map of adjacency). Note, however, that here the operation may be implemented in a more efficient way since it is not needed to actually compute the intersection polytope, but rather to decide whether the two polytope operands intersect or not.
In the numerical tests discussed in the next section the computations have started to become considerable for $n > 4$.

4.6 Numerical Illustrations

This section provides some examples in order to illustrate the results presented in the previous sections. It also presents an application of the results to the computation of approximate explicit MPC according to the approach presented in Chapter 3.

4.6.1 Feasible Set Computation

The computation time efficiency of the proposed approach based on set relations (SR) has been compared with the standard approach based on projection (P) in Matlab by using the Multi-Parametric Toolbox (MPT)[4] (Kvasnica et al. (2006)). Both the algorithms have been implemented incrementally on the horizon length. The incremental implementation is inherent in the set relation-based approach, while for the traditional projection-based approach it may speed up the calculation in many situations.
Extensive simulations have been carried out on several random systems for different state (n) and input (r) dimensions. The common MPC settings used are: $Q = I$ and $R = I$ where I represents the identity matrix; state constraints $-10*\mathbb{1} \leq x \leq 10*\mathbb{1}$, input constraints $-\mathbb{1} \leq u \leq \mathbb{1}$, where $\mathbb{1}$ is a suitably dimensioned vector with all elements equal to 1.
Table 4.1 reports some of the results obtained during the simulations to give a picture of the typical performance from both approaches. In general the proposed approach has performed significantly more efficiently than the standard approach. Furthermore, the projection approach led several times to unsatisfactory results such as no result after one hour of computation or numerical errors.

Remark 6. It must be noted that the proposed approach also led numerical errors, though in a considerably lower number of cases than the projection approach. This happened mainly when high-dimension state spaces ($n > 4$) were involved in the computations. The explanation lies in the dependence on the MPT routines used for computing the Minkowski sum and the intersection which, although they have demonstrated to be algorithmically more robust than the MPT projection routines used, they may fail for particularly complex high dimensional polytopes. It is reasonable to believe that the numerical issues faced could be removed by a careful

[4]MPT for Matlab offers several algorithms for computing the projection of a polytope (vertex enumeration/convex hull-based method, Fourier-Motzkin elimination, iterative hull, block elimination, equality set projection). In the simulations, the MPT projection function has been set to automatically select the best method.

re-implementation of these routines. Also, Minkowski sum and intersection are operations which are, in practice, most often used on 2/3-dimensional polytopes (several algorithmic implementations optimized for these dimensions exist). An optimized code for higher dimensional polytopes may give better computational performance (cf. Section 4.5).

Table 4.1: Set relation-based approach vs. projection-based approach. Time computation measured in seconds. † indicates no result after 1 hour computation. ‡ indicates simulation terminated by Matlab errors.

Rand. sys.	N=5	N=7	N=10
(n, r)	SR−P	SR−P	SR−P
$(3, 1)$	$0.14 - 0.40$	$0.55 - 2.70$	$2.01 - 19.33$
	$0.20 - 0.52$	$0.26 - 0.30$	$1.91 - 1.72$
$(3, 2)$	$0.73 - 5.53$	$4.92 - 12.10$	$3.72 - 19.06$
	$4.18 - 12.72$	$2.20 - 4.79$	$9.21 - 96.64$
$(4, 1)$	$4.76 - 25.63$	$21.30 - 32.67$	$59.21 -$ †
	$8.82 - 45.81$	$6.89 - 15.81$	$26.61 - 475.24$
$(4, 2)$	$34.85 - 291.7$	$121.5 -$ ‡	$47.09 - 1067$
	$38.59 - 88.10$	$147.4 -$ †	$998.09 -$ ‡

4.6.2 Feasible Set Approximation

To show the validity of the idea for computing simpler approximate feasible sets for MPC, the following considers the double integrator system introduced in Chapter 3.

The goal of the algorithm here is to reduce the complexity in terms of number of vertices ($n_{\mathcal{V}}$) as much as possible while satisfying conditions (4.32) and (4.33) (as discussed in Section 4.4.3).

The double integrator is represented by the continuous-time linear system

$$\dot{x} = Ax + Bu \tag{4.34}$$

where $x \in \mathbb{R}^2$, $u \in \mathbb{R}$,

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \; B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{4.35}$$

The problem formulation includes a state constraint $-3 \leq x_2 \leq 3$ and an input constraint $-1 \leq u \leq 1$.

The state components x_1 and x_2 can represent, for example, the position and velocity, respectively, of a body having mass 1.

In Table 4.2 the discrete counterpart of system (4.34) is considered for several sampling times. For faster sampling times in general the polyhedral borders of the feasible sets approximate ellipsoidal shapes, which therefore imply complex $\mathcal{V}$-representations. Many of the vertices can be removed with a minimal loss in terms of volume. Moreover, faster sampling time means generally a more complex explicit MPC solution, which more likely may require the use of simpler feasible sets to use in approximate explicit MPC approaches. In the table, $n_{\mathcal{V}}$ indicates the number of vertices of the feasible set, $\tilde{n}_{\mathcal{V}}$ indicates the number of vertices of the approximate feasible set.

Figures 4.1, 4.2 and 4.3 graphically illustrate the idea for sampling time 0.3 and $N = 10$. As operating set we assume that the system has to operate in a range of positions $-10 \le x_1 \le 10$, for any feasible x_2 (Figure 4.1). Analogously chosen operating sets have been used for the results in Table 4.2.
Note that the feasible set, characterized by 24 vertices, is approximated by a less complex feasible set characterized by 10 vertices (Figure 4.3). The loss of volume introduced by the approximate feasible set is less than 3%.
To compute the set $\mathcal{X}_F^+$ in Figure 4.2, the explicit solution of the MPC is obtained and then each region comprising the feasible set is propagated one step forward.

Table 4.2: Complexity reduction by approximate feasible sets.

Samp. time	N=7	N=10
seconds	$n_{\mathcal{V}}$ - $\tilde{n}_{\mathcal{V}}$ - loss$\%$	$n_{\mathcal{V}}$ - $\tilde{n}_{\mathcal{V}}$ - loss$\%$
0.3	18 - 6 - 0.04$\%$	24 - 10 - 0.03$\%$
0.1	34 - 12 - 0.01$\%$	28 - 8 - 0.01$\%$
0.01	56 - 30 - 0.00$\%$	66 - 34 - 0.00$\%$

Extensive simulations have been carried out on several random systems for different state (n) and input (r) dimensions. The common MPC settings used are: horizon $N = 5, Q = I$ and $R = I$; state constraints $-20 * \mathbb{1} \le x \le 20 * \mathbb{1}$, input constraints $-\mathbb{1} \le u \le \mathbb{1}$.
Table 4.3 lists some of the results obtained in the simulations. As expected from the discussion in Section 4.4.4, in most of the cases the reduction in the number of vertices also led to a reduction in the number of half-spaces. However, a few cases where this did not happen are reported to illustrate that the upper bound theorem guarantees only that a reduction of vertices will not cause an extreme increase in the number of half-space.

Remark 7. In some cases the algorithmic implementation of the approach faced numerical errors, particularly with feasible sets of dimension higher than 3. Analo-

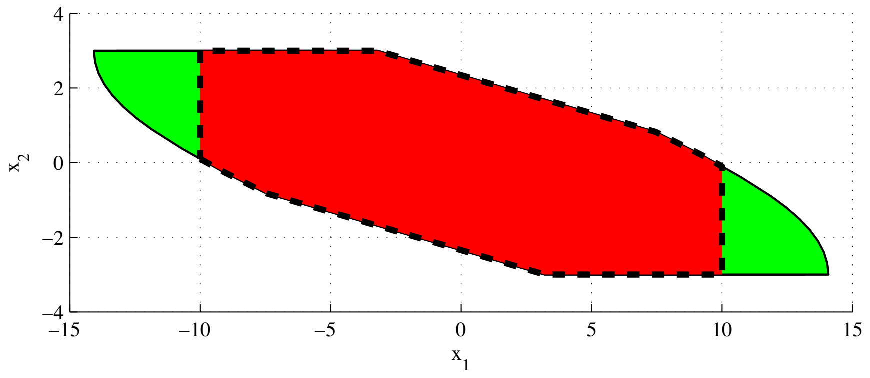


Figure 4.1: The largest polytope is the feasible set $\mathcal{X}_F$ (green). The set marked with the wide dashed line represents the operating set (red), contained in the feasible set.

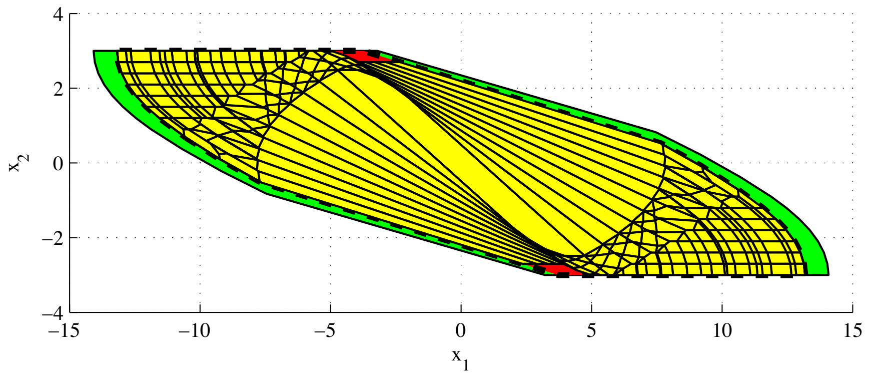


Figure 4.2: The largest polytope represents the feasible set $\mathcal{X}_F$ (green). The set marked with wide dashed line represents the set $\mathcal{X}_F^{(N-1)}$ (red and yellow), which corresponds with the feasible set for an horizon length $N-1$. The internal regions (in yellow) marked with thin lines comprise the next time-step feasible set $\mathcal{X}_F^+$. Note that $\mathcal{X}_F^+ \subset \mathcal{X}_F^{(N-1)} \subset \mathcal{X}_F$ and that $\mathcal{X}_F^+$ is non-convex.

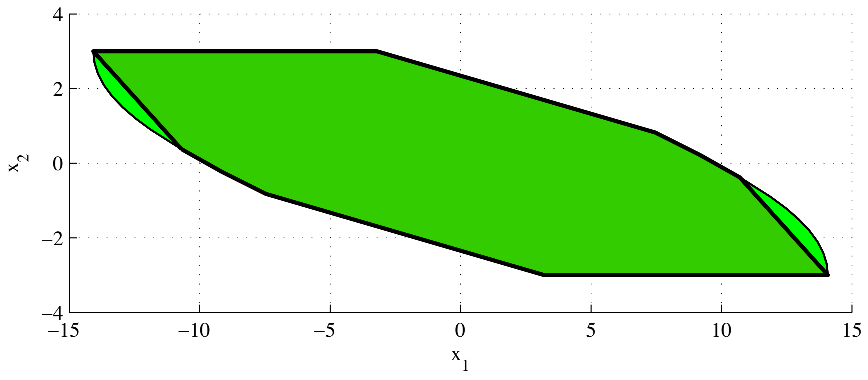
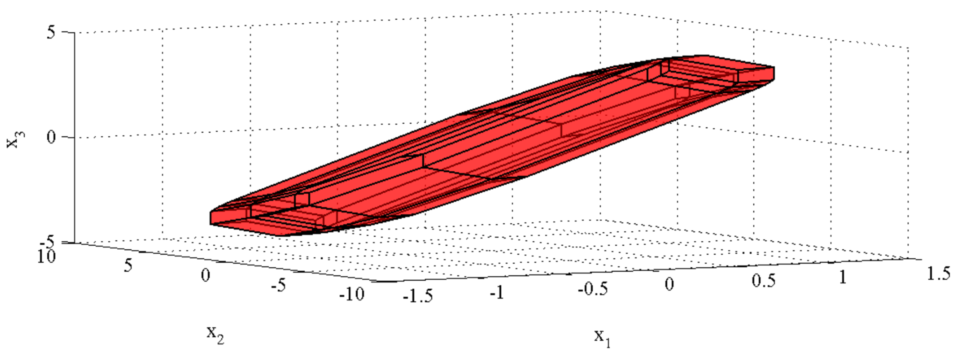


Figure 4.3: The largest polytope represents the feasible set $\mathcal{X}_F$. The internal set marked with wide solid line represents the reduced feasible set.

gously to the remark in Section 4.6.1, the explanation seems to lie in the dependence on the MPT routines implementing the basic operations on polytopes. It must also be said that, apart from the use of pointers to make computations more efficient, no particular emphasis has been put in coding an efficient implementation of the approach. The scope here was primarily to provide evidence of its effectiveness. Careful re-implementations of the routines for the basic polytope operations and re-implementation of the approach would reasonably remove most of the numerical issues and improve the computational performance (c.f. Section 4.5).

Table 4.3: Feasible set vs approximate feasible set for several random systems. The case marked with (*) is illustrated in Figure 4.4.

Rand. sys.		Feasible set	Approximate feasible set
(n,r)		$n_{\mathcal{V}} - n_{\mathcal{H}}$	$\tilde{n}_{\mathcal{V}} - \tilde{n}_{\mathcal{H}}$
$(3,1)$		$28 - 16$	$16 - 10$
		$58 - 56$	$35 - 46$
	(*)	$82 - 78$	$54 - 64$
$(3,2)$		$74 - 44$	$41 - 38$
		$64 - 34$	$25 - 28$
		$52 - 28$	$30 - 29$
$(4,1)$		$124 - 38$	$87 - 57$
		$116 - 58$	$92 - 56$
		$108 - 34$	$84 - 40$



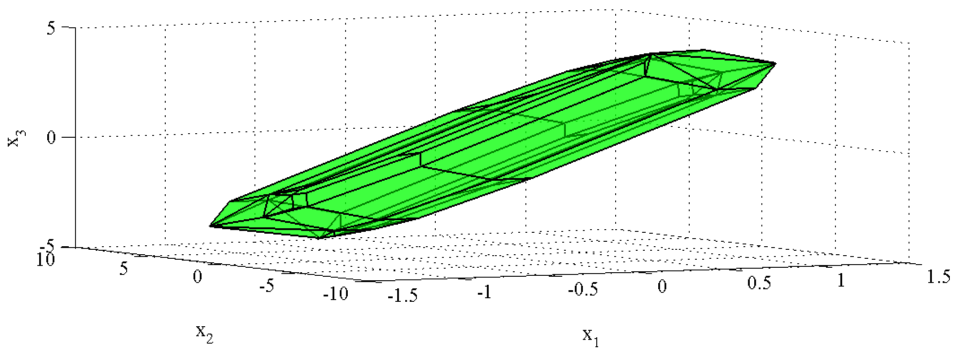


Figure 4.4: The upper graph shows the feasible set in the 3-dimensional state space for a system in Table 4.3. The graph below shows the corresponding approximate feasible set.

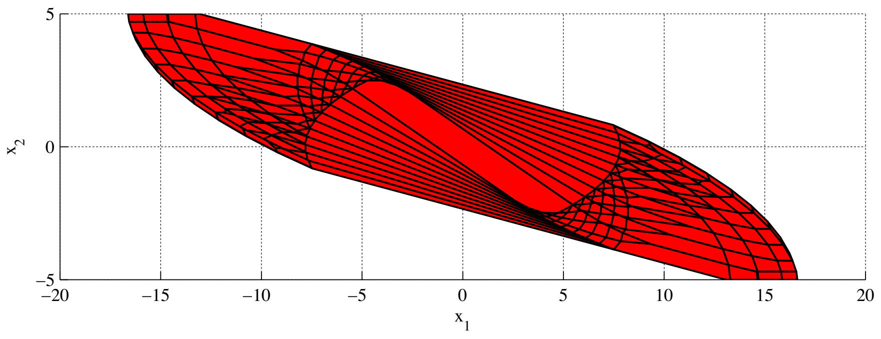
4.6.3 Application to Approximate Explicit MPC

Consider the approach for deriving a suboptimal explicit MPC solution discussed in Chapter 3. For this approach, the number of components of the resulting PWA control law depends on the complexity of the feasible set in terms of number of vertices.
The system is the double integrator (4.34) discretized with the sampling time 0.1. The MPC settings are: horizon $N = 10$, $Q = I$ and $R = I$; state constraints $-5 \leq x_2 \leq 5$, input constraints $-1 \leq u \leq 1$; tolerance set to 50%.
For simplicity, the operating set is assumed to be within the set $\mathcal{X}_F^{(N-1)}$ so that only the positive invariance condition (4.32) needs to be verified.
Figure 4.5 illustrates the feasible set and its partitions for the optimal explicit MPC (characterized by 359 regions) and for the suboptimal explicit MPC (characterized by 97 regions).
Figure 4.6 shows the reduced feasible set. It also illustrates that the use of the approximate feasible set allows a reduction in the total number of regions (77 regions). In fact, the regions needed to cope with the curve-like shapes of the feasible set borders are removed.

4.7 Discussion and Conclusions

The chapter has presented an alternative approach for computing feasible sets when MPC techniques are used. The proposed approach uses set relations instead of the conventional projection, which then unfolds to a procedure based on Minkowski sum and intersection routines. This proves to be computationally more efficient and algorithmically more robust than using projection routines, particularly when high dimensional polytopic sets are involved (i.e. for long prediction horizons, high dimensional state and/or input).
However, some numerical issue suggested the need of future work to improve the algorithmic robustness of the routines for the needed polytopic operations.

When the feasible set is characterized by a critical complexity of representation in terms of number of vertices, an approach to compute a simplified feasible set with a reduced number of vertices has been given. The approach is based on the introduction of certain conditions which extend existing approaches for the computation of polytope approximations, so that the approximating polytope maintains all the fundamental properties of the feasible set required for MPC applications like positive invariance and inclusion of the set of expected operating conditions.
Preserving the positive invariance property in the feasible set approximation is cru-



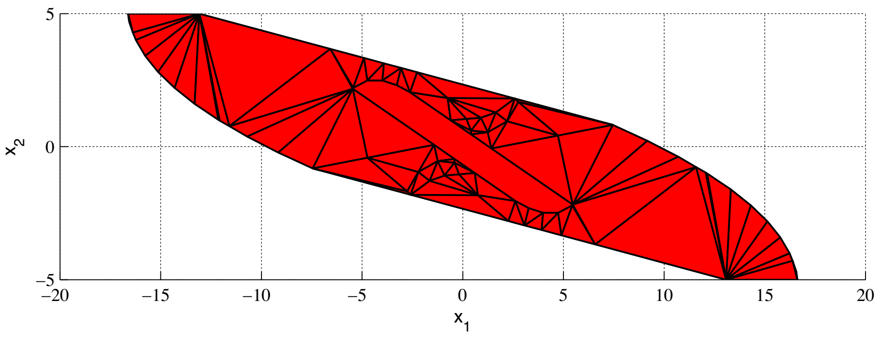


Figure 4.5: On top, the figure shows the the feasible set and its optimal partition for the optimal explicit MPC; number of regions: 359. The figure below shows the approximate explicit MPC generated by Algorithm 2 in Chapter 3 for a tolerance set to 50%; number of regions: 97.

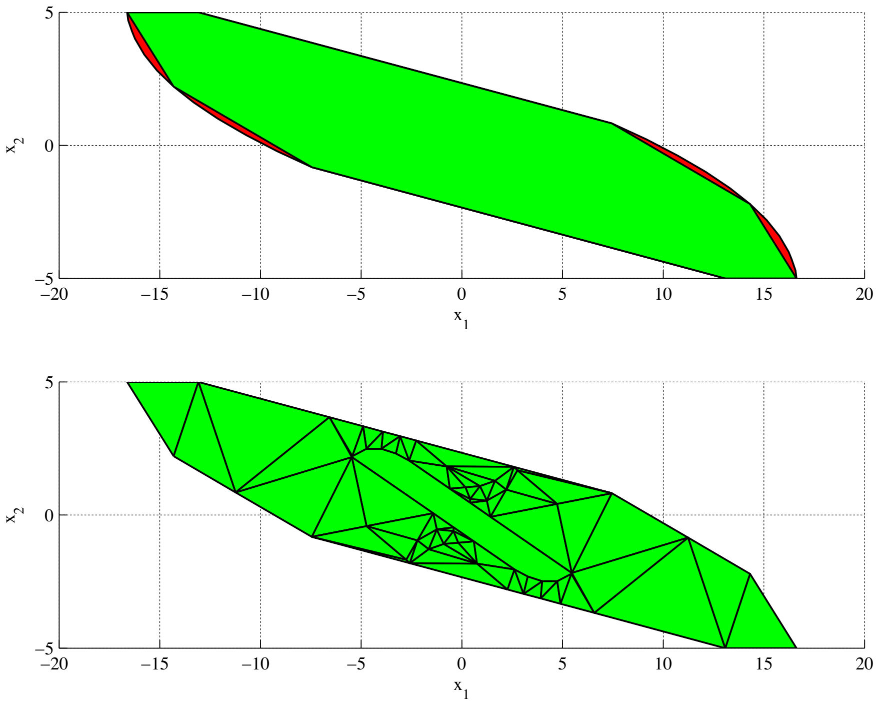


Figure 4.6: On top, the figure shows the feasible set (red and green) with $n_{\mathcal{V}} = 28$ and the approximate feasible set (green) with $\tilde{n}_{\mathcal{V}} = 8$. The figure below shows the approximate explicit MPC generated by Algorithm 2 in Chapter 3 for a tolerance set to 50% and using the approximate feasible set; number of regions: 77.

cial. This issue is inherently difficult to handle since it is concerned with the nonlinear dynamics of the closed-loop system. The proposed approach typically allows a considerable decrease in the $\mathcal{V}$-representation complexity by removing most of the vertices needed to deal with feasible set borders which approximate ellipsoids (according to the operating set considered). However, this approach does not allow to consider possible even simpler feasible set approximations which, while including the operating set, may have borders within $\mathcal{X}_F^{(N-1)}$. A potential future research direction could be to search for different approaches which would give more flexibility. One could for example look at solutions which use level surfaces of Lyapunov functions (Alessio et al. (2006)) to find different vertices than the original ones from the feasible set.
The conditions introduced constrain the goal of minimizing the loss of volume in the approximation. Finding suitable approximating polytopes characterized by the minimum loss of volume is a well known problem. Requiring that the approximation minimizes the loss of volume while satisfying conditions related to system dynamics remains challenge. Here the minimization of the loss of volume was not considered critical, since in the context of the present work the interest often is to preserve given crucial parts of the feasible set, which can be done via the operating set condition. In fact, the algorithm proposed tends to minimize the loss of volume in the sense that at each iteration the suitable vertex which results in the lowest loss of volume in the current approximating polytope is removed. Pointer structures were used to enhance the implementation efficiency, though it may be further improved by a careful re-implementation of the approach. Simulations proved the effectiveness of the results presented.

Chapter 5

Robust Feasibility for Constrained Linear Systems with PWA Controllers

Piecewise affine (PWA) feedback control laws represent an important class of controller for linear systems subject to linear constraints, with explicit Model Predictive Control approaches being probably the most popular techniques to obtain such control laws. These controllers are usually defined within a polyhedral set of initial states called the feasible set. In the presence of model mismatch, when the controller designed using the nominal model is applied to the real plant, the feasible set may lose its invariance property, resulting violation of constraints. Since the controller is only designed over the feasible set, there is also the technical problem that the control action is undefined if the state moves outside of the feasible set. This chapter proposes a tool to analyze how uncertainty in the model affects the piecewise affine control law computed using a nominal model. Given the linear system describing the plant and the piecewise affine control law, the algorithm that is presented considers a polytopic model uncertainty defined by the user and constructs the maximal robust feasible set, i.e. the largest subset of the feasible set which is guaranteed to be feasible for any model in the family of models described by the polytopic uncertainty.

5.1 Introduction

The concept of invariant sets has been shown to play an important role in the control and analysis of constrained systems (Blanchini (1999), Kerrigan and Maciejowski (2000), Gilbert and Tan (1991)). Given an autonomous dynamic system, a subset of the state space is said to be positively invariant if it has the property that, if it contains the system state at some time, then it will also contain it at all future times. The presence of constraints on the state variables defines an admissible set in the state space, i.e., the set of states that satisfies the constraints at the present time. Due to the system dynamics, in general, not all the trajectories originating from admissible initial states will remain in such a set. Conversely, for any initial condition which belongs to a positively invariant subset of the admissible domain, constraint violations are avoided at future times.
Such characterizations have relevant control applications. Consider the discrete-time linear time-invariant system

$$x\left(t+1\right)=Ax\left(t\right)+Bu\left(t\right) \tag{5.1}$$

$$y\left(t\right)=Cx\left(t\right), \tag{5.2}$$

and a linear state feedback control law that regulates the system to the origin

$$u\left(t\right)=Kx\left(t\right), \tag{5.3}$$

where $x \in \mathbb{R}^n$ is the state vector, $y \in \mathbb{R}^m$ is the output vector and $u \in \mathbb{R}^r$ is the input vector, $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times r}$, $C \in \mathbb{R}^{m \times n}$, K is a constant matrix gain. Suppose that it is required that the closed-loop system satisfies the output and input constraints

$$u_{min} \leq u\left(t\right) \leq u_{max}, \tag{5.4}$$

$$y_{min} \leq y\left(t\right) \leq y_{max}, \tag{5.5}$$

for all time instants $t \geq 0$, where y_{min}, y_{max} and u_{min}, u_{max} are constant vectors of suitable dimension. The closed-loop system represents an autonomous system, and the constraints can be easily rewritten as constraints on the state variables, giving the admissible domain in the state space. Then, starting from any initial condition inside a positively invariant subset of the admissible domain will guarantee convergence to the origin without violation of the constraints.
Among the families of positively invariant sets, the polyhedral sets are of particular importance because of their flexibility and the fact that they are often natural expressions of physical constraints. The analysis of feasible positively invariant sets for linear autonomous systems was considered in Gilbert and Tan (1991), where the authors provide a systematic way to construct polyhedral invariant sets.

The concept of invariant sets extends naturally when a control input is present: a set is said control invariant if, for any initial state in the set, it is possible to keep the trajectory inside the set by means of an admissible feedback control law. Invariant sets are central in Model Predictive Control (MPC), the predominant control approach for systems subject to constraints. When linear models and linear constraints are considered, the stability of the nominal closed-loop system can be guaranteed by imposing positively invariant terminal set constraints (Mayne et al. (2000)). The constrained optimization in the MPC problem also characterizes the relative maximal feasible control invariant set (*feasible set* for short), i.e. the largest set of initial conditions such that the objective of the control is obtained without violating the constraints.
Posing the MPC problem as a (multi-)parametric optimization problem, the controller can be given as an explicitly defined continuous PWA function of the state over the feasible set (cf. Chapter 2). Many solutions have also been proposed to obtain PWA controllers as approximations of the optimal explicit MPC controller when this is impractical (cf. Chapter 3). This explains the importance of PWA feedback state laws in the control of constrained linear systems. Indeed, in the following we will assume that the PWA controller considered is the result of some (approximate) explicit MPC approach, since this is probably the most common way to obtain such controllers.

Linear models always involve approximations since all real plants are, to some extent, nonlinear, time-varying and distributed (Ikonen and Najim (2002), van den Boom and Haverkamp (2000)). Thus, any controller obtained by model-based design has to deal with the inherent model uncertainty. Model errors can also be introduced when the available model is of prohibitive order for real-time control and model reduction techniques are adopted to obtain a suitable low order model (Hovland et al. (2008), Johansen (2003)). Naturally, the ultimate goal of the control is to meet the performance requirements when implemented in the real plant. In order to meet such a goal, the control law should guarantee acceptable performance not only for the nominal plant model but also for a family of models which includes, by assumption, the real plant.
A popular paradigm used to cope with model uncertainty is polytopic model uncertainty. Polytopic model uncertainty constitutes a flexible and powerful tool to describe families of models and therefore also model uncertainties, and has been studied for many years (Boyd et al. (1994), van den Boom and Haverkamp (2000)). Robustness to model uncertainties in the MPC context has attracted great attention in the literature (Mayne et al. (2000)). An exhaustive review is out of the scope of this chapter, it is instead interesting to focus on some relevant previous work. Polytopic uncertainties have been taken explicitly into consideration in the control design, resulting in robust MPC formulations where the constrained optimization

problem is modified to a min-max problem which minimizes the worst-case value of the cost function, where the worst-case is taken over the set of uncertain models (Kothare et al. (1996), Kouvaritakis et al. (2000), Cuzzola et al. (2002), Mayne et al. (2000)). The same min-max approach has also been considered in explicit MPC (de la Peña et al. (2004), Grieder et al. (2003), Cychowski et al. (2005)). However, the solutions obtained are in general rather complex and conservative. In Pluymers et al. (2005a) a simpler and less conservative approach was proposed. The nominal MPC formulation is used, and robustness is defined in terms of satisfaction of input and output constraints for all possible uncertainty realization. An explicit implementation based on this approach was proposed in Rossiter et al. (2005).

Polytopic uncertainties are also useful in performance analysis of nominal controller with respect to possible model uncertainties. The work in Pluymers et al. (2005b) considers linear systems controlled by linear feedback controllers and subject to linear state and input constraints, and proposes an algorithm for constructing the largest set of initial condition which is guaranteed to be positively invariant for all possible models in a given polytopic uncertainty set.

This chapter proposes a tool to analyze how uncertainty on the model affects the (approximate) explicit MPC solution computed using the nominal model. In fact, it has been shown that MPC approaches possess a remarkable level of inherent robustness, and stability and good performance are maintained for sufficiently small uncertainties (Nicolao et al. (1996), Mayne et al. (2000)). However, when constrains are present, it is also necessary to ensure that the uncertainty does not cause any violation of constraints. Given a nominal linear system describing the plant and a PWA feedback control law designed accordingly, the algorithm that is presented considers a polytopic model uncertainty defined by the user and constructs the *maximal robust feasible set*[1]. This is the largest subset of the nominal feasible set which is guaranteed to generate feasible state trajectories for any model in the family of models described by the polytopic uncertainty. Therefore, for any initial condition within the maximal robust feasible set, the closed-loop system is guaranteed to be feasibly stable.

This can be useful, for example, in the case of control systems for plants which are time-varying due to wear, and subject to state and input constraints. In this case, designing a controller which accounts explicitly for the model mismatch may be unnecessarily conservative, decreasing the performance. Instead, a control design based on the nominal model may represent a better choice, resorting to the intrinsic robustness of the nominal controller to deal with the slowly progressive plant variation. Then, the results here presented can be used to investigate whether the constraints may be violated over time.

[1]This notation may be in contrast with some work in the literature where a robust feasible set results from using a robust MPC design.

5.2 Basic Notions

5.2.1 Polytopic Uncertainty

Consider a linear system of the form (5.1). Model uncertainty can be expressed by saying that

$$[A|B] \in \mathcal{M}, \tag{5.6}$$

where $\mathcal{M}$ is a polytope in the parameter space defined by its vertices

$$\left\{ \left[A^{(1)}|B^{(1)} \right], ..., \left[A^{(L)}|B^{(L)} \right] \right\}, \tag{5.7}$$

L is the number of vertices, as

$$\mathcal{M} \triangleq \texttt{conv}\left(\left\{ \left[A^{(1)}|B^{(1)} \right], ..., \left[A^{(L)}|B^{(L)} \right] \right\} \right). \tag{5.8}$$

The function $\texttt{conv}\,()$ refers to the convex hull (cf. Chapter 2).
This is equivalent to say that there exist L non-negative coefficients λ_l, $l = 1, ..., L$, $\sum_{l=1}^{L} \lambda_l = 1$, such that

$$[A|B] = \sum_{l=1}^{L} \lambda_l \left[A^{(l)}|B^{(l)} \right]. \tag{5.9}$$

The case $L = 1$ corresponds to the case of no model uncertainty.
Polytopic model uncertainty is a flexible tool to describe uncertainties. Consider for example $A \in \mathbb{R}^{1\times 1}$ and $B = 0$. If the nominal value $A = a_n$ is known to describe the real value, a_r, with an accuracy ε: $a_n - \varepsilon \le a_r \le a_n + \varepsilon$, then $\mathcal{M} = \texttt{conv}\left(\{a_n - \varepsilon, a_n + \varepsilon\}\right)$.

5.2.2 Definitions

Consider the polyhedral convex sets $\mathcal{U} \subset \mathbb{R}^r$ and $\mathcal{Y} \subset \mathbb{R}^m$ given as

$$\mathcal{U} = \{u \in \mathbb{R}^r | D_{\mathcal{U}} u \le d_{\mathcal{U}}\} \tag{5.10}$$

$$\mathcal{Y} = \left\{y \in \mathbb{R}^m | D_{\mathcal{Y}} y \le d_{\mathcal{Y}}\right\}. \tag{5.11}$$

The input and output constraints are of the form

$$u(t) \in \mathcal{U} \tag{5.12}$$

$$y(t) \in \mathcal{Y}, \tag{5.13}$$

for all $t > 0$. We will assume also that the origin is an interior point of the sets $\mathcal{U}$ and $\mathcal{Y}$.
Note that the constraints (5.4, 5.5) are special cases of (5.12-5.13).

Let us now particularize the definitions of "controlled positive invariance" and "feasible controlled positive invariance" given in Chapter 2 for the case where a particular control law is considered.

Definition 21. (Feasible positive invariance) A positively invariant set $\mathbb{S}$ for a system of the form (5.1-5.2) in closed-loop with a particular feedback control law $u(t) = \Phi(x(t))$ is termed *feasible* with respect to constraints (5.12-5.13) if

$$\forall x(0) \in \mathbb{S} \ : \ u(t) \in \mathcal{U}, \ y(t) \in \mathcal{Y} \text{ for } t \geq 0 \tag{5.14}$$

Definition 22. (Robustly feasible positive invariance) Given a positively invariant set $\mathbb{S}$ for a system of the form (5.1-5.2) in closed-loop with a particular feedback control law, $u(t) = \Phi(x(t))$, feasible with respect to constraints (5.12-5.13), a subset $\mathbb{S}_R \subseteq \mathbb{S}$ is said to be *robustly feasible* for the family of dynamics in an uncertainty set of form (5.8) if

$$\forall x(0) \in \mathbb{S}_R \ : \ u(t) \in \mathcal{U}, \ y(t) \in \mathcal{Y} \text{ for } t \geq 0, \ \forall\, [A|B] \in \mathcal{M} \tag{5.15}$$

The set $\mathbb{S}_R$ is *maximal* if it also contains all the other robustly feasible sets.

Note that definition 22 implies that for all $x(0) \in \mathbb{S}_R \subseteq \mathbb{S}$, the state evolution $x(t)$, for all $t > 0$, is contained within $\mathbb{S}$ for any time invariant $[A|B] \in \mathcal{M}$.

5.3 Problem Formulation

Consider the problem of regulating to the origin a plant with a given nominal model of the form (5.1-5.2), such that constraints like (5.12-5.13) are satisfied. Assuming that the state is available for measurement, the regulation problem is solved by the finite horizon MPC

$$\min_{\mathbf{u} \triangleq [\mathbf{u_0}, \ldots, \mathbf{u_{N-1}}]} \left\{ J\left(\mathbf{u}, x\left(t\right)\right) = x_N^T P x_N + \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k \right\} \tag{5.16}$$

$$\begin{array}{lll} \text{s.t.} & x_0 = x(t), & \\ & x_{k+1} = Ax_k + Bu_k & \\ & y_k = Cx_k & k = 0, 1, ..., N \\ & y_k \in \mathcal{Y}, & k = 1, 2, ..., N \\ & u_k \in \mathcal{U}, & k = 0, 1, ..., N-1 \\ & x_N \in \Omega & \end{array} \tag{5.17}$$

where x_k denotes the predicted state vector at time $t+k$ obtained by applying the k first elements of the input sequence $\mathbf{u} \triangleq [u_0, ..., u_{N-1}]$; N is the prediction horizon; $Q \succeq 0$ (positive semidefinite) and $R \succ 0$ (positive definite) are symmetric matrices corresponding to weights on state and input; P is the terminal cost matrix and $x_N \in \Theta$ the terminal constraint, which are defined to guarantee stability (cf. Chapter 2). The matrix $P \succ 0$ is the solution of the algebraic Riccati equation resulting from the corresponding unconstrained LQR problem. The terminal set Ω is chosen to be feasible and positively invariant for the closed-loop system with this LQR.
The MPC will regulate the system to the origin for all the initial conditions contained in the feasible set

$$\mathcal{X}_F = \{x \in \mathbb{R}^n | \exists\, \mathbf{u} \text{ satisfying } (5.17)\}. \tag{5.18}$$

Note that $\mathcal{X}_F$ is a convex polyhedron due to the nature of the constraints (cf. Chapter 4). The feasible set is positively invariant with respect to the closed-loop system, i.e. for any initial state contained in the feasible set, the state evolution of the closed-loop system is also contained in the feasible set for all future times.
There are two ways to implement the constrained optimization problem (5.16)-(5.17). The first is to formulate the MPC as a quadratic program (QP) and solve it online at each sampling time. Only the first element of the optimal control sequence is applied to the system, and at the next time step, the computation is repeated starting from the new state and over a shifted horizon. The second way is to formulate the MPC as a multi-parametric QP (mp-QP) which can be solved offline. In this case, the optimal control is given as an explicitly defined continuous piecewise affine (PWA) function depending on the current state, and defined over $\mathcal{X}_F$. The online computation reduces to the simple evaluation of the piecewise affine function. Details about how to obtain the QP and the mp-QP formulations can be found in Chapter 2.
For the cases where the explicit optimal PWA controller is so complex as to be impractical, several approaches have been proposed to obtain an approximate continuous PWA controller (cf. Chapter 3).

The following considers a continuous PWA feedback control law

$$u(x) = L_j x + g_j,\ \forall\, x \in CR_j, \tag{5.19}$$

defined over the partition of the feasible set

$$\mathcal{X}_F = \bigcup_{j=1\dots n_r} CR_j, \tag{5.20}$$

where $\mathcal{P}_N = \{CR_1, ..., CR_{n_r}\}$ is the collection of polytopic disjoint regions into which $\mathcal{X}_F$ is partitioned.
The controller (5.19) can represent either the optimal MPC solution or any suitable approximation thereof.

Apart from the natural assumption that the uncertainty set $\mathcal{M}$ contains both the nominal model and the real model, we will make the following assumptions.

A1 Assume that there exists a subset $\mathcal{X}_{in} \subset \mathcal{X}_F$ containing the origin, in which the controller (5.19) asymptotically stabilizes the system (5.1) for any time-invariant $[A|B] \in \mathcal{M}$.

A2 Assume that for all initial states $x_0 \in \mathcal{X}_F$, for any $[A|B] \in \mathcal{M}$, if the closed-loop trajectory remains inside $\mathcal{X}_F$ for all future time, then it converges to the origin asymptotically.

Then, the problem tackled in this chapter is as follows:

- Given a controller of the form (5.19) computed for a nominal system of the form (5.1-5.2). Given an uncertainty set $\mathcal{M}$ of the form (5.6). Find the *maximal robust feasible set*, i.e. the set of initial conditions $\mathcal{X}_{FR} \subseteq \mathcal{X}_F$ such that for any possible time-invariant $[A|B] \in \mathcal{M}$, the closed-loop system remains feasible at all times.

Assumption A1 guarantees that $\mathcal{X}_{FR}$ will not be an empty set. This assumption is easy to confirm, e.g. the results in Pluymers et al. (2005b) can be used to find a robust positively invariant polyhedral set for the closed-loop system with the LQR.

Assumption A2 is needed to exclude that some system dynamics in the uncertainty set can lead to limit cycles or chaotic behavior in the feasible set. This assumption can be checked by means of a radially unbounded Lyapunov function (possibly dependent on the dynamics). Finding such a function may be difficult. A more immediate, but also conservative, approach is to check that in each region $CR_j \in$

$\mathcal{P}_N$ the following inequality holds for all the vertices $\left[A^{(i)}|B^{(i)}\right]$ of $\mathcal{M}$

$$\|x\| > \frac{\|B^{(i)}g_j\|}{1-\|A^{(i)}+B^{(i)}L_j\|} \;\; \forall x \in CR_j. \tag{5.21}$$

$\|\cdot\|$ is a norm or the corresponding induced (matrix) norm, depending on the argument. Due to convexity, (5.21) needs to be checked only on the vertices of each CR_j.

Proposition 3. *If condition (5.21) is satisfied for all regions $CR_j \in \mathcal{P}_N$, then the closed-loop evolutions satisfy*

$$\|x(t)\| > \|x(t+1)\| \; \forall x(t) \in \mathcal{X}_F \setminus \{0\} \tag{5.22}$$

for any system dynamics in the uncertainty set $\mathcal{M}$.

Proof. Inside each region $CR_j \in \mathcal{P}_N$, the closed-loop system is an affine system of the form

$$x(t+1) = \Phi_j x(t) + \varphi_j \tag{5.23}$$

where $\Phi_j = A + BL_j$ and $\varphi_j = Bg_j$. Then, using (5.23) in (5.22) the inequality can be written as

$$\|x(t)\| > \|\Phi_j x(t) + \varphi_j\| \tag{5.24}$$

The triangle inequality implies that

$$\|\Phi_j x(t) + \varphi_j\| \leq \|\Phi_j x(t)\| + \|\varphi_j\| \tag{5.25}$$

Thus, requiring that

$$\|x(t)\| > \|\Phi_j x(t)\| + \|\varphi_j\| \tag{5.26}$$

implies that also (5.24) is satisfied.
By a property of the induced norm it is

$$\|\Phi_j x(t)\| \leq \|\Phi_j\|\|x(t)\| \; \forall \, x(t) \tag{5.27}$$

Thus, if

$$\|x(t)\| > \|\Phi_j\|\|x(t)\| + \|\varphi_j\| \tag{5.28}$$

is satisfied then also (5.24) is satisfied. So it can be finally seen that if the inequality

$$\|x(t)\| > \frac{\|\varphi_j\|}{1-\|\Phi_j\|} \tag{5.29}$$

holds, then $\|x(t)\| > \|x(t+1)\|$. □

5.4 Algorithm

This approach follows a simple idea: remove from the feasible set all the initial states which, for any of the uncertain dynamics, lead to an infeasible closed-loop trajectory. Uncertain dynamics here means that the system is described by some time-invariant dynamics contained in the uncertainty set (5.8). Due to linearity, we need to consider only the vertices of the uncertainty set, which corresponds to considering the worst case dynamics. If the feasible set is robust for the worst case system dynamics, then it will be robust for all the system dynamics in the uncertainty set.
The complication with control laws of the form (5.19) is that the evolution of the closed-loop system changes depending on where the current state is in the feasible set.
We can now consider how the algorithm explores the feasible set by searching and removing all the initial states that may lead to infeasibility. For each vertex $\left[A^{(i)}|B^{(i)}\right]$ in the uncertainty set $\mathcal{M}$, the algorithm works in two phases.

In the first phase, each region $CR_j \in \mathcal{P}_{_N}$ forming the partition (5.20) is moved one time step forward, according to the control law associated with the region, to compute the next time-step region. The next time-step region is defined as follows.

Definition 23. (Successor set) The *successor set* of all states which can be reached in one time step from R_j, given system dynamics $[A|B]$, is defined as

$$\texttt{succ}\left(CR_j, [A|B]\right) = \left\{x^+ \in \mathbb{R}^n | x^+ = \left(A + BL_j\right)x + Bg_j,\ x \in CR_j\right\}. \quad (5.30)$$

Remark 8. The successor region can be computed simply by applying the control at the vertices of the region and taking the convex hull of the next time-step vertices.

This phase allows the identification of all the (sub)regions in $\mathcal{X}_F$ that in one time step would lead to infeasibility (Figure 5.1). It allows also the formation of a *map of reachability*, i.e. for each region $CR_j \in \mathcal{P}_{_N}$ to identify all the regions in $\mathcal{P}_{_N}$ containing states from which it is possible to reach the current region in one time step.

Certainly, all the states in the (sub)regions that in one time step would lead to infeasibility have to be removed from the feasible set. However, this is clearly not enough, also all the initial states whose closed-loop trajectory moves through these

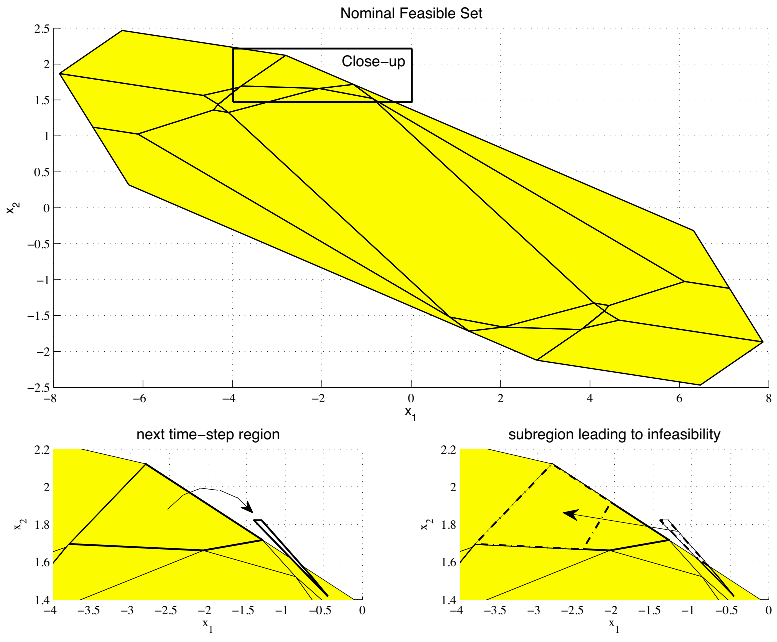


Figure 5.1: An example of nominal feasible set partitioned in regions CR_j. In the close-up on the left, the next time-step region is computed. In the close-up on the right the subregion of the feasible set leading to infeasibility is identified.

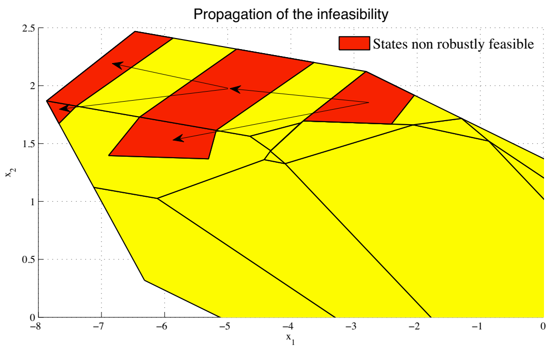


Figure 5.2: The subregion not robustly feasible identified in Figure 5.1 is propagated backwards in the feasible set, finding all the initial states which would lead to infeasibility.

(sub)regions need to be removed. This is done in the second phase of the algorithm, with a mechanism of propagation based on the following definition.

Definition 24. (Predecessor set) Given a region $S \subseteq CR_j \in \mathcal{P}_N$, a region $CR_k \in \mathcal{P}_N$ (CR_j and CR_k may coincide) and system dynamics $[A|B]$, all the states in CR_k for which the next time-step state is in S define the set (the “predecessor" states)

$$\texttt{pred}\left(S, CR_k, [A|B]\right) = \left\{x \in CR_k |\left(A + BL_k\right) x + Bg_k \in S\right\}. \tag{5.31}$$

Remark 9. S and CR_k can be represented as

$$S = \left\{x \in \mathbb{R}^n | D_S x \leq d_S\right\}, \; CR_k = \left\{x \in \mathbb{R}^n | D_{CR_k} x \leq d_{CR_k}\right\}. \tag{5.32}$$

Then, we can compute the predecessor set as the intersection of a finite number of half-spaces

$$\texttt{pred}\left(S, CR_k, [A|B]\right) = \left\{x \in \mathbb{R}^n | D_{\texttt{pred}} x \leq d_{\texttt{pred}}\right\},$$

where

$$D_{\texttt{pred}} = \left[\begin{array}{c} D_{CR_k} \\ D_S\left(A + BL_k\right)\end{array}\right], \; d_{\texttt{pred}} = \left[\begin{array}{c} d_{CR_k} \\ d_S - D_S B g_k\end{array}\right],$$

Using definition 24, all the infeasible (sub)regions in $\mathcal{X}_F$ identified during the first phase of the algorithm are propagated backwards in the feasible set, according to the map of reachability (Figure 5.2).

The procedure based on these two phases can be formalized as in Algorithm 4. Initially the maximal robust feasible set is initialized as the nominal feasible set. Then, for each vertex of the polytopic uncertainty set (for-loop $2-21$) the two phases (phase A: for-loop $3-7$; phase B: for-loop $8-20$) are iterated in sequence.

In general, $\mathcal{X}_{FR}$ is not robustly positively invariant. The set $\mathcal{X}_{FR}$ has the property of containing all and only the states in $\mathcal{X}_F$ which, when used as initial conditions, are guaranteed to have feasible closed-loop trajectories for any possible time-invariant dynamics in the uncertainty set. An initial state x which is not in $\mathcal{X}_{FR}$ does not possess a feasible closed-loop trajectory for all the possible system dynamics. However, this does not mean that x cannot be part of the feasible closed-loop trajectory starting from some state in $\mathcal{X}_{FR}$. (This is discussed further in Section 5.5).
On the other hand, requiring the positive invariance property of $\mathcal{X}_{FR}$ would have been too conservative and unnecessary. In fact, it follows that any set with guaranteed positive invariance despite model uncertainty is a subset of $\mathcal{X}_{FR}$. This would unnecessarily limit the possible initial conditions, since we are only interested in guaranteeing that any closed-loop trajectory stays in the feasible set $\mathcal{X}_F$ despite model uncertainty. Moreover, the algorithm for constructing such a positively invariant set would be much more computationally complex than that presented here. As can also be seen from the numerical examples in Section 5.5, in general $\mathcal{X}_{FR}$ is not a convex set, though it can be expressed as a finite union of polytopes. This is expected since the piecewise affine control law is a nonlinear controller.

Correctness and convergence of the algorithm are proven by the following theorems.

Theorem 10. *The robust feasible set $\mathcal{X}_{FR} \subseteq \mathcal{X}_F$ contains all and only the initial states such that, for any $[A|B] \in \mathcal{M}$, the closed-loop trajectory is feasible.*

Proof. To prove the theorem, we show first that if a state $x \in \mathcal{X}_F$ has a closed-loop trajectory that moves outside the feasible set for some $[A|B] \in \mathcal{M}$, then $x \notin \mathcal{X}_{FR}$. Since $[A|B]$ are inside the polytopic uncertainty set, they can be expressed as a convex linear combination of the vertices of $\mathcal{M}$ as in (5.9). Thus, there is at least a vertex $\left[A^{(i)}|B^{(i)}\right]$, $i \in \{1, ..., L\}$, such that, when used as system dynamics, causes the trajectory starting from x to exit the feasible set, which means that x cannot be in $\mathcal{X}_{FR}$ because it is removed by Algorithm 4 during iteration i of the for-loop at step 2.
It remains to prove that if a state $x \in \mathcal{X}_F$ has feasible closed-loop trajectories for all

Algorithm 4: Robust feasible set

```
Input: The nominal feasible set $\mathcal{X}_F$. The nominal PWA controller
       and the corresponding feasible set partition $\mathcal{P}_{_N}$. The
       uncertainty set $\mathcal{M}$.
Output: The maximal robust feasible set $\mathcal{X}_{FR} \subseteq \mathcal{X}_F$.
1  Initialize the robust feasible set as $\mathcal{X}_{FR} = \mathcal{X}_F$;
2  foreach $\left[A^{(i)}, B^{(i)}\right]$ of $\mathcal{M}$ do
3      foreach $CR_j \in \mathcal{P}_{_N}$ do
4          compute $S_{i,j} = \mathtt{succ}\left(R_j \cap \mathcal{X}_{FR}, \left[A^{(i)}|B^{(i)}\right]\right)$, the
           successor region for the remaining points in each region
           $CR_j$ of the $n_r$ such regions comprising $\mathcal{X}_F$;
5          define $Z_{i,j} = S_{i,j} - \mathcal{X}_{FR} \cap S_{i,j}$ and the union of all these
           sets $\mathcal{Z}_i = \bigcup_j Z_{i,j}$, which represents the set of infeasible
           states reachable in one step from any point in $\mathcal{X}_{FR}$ for this
           $[A^{(i)}|B^{(i)}]$;
6          build the function $\mathtt{rch}_i\left(CR_j\right)$, that gives all the regions
           containing states from which it is possible to reach $CR_j$ in
           one time step;
7      end
8      foreach $CR_r \in \mathcal{P}_{_N}$ do
9          compute $P_{i,r} = \mathtt{pred}\left(\mathcal{Z}_i, CR_r, \left[A^{(i)}|B^{(i)}\right]\right) \cap \mathcal{X}_{FR}$;
10         define $\mathcal{P}_i = \bigcup_r P_{i,r}$, the admissible predecessor set of $\mathcal{Z}_i$;
11         define $\mathcal{P}_{_N}^{rch} = \mathtt{rch}_i\left(\mathcal{P}_i\right)$ the set of all the regions containing
           states which in one time step can reach $\mathcal{P}_i$;
12         replace $\mathcal{X}_{FR} = \mathcal{X}_{FR} - \mathcal{P}_i$
13         repeat
14             foreach $CR_k \in \mathcal{P}_{_N}^{rch}$ do
15                 compute $P_{i,k} = \mathtt{pred}\left(\mathcal{P}_i, R_k, \left[A^{(i)}|B^{(i)}\right]\right) \cap \mathcal{X}_{FR}$;
16                 replace $\mathcal{P}_i = \bigcup_k P_{i,k}$ and $P_n^{rch} = \mathtt{rch}_i\left(\mathcal{P}_i\right)$;
17                 replace $\mathcal{X}_{FR} = \mathcal{X}_{FR} - \mathcal{P}_i$.
18             end
19         until $\mathcal{P}_i = \emptyset$ ;
20     end
21 end
```

$[A|B] \in \mathcal{M}$, then $x \in \mathcal{X}_{FR}$. Suppose by contradiction that $x \notin \mathcal{X}_{FR}$. Then there exist some vertex of $\mathcal{M}$ such that the closed-loop trajectory exits the feasible set, which contradicts the assumption that the closed-loop trajectory is feasible for all the dynamics in $\mathcal{M}$. Thus $x \in \mathcal{X}_{RF}$. □

Theorem 11. *Given the assumptions in Section 5.3 hold, the algorithm will terminate in a finite number of iterations providing a non-empty robust feasible set* $\mathcal{X}_{FR} \subseteq \mathcal{X}_F$.

Proof. The algorithm iterates the two phases A (for-loop $3-7$) and B (for-loop $8-20$) L times, where L is a finite number. Thus, we have to prove that phases A and B execute in finite time. Since $\mathcal{X}_F$ is assumed partitioned into a finite number of polytopes, it is immediate to see that phase A is executed in finite time, and that at each iteration the set of infeasible states $\mathcal{Z}_i$ is described as the union of a finite number of polytopic regions. During phase B, $\mathcal{Z}_i$ is propagated backwards in $\mathcal{X}_F$ according to definition 24. $\mathcal{P}_i$ is initialized as the admissible predecessor set of $\mathcal{Z}_i$, and then iteratively updated in the repeat-until loop at step 13. Since $\mathcal{Z}_i$ is the union of a finite number of polytopes, $\mathcal{P}_i$ will also have this property for all the iterations. At each iteration, the states comprising $\mathcal{P}_i$ are removed from the current $\mathcal{X}_{FR}$, and once removed they are not considered again in the future iterations. Thus, since $\mathcal{X}_{FR}$ is bounded, eventually $\mathcal{P}_i$ will be an empty set comporting the termination of phase B.
Assumption A1 guarantees that there exist a non-empty region, containing the origin, that will never be in $\mathcal{Z}_i$, thus $\mathcal{X}_{FR}$ will not be empty, and since at all iterations $\mathcal{P}_i$ is the union of a finite number of polytopes, $\mathcal{X}_{FR}$ will be represented as union of polytopic regions. □

Assumption A2 guarantees that for any initial state $x \in \mathcal{X}_{FR}$, for any time-invariant $[A|B] \in \mathcal{M}$, the closed-loop system is (feasibly) asymptotically stable.

5.5 Numerical Illustrations

This section provides examples in order to illustrate the results presented in the previous sections. Here an example is also used to discuss how the presented analysis approach can be related to existing robust control design approaches.

5.5.1 Robust Feasibility for Optimal Explicit MPC

Consider the double integrator system with input and state constraints, already introduced in the previous chapters. As mentioned, the model of the double integrator is one of the most important in control applications, representing single-degree-of-freedom translational or rotational motion. Thus it can be used to model for instance low-friction, free rigid-body motion, such as single-axis spacecraft rotation and rotary crane motion (Rao and Bernstein (2001)).

The double integrator is given by the continuous-time system

$$\dot{x} = Ax + Bu \tag{5.33}$$

$$y = Cx \tag{5.34}$$

where $x \equiv y \in \mathbb{R}^2$, $u \in \mathbb{R}$,

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix},\ B = \begin{bmatrix} 0 \\ 1/m \end{bmatrix},\ C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{5.35}$$

The state components x_1 and x_2 can represent for instance the position and velocity, respectively, of a body having mass m. Considering a mass $m = 1$, and discretizing with sampling time 0.3 we obtain the following discrete-time double integrator system matrices

$$A = \begin{bmatrix} 1 & 0.3 \\ 0 & 1 \end{bmatrix},\ B = \begin{bmatrix} 0.04 \\ 0.3 \end{bmatrix},\ C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{5.36}$$

The system is subject to the input constraints $-1 \leq u \leq 1$, and to the velocity constraints $-3 \leq x_2 \leq 3$.
We consider the uncertainty set $\mathcal{M}$ defined by the following vertices

$$A^{(1)} = \begin{bmatrix} 1 & 0.3 \\ 0 & 1 \end{bmatrix},\ B^{(1)} = \begin{bmatrix} 0.06 \\ 0.37 \end{bmatrix}, \tag{5.37}$$

$$A^{(2)} = \begin{bmatrix} 1 & 0.3 \\ 0 & 1 \end{bmatrix},\ B^{(2)} = \begin{bmatrix} 0.04 \\ 0.25 \end{bmatrix}. \tag{5.38}$$

which correspond to the mass being known with an uncertainty of $\varepsilon = 0.2$, i.e. the real mass value is $m = 1 \pm \varepsilon$.

Consider a PWA state feedback controller which represents the optimal solution of the MPC problem (cf. Chapter 2). The weight matrices are chosen as $Q = I$, $R = 1$

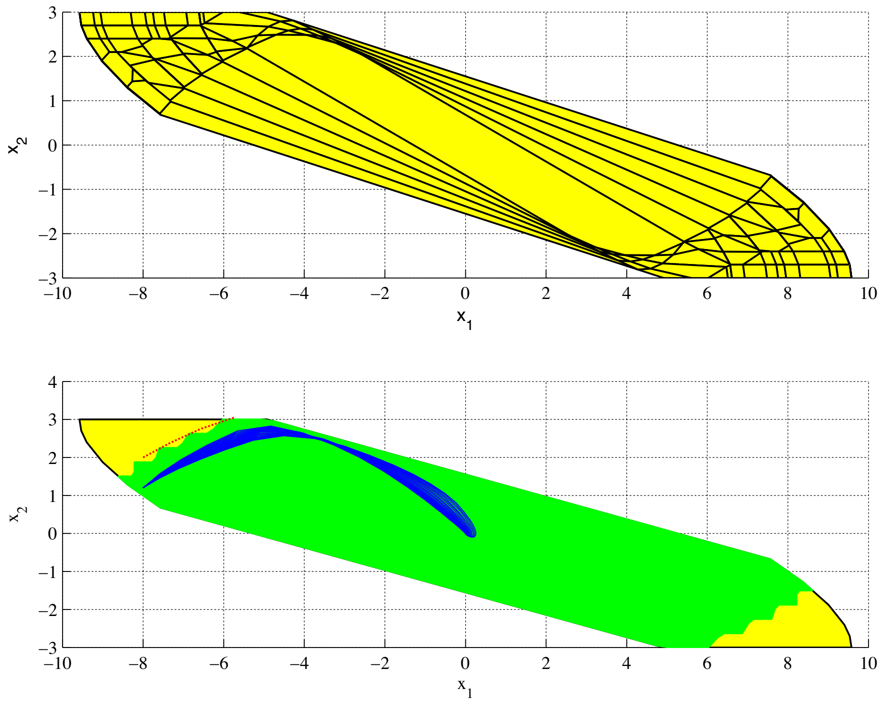


Figure 5.3: The upper graph shows the nominal feasible set and its partition for the optimal explicit MPC. The graph below shows the maximal robust feasible set within the feasible set. The feasible state trajectories for different system dynamics in the uncertainty set (blue solid lines) all start from initial position/speed $x_0 = [-8\ 1.2]^T$. The infeasible state trajectory (red dot line) starts from initial position/speed $x_0 = [-8\ 2]^T$ and is given by the system dynamics $\left[A^{(1)}|B^{(1)}\right]$.

and the horizon is $N = 5$.

Figure 5.3 shows the nominal feasible set, partitioned into 161 regions, and the portion of the nominal feasible set which is robustly feasible for the uncertainty considered. An initial state in the maximal robust feasible set is shown to generate feasible trajectories for different system dynamics within the uncertainty set. Contrarily, an initial state not in the maximal robust feasible set is shown to originate an infeasible trajectory: when the trajectory exits the feasible set, the control input is undefined.

Remark 10. Some of the feasible trajectories originating inside $\mathcal{X}_{FR}$ may contain states which are not in the set $\mathcal{X}_{FR}$ (but still in $\mathcal{X}_F$). This at first may seem nonsense, but it is perfectly reasonable if one considers that the real system is assumed uncertain but still time invariant: a state $\tilde{x} \notin \mathcal{X}_{FR}$ belonging to the closed-loop trajectory starting from $x \in \mathcal{X}_{FR}$ for certain system dynamics $\left[\tilde{A}|\tilde{B}\right]$ means that $\tilde{x}$ is a robustly feasible initial condition for a part of the uncertainty set including $\left[\tilde{A}|\tilde{B}\right]$, but this is not true for all the possible system dynamics and thus $\tilde{x}$ cannot be included in the set of allowed initial condition $\mathcal{X}_{FR}$.

5.5.2 Robust Feasibility for Approximate Explicit MPC

Consider the same double integrator system and polytopic uncertainty of the previous section, and a PWA state feedback controller which represents the approximate solution of the MPC problem computed according to the results in Chapter 3. The weight matrices are $Q = I$, $R = 1$ and the horizon is $N = 5$. For this simple example no extra vertices have been introduced to reduce the approximation error in the cost function, in fact both stability and good performance can be easily proven by post processing the PWA controller.
Figure 5.4 presents the feasible set with its partition into 41 regions and the maximal robust feasible set. As can be noted from the close-up, only a minimal part of the nominal feasible set is removed, almost the entire feasible set remains feasible under the uncertainty considered.
It is interesting to note from the simulations that for the case of the double integrator, the closed-loop system with the approximate explicit MPC is characterized by more robust feasibility to model uncertainty than the closed-loop system with the optimal MPC. This can also be seen from Figure 5.5, where the mass uncertainty $\varepsilon = 0.5$ is considered.

5.5.3 Relation to Existing Robust MPC Approaches

The approach proposed in this chapter represents a tool to analyze the feasibility robustness of nominal explicit MPC approaches (or in general, PWA feedback control laws) with respect to model uncertainty. This section discusses how this relates to a robust MPC design instead, illustrating it by a simple example.
The robust MPC design considered is the one presented in Pluymers et al. (2005a) (Rossiter et al. (2005)), which is based on a nominal MPC formulation where robustness is defined in terms of satisfaction of input and output constraints for all possible uncertainty realization. Given the connection with the nominal MPC design, it is reasonable to believe that this approach represents a better comparison than other robust MPC approaches based on min-max optimization problems.
The robust MPC can be summarized as follows. At each time step, the algorithm

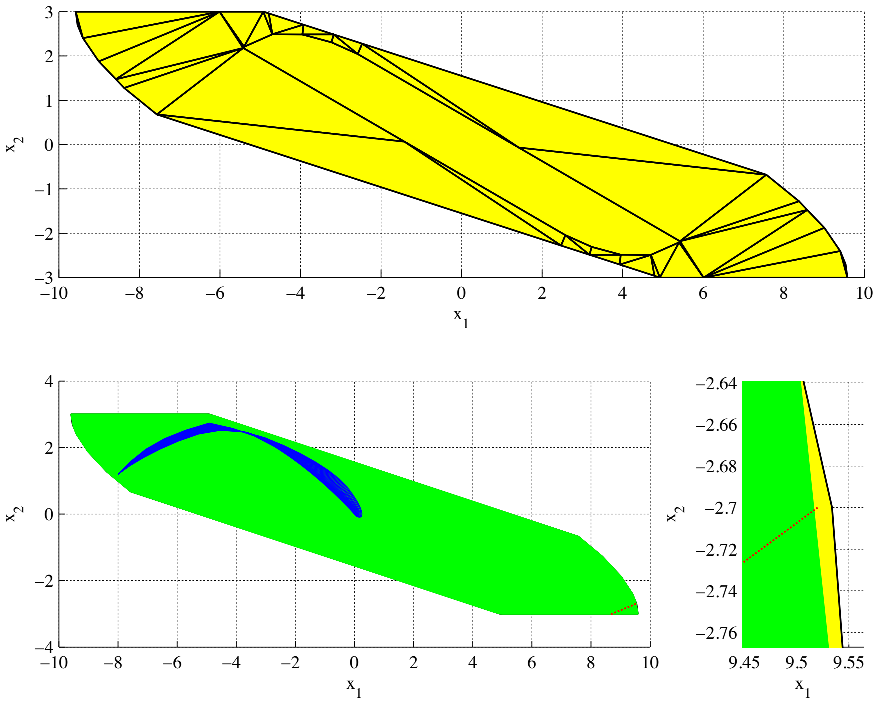


Figure 5.4: The upper graph shows the nominal feasible set and its partition for the approximate explicit MPC. The graph below shows the maximal robust feasible set within the nominal feasible set (emphasized in the close-up) for a mass uncertainty $\varepsilon = 0.2$. The feasible state trajectories for different system dynamics in the uncertainty set (blue solid lines) all start from initial position/speed $x_0 = [-8\ 1.2]^T$. The infeasible state trajectory (red dot line) starts from initial position/speed $x_0 = [9.52\ -2.7]^T$ and is given by the system dynamics $\left[A^{(1)}|B^{(1)}\right]$.

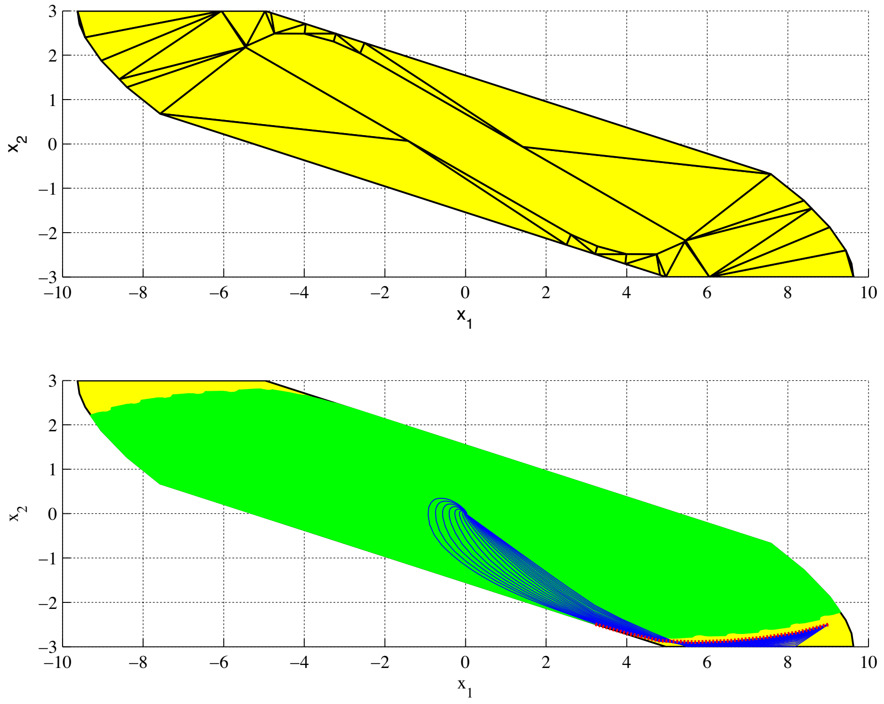


Figure 5.5: The upper graph shows the nominal feasible set and its partition for the approximate explicit MPC. The graph below shows the maximal robust feasible set within the nominal feasible set for a mass uncertainty $\varepsilon = 0.5$. The state trajectories for different system dynamics in the uncertainty set all start from initial position/speed $x_0 = [9 \ -2.5]^T$. Since x_0 is outside the robust feasible set, only for some of the system dynamics the trajectories are feasible (blue solid lines). There are system dynamics in the uncertainty set which lead to infeasible trajectories (red dot line).

minimizes a cost function like (5.16), where the nominal model is used for the future predictions along the horizon. The minimization is subject to constraints like (5.12-5.13) which, for robust constraints handling, are applied to all possible predictions according to the following k-step ahead prediction

$$x_k = \prod_{i=0}^{k-1} A_i x_0 + \sum_{j=0}^{k-1} \prod_{l=j+1}^{k-1} A_l B_j u_j \tag{5.39}$$

where $[A_i, B_i] \in \mathcal{M}$. A terminal constraint is imposed, where the (robust) terminal set is chosen as the largest set of initial condition which is guaranteed to be positively invariant for all possible models in $\mathcal{M}$, assuming the nominal LQR as controller (Pluymers et al. (2005b)). The resulting optimization problem remarkably remains a QP, even if, with respect to the QP resulting from the nominal MPC, more complexity in terms of number of constraints is needed in order to achieve robustness. A multi-parametric QP solution to this robust MPC is proposed in Rossiter et al. (2005). For more details the reader is referred to Pluymers et al. (2005a) and Rossiter et al. (2005).

Note that this robust MPC design is able to deal with linear parameter varying (LPV) systems, while the approach presented here considers uncertain linear parameter invariant systems.

Consider the simple example used in Pluymers et al. (2005a) which has polytopic uncertainty set defined by

$$A^{(1)} = \begin{bmatrix} 1 & 0.1 \\ 0 & 1 \end{bmatrix}, \; B^{(1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{5.40}$$

$$A^{(2)} = \begin{bmatrix} 1 & 0.2 \\ 0 & 1 \end{bmatrix}, \; B^{(2)} = \begin{bmatrix} 0 \\ 1.5 \end{bmatrix}. \tag{5.41}$$

and the nominal model defined as

$$A = \frac{1}{2}(A^{(1)} + A^{(2)}), \;\; B = \frac{1}{2}(B^{(1)} + B^{(2)}), \;\; C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{5.42}$$

The system is subject to the input constraint $-1 \leq u \leq 1$, and to the state constraints $[-10 \;\; -10]^T \leq x \leq [10 \;\; 10]^T$. For this system, the robust MPC and the nominal MPC are formulated both with weight matrices chosen as

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 0.01 \end{bmatrix}, \;\; R = 3. \tag{5.43}$$

and horizon $N = 3$.

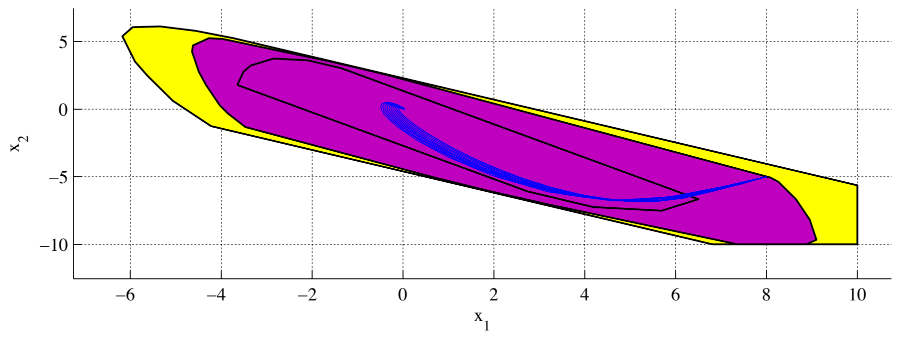
Figure 5.6 illustrates the feasible set resulting from the robust MPC. The same figure also shows the portion of the nominal feasible set which is robustly feasible with

the nominal MPC. Both robust and nominal MPC give the same performance, as it can be qualitatively seen from the closed-loop trajectories obtained for the same set of different time-invarying dynamics.
It is not hard to identify regions of initial states for which the nominal MPC would not be sufficient, while instead the robust MPC would be. However, it is also immediate to identify considerably larger regions of initial states which would be satisfactorily controlled by the nominal MPC and which are instead excluded by the feasible set with the robust MPC. Then, assuming that the set of initial conditions of interest is within the maximal robust feasible set from the nominal MPC, the analysis method presented in this chapter can be used to decide that the nominal controller is enough and therefore there is no need for the supplementary complexity associated with the robust control design. Of course, this does not exclude a number of cases where the robust design is instead necessary.

The analysis tool presented in this chapter may be useful, for example, in the practical case of a crane which has to move objects whose weight may be within a given range, satisfying constraints on position and speed. Reasonably, the parameters of the crane model can be expected to change for each possible weight (cf. Section 5.5.1). However, once the object has been fixed, from the point of view of the controller the model remains time invariant for the whole operation (until a new object is considered). In this case, a controller design based on a nominal model (for example one which considers the average weight) may be considered satisfactory, after the associated maximal robust feasible set has guaranteed that constraints will not be violated for any possible weight.

5.6 Conclusions

This chapter has proposed a tool for analyzing how uncertainty in the real plant affects the piecewise affine feedback law computed using the nominal model, thereby providing the maximal subset of the state space which contains safe initial conditions under the model uncertainty considered. This is a fundamental step towards any successful practical implementation of the controller.
The maximal robust feasible set thus obtained is, in general, non-convex. It is not required to be robustly positive invariant, and is computed in finite time. Moreover, any subset, and thus any convex subset, still preserves the property of being robustly feasible.
These results should not be seen as a competitor to robust MPC design. They are instead a tool to decide whether a nominal design can be used without resorting to a more complex robust design. On the other hand, the results can also be seen as an



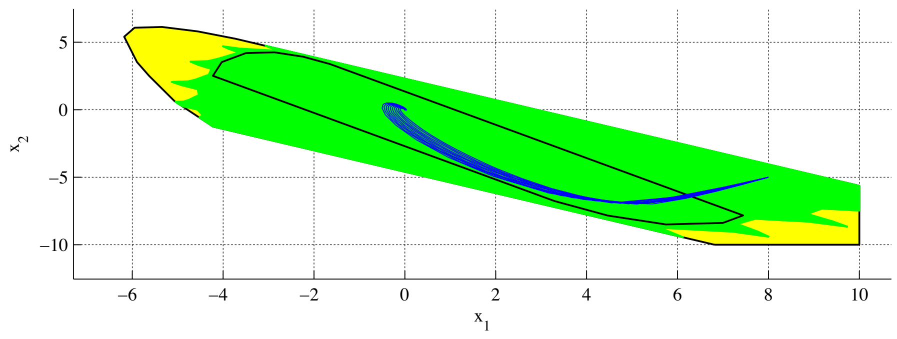


Figure 5.6: The upper graph shows the feasible set for the nominal MPC (yellow) and the feasible set for the robust MPC (magenta) with depicted in its interior the robust terminal set. The graph below shows the feasible set for the nominal MPC (yellow), the maximal robust feasible set (green) and the nominal terminal set. In both graphs, several state trajectories are plotted starting all from the initial state $x_0 = [8\ \ -5]^T$, for the same different system dynamics.

enabling technology for several future approaches to the problem of enhancing the robustness of (approximate) explicit MPC solutions towards model uncertainty. If the maximal robust feasible set does not cover the portion of state space of interest, the next step could be to consider just the regions that do not satisfy the robust feasibility condition, and search for suitable controllers for those regions. One approach could be to define and solve a new explicit MPC problem for each infeasible region, with proper constraints ensuring robust feasibility, and the maximal robust feasible set as the new terminal set.
Assumption A2 is needed to exclude the possibility of limit cycles or chaotic behavior of the uncertain system in closed-loop with the controller, originally designed for the nominal system. The assumption is easy to check, but rather conservative. Future research can be directed to reduce this conservativeness.
An interesting future work would be the extension to LPV systems. This could be achieved propagating back the infeasible regions from the phase A of the algorithm for all the possible uncertain realizations in the polytopic uncertainty set. This however would reasonably result in heavier computational loads.
The inclusion of robustness with respect to disturbances represents another future work of interest.

Chapter 6

Conclusions and Recommendation for Further Work

This thesis has been concerned with explicit MPC solutions. The geometric nature of the problem has been exploited to propose approaches for dealing with important issues entailed with this MPC implementation strategy.
This chapter briefly recalls the initial motivations, provides a summary of the results achieved and concludes with possible directions for future work.

6.1 Final Remarks and Possible Future Research Directions

Different problematics arising in explicit MPC have been considered in the thesis. Explicit MPC formulations represent a promising approach to extend the scope of applicability of MPC approaches to situations where the online computations required for the conventional MPC implementations are prohibitive for technical or cost reasons. Formulating the MPC problem as a multi-parametric programming problem, the undesired online optimization effort is moved offline, and the optimal control is given as an explicitly defined PWA function with dependence on the current state. The domain of the PWA function is the feasible set, which is partitioned into convex regions. The online computation reduces to the simple evaluation of the PWA function. Thus, explicit solutions enable the implementation of MPC approaches using simple hardware with fast sampling rates, and provide insight of the controller behavior that is useful for performance analysis.

However, explicit MPC implementations may still be prohibitively costly for large optimization problems, in fact the offline computation effort needed to solve the multi-parametric optimization problem increases quickly with the dimensions of state and input, the number of constraints involved in the optimization problem and the length of the prediction horizon. Moreover, as the optimization complexity increases, the number of linear gains associated with the PWA control law also increases enormously, which may cause serious difficulties on low-cost hardware.

The interest in deriving effective MPC solutions for a wide range of situations has driven the research community to study and propose answers to the issues entailed with explicit MPC approaches.

This thesis has addressed the following three issues connected with explicit MPC approaches.

- Approximation in explicit MPC approaches via Delaunay tessellations.
- Computation of feasible sets for MPC and of their suitable approximations.
- Maximal robust feasible sets for PWA controllers under polytopic model uncertainty.

The proposed results look into the geometric nature that is intrinsic to explicit MPC solutions.

6.1.1 Approximate Explicit MPC via Dealunay Tessellations

A geometric approach was presented for deriving an approximate explicit solution to linear constrained MPC problems. The solution is optimal for the portion of the feasible set where constraints are not active, on the remaining part of the feasible set the prohibitive optimal explicit MPC solution is replaced by an approximation based on Delaunay tessellations and computed from a finite number of samples of the exact solution. Finer tessellations can be obtained so as to achieve desired tolerance with the cost function approximation error.
The proposed approach suggests an answer to the cases where explicit MPC implementations are desired but impractical both due to the offline computational effort needed to compute the explicit optimal solution and due to its complexity for online evaluations. The approach draws its methods from computational geometry, a branch of computer science which focuses heavily on computational complexity since the algorithms are intended to be used on large data sets containing millions

of geometrical objects. The effectiveness of the approximate explicit MPC solution proposed is connected with the effective and efficient algorithms found in the computational geometry literature to generate Delaunay tessellations, store the relative data and point locate a query point.

The approach uses the cost function approximation error to decide whether a finer tessellation is needed or not. Since computing this error is in general computationally demanding, the approach proposes the use of easily computable over estimates. However, these bounds may be quite conservative, resulting in finer tessellations even if not really needed. This has been observed in the simulations section.
A future research direction would be to find less conservative, yet easily computable, estimates of the cost function approximation error.
Another possible research direction could try to decide whether or not a finer tessellation is needed looking at the optimal control values rather than the optimal cost function values. This has been recognized to be an hard problem, as discussed in Bemporad and Filippi (2006). An approximate explicit MPC approach following this line has been proposed in Jones and Morari (2009) using bilevel optimization.

Rigorous stability proofs were given when the approximate cost function is within a certain bound around the optimal cost function, such that the optimal cost function is a Lyapunov function. This is the idea followed by several approaches to approximate explicit MPC strategies. However, the conditions given are quite conservative. In fact, stability was proven post-analyzing approximate PWA controllers with cost functions values that did not satisfy the theoretic maximum allowed tolerance.
A research direction in this aspect would be to derive less conservative stability conditions to use within the algorithm for generating guaranteed stabilizing suboptimal PWA controllers.
Research effort could be put also on different stability approaches like the one presented in Hovd et al. (2009), where the quality of the approximation is gradually increased by refining the Delaunay tessellation until it can be shown that the approximate cost function is itself a Lyapunov function.

The approach introduced the concept of virtual constraints and virtual vertices to deal with the non-convexity of the portion of feasible set for which an approximate controller is needed. However the virtual vertices have the disadvantage to add a certain level of complexity to the approximate explicit solution.
A further research direction would be to investigate the possibility to use *constrained Delaunay tessellations* to deal with the non-convex region. Given a set of vertices with a set of non-crossing edges, the constrained Delaunay tessella-

tion is the simplicial tessellation of the vertices such that the prespecified edges are included in the tessellation and it is as close as possible to the “unconstrained” Delaunay tessellation (Seidel (1988), Bern and Plassmann (2000)).

6.1.2 Computation of Feasible Sets and of Suitable Approximations

An alternative approach was suggested for computing feasible sets when MPC techniques are used. The approach uses set relations instead of the conventional orthogonal projection, which then unfolds to a procedure based on Minkowski sum and intersection routines. This proves to be computationally more efficient and algorithmically more robust than using projection routines, particularly when high dimensional polytopic sets are involved (i.e. for long prediction horizons, high dimensional state and/or input).
However, some numerical issue suggested the need of future work to improve the algorithmic robustness of the routines for the needed polytopic operations.

When the feasible set is characterized by a critical complexity of representation in terms of number of vertices, an approach to compute a simplified feasible set with a reduced number of vertices was given. The approach is based on the introduction of certain conditions which extend existing approaches for the computation of polytope approximations, such that the approximating polytope maintains the fundamental properties of the feasible set required for MPC applications: positive invariance and inclusion of the operating set.

Preserving the positive invariance property in the feasible set approximation is crucial. This issue is inherently difficult to handle since it is concerned with the nonlinear dynamics of the closed-loop system. The proposed approach typically allows a considerable decrease in the $\mathcal{V}$-representation complexity by removing most of the vertices needed to deal with the feasible set borders which approximate ellipsoids (according with the operating set considered). However, this approach does not allow to consider possible simpler feasible set approximations which, while including the operating set, may have borders within $\mathcal{X}_F^{(N-1)}$. A potential future research direction could be to search for different approaches which would give more flexibility. One could for example look at solutions which use level surfaces of Lyapunov functions (Alessio et al. (2006)) to find different vertices than the original ones from the feasible set.

The positive invariance and operating set inclusion conditions constrain the goal of minimizing the loss of volume in the approximation. In general, finding suitable approximating polytopes characterized by the minimum loss of volume is a well known problem. Requiring that the approximation minimizes the loss of volume while satisfying conditions related to system dynamics represents a continuous challenge which could be an interesting topic for future work. Here the minimization of the loss of volume was not considered critical, since in the context of the present work the interest often is to preserve given crucial parts of the feasible set, which can be done via the operating set condition. Indeed, the algorithm proposed tends to minimize the loss of volume in the sense that at each iteration the suitable vertex which results in the lowest loss of volume in the current approximating polytope is removed. Pointer structures were used to enhance the implementation efficiency, though it may be further improved by a careful re-implementation of the approach.

6.1.3 Maximal Robust Feasible Sets for PWA controllers under Polytopic Model Uncertainty

An algorithm was presented which, given a nominal model with an associated polytopic uncertainty and a continuous PWA controller, identifies the largest subset of the state space which is guaranteed to contain all and only the initial conditions feasible for the entire family of models described by the polytopic uncertainty.
This can be used as a tool for analyzing how uncertainty in the real system affects the PWA feedback law computed using the nominal model (e.g. by means of some explicit MPC approach), thereby providing the maximal subset of the state space which contains safe initial conditions under the model uncertainty considered. This is a fundamental step towards any successful practical implementation of the controller.
The maximal robust feasible set thus obtained is, in general, non-convex. It is not required to be robustly positive invariant, and is computed in finite time. Moreover, any subset, and thus also any convex subset, still preserves the property of being robustly feasible.

This result may be used to decide whether or not a nominal design can be used without resorting to a more complex robust design. On the other hand, it can also be seen as an enabling technology for several future approaches to the problem of enhancing the robustness of (approximate) explicit MPC solutions towards model uncertainty. If the maximal robust feasible set does not cover the portion of state space of interest, the next step could be to consider just the regions not satisfying the robust feasibility condition, and search for suitable controllers for those regions.

One approach could be to define and solve a new explicit MPC problem for each infeasible region, with proper constraints ensuring robust feasibility, and the maximal robust feasible set as the new terminal set.

The possibility of limit cycles or chaotic behavior of the uncertain system in closed-loop with the controller, originally designed for the nominal system, is excluded when a particular condition is verified (assumption A2). This condition is easy to check, but it is also quite conservative. Future research can be directed to reduce this conservativeness.

An interesting future work would be the extension to LPV systems. This could be achieved propagating back the infeasible regions from the phase A of the algorithm for all the possible uncertain realizations in the polytopic uncertainty set. This however would reasonably result in heavier computational loads.
Research effort could also be directed to the inclusion of robustness with respect to disturbances.

References

Alessio, A., Bemporad, A., Lazar, M., and Heemels, W. P. M. H. (2006). Convex polyhedral invariant sets for closed-loop linear MPC systems. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 4532–4537.

Anderson, B. D. O. and Moore, J. B. (1979). *Optimal filtering*. Prentice-Hall.

Bemporad, A., Borrelli, F., and Morari, M. (2002a). Model predictive control based on linear programming - the explicit solution. *IEEE Transactions on Automatic Control*, 47(12):1974–1985.

Bemporad, A. and Filippi, C. (2003). Suboptimal explicit receding horizon control via approximate multiparametric quadratic progamming. *Journal of Optimization Theory and Applications*, 117(1):9–38.

Bemporad, A. and Filippi, C. (2006). An algorithm for approximate multiparametric convex programming. *Computational Optimization and Applications*, 35(1):87–108.

Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N. (2002b). The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20.

Bern, M. and Plassmann, P. (2000). Mesh generation. In *Handbook of Computational Geometry. Elsevier Science*, pages 291–332.

Blanchini, F. (1994). Ultimate boundedness control for uncertain discrete-time systems via set-induced Lyapunov functions. *IEEE Transactions on Automatic Control*, 39(2):428–433.

Blanchini, F. (1999). Set invariance in control. *Automatica*, 35:1747–1767.

Blanchini, F. and Miani, S. (2008). *Set-Theoretic Methods in Control*. Birkhauser.

Blandford, D. K., Blelloch, G. E., Cardoze, D. E., and Kadow, C. (2005). Compact representations of simplicial meshes in two and three dimensions. *International Journal of Computational Geometry and Applications*, 15(1):3–24.

Boissonnat, J.-D., Devillers, O., and Hornus, S. (2009). Incremental construction of the Delaunay triangulation and Delaunay graph in medium dimension. In *25th Annual Symposium on Computational Geometry*.

Boyd, S., Ghaoui, L. E., Feron, E., and Balakrishnan, V. (1994). *Linear Matrix Inequalities in System and Control Theory*. SIAM.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Brisson, E. (1993). Representing geometric structures in d dimensions: topology and order. *Discrete and Computational Geometry*, 9(4):387–426.

Bronstein, E. M. (2008). Approximation of convex sets by polytopes. *Journal of Mathematical Science*, 153(6):727–762.

Burger, T., Gritzmann, P., and Klee, V. (1996). Polytope projection and projection of polytopes. *The American Mathematical Monthly*, 103(9):742–755.

Cazals, F. and Giesen, J. (2006). *Effective Computational Geometry for Curves and Surfaces*, chapter Delaunay Triangulation Based Surface Reconstruction, pages 231–276. Springer Berlin Heidelberg.

Celes, W., Paulino, G. H., and Espinha, R. (2005). A compact adjacency-based topological data structure for finite element mesh representation. *International Journal for Numerical Methods in Ingeneering*, 64:1529–1556.

Chmielewski, D. and Manousiouthakis, V. (1996). On constrained infinite-horizon linear quadratic optimal control. *Systems Control Lett.*, 29(3):121–129.

Christophersen, F. J. (2007). *Optimal Control of Constrained Piecewise Affine Systems*, volume 359 of *Lecture Notes in Control and Information Sciences*, chapter Efficient evaluation of piecewise control laws defined over a large number of polyhedra, pages 150–165. Springer Berlin Heidelberg.

Cignoni, P., Montani, C., and Scopigno, R. (1998). DeWall: A fast divide and conquer Delaunay triangulation algorithm in E^d. *Computer-aided design*, 30(5):333–341.

Cuzzola, F. A., Geromel, J. C., and Morari, M. (2002). An improved approach for constrained robust model predictive control. *Automatica*, 38:1183–1189.

Cychowski, M. T., Ding, B., and O'Mahony, T. (2005). An orthogonal partitioning approach to simplify robust model predictive control. In *Proceedings of the 13th Mediterranean Conference on Control and Automation*, pages 877–882.

Dabbene, F., Gay, P., and Polyak, B. T. (2003). Recursive algorithms for inner ellipsoidal approximation of convex polytopes. *Automatica*, 39(10):1773–1781.

Darby, M. L. and Nikolaou, M. (2007). A parametric programming approach to movin-horizon state estimation. *Automatica*, 43:885–891.

de Berg, M., Cheong, O., van Kreveld, M., and Overmars, M. (2008). *Computational Geometry, Algorithms and Applications*. Springer Berlin Heidelberg.

de la Peña, D. M., Bemporad, A., and Filippi, C. (2004). Robust explicit MPC based on approximate multi-parametric convex programming. In *Proceedings of the 43rd IEEE Conference on Decision and Control*.

Devroye, L., Lemaire, C., and Moreau, J. (1999). Fast Delaunay point location with search structures. In *Proceedings of the 11th Canadian Conference on Computational Geometry*, pages 15–18.

Devroye, L., Lemaire, C., and Moreau, J. (2004). Expected time analysis for Delaunay point location. *Computational Geometry*, 29:61–89.

Dyer, R., Zhang, H., and Möller, T. (2009). A survey of Delaunay structures for surface representation. Technical Report TR 2009-01, GrUVi Lab, School of Computing Science.

Fukuda, K. (2004). Frequently asked questions in polyhedral computation. Technical report, [Online], http://www.ifor.math.ethz.ch/ fukuda/polyfaq/polyfaq.html.

Gallier, J. (2000). *Geometric Methods and Applications for Computer Science and Engineering*. Springer-Verlag.

Geyer, T., Torrisi, F. D., and Morari, M. (2004). Optimal complexity reduction of piecewise affine models based on hyperplane arrangements. In *Proceedings of the American Control Conference*, pages 1190–1195.

Gilbert, E. G. and Tan, K. T. (1991). Linear systems with state and control constraints: the theory and application of maximal output admissible sets. *IEEE Transactions on Automatic Control*, 36(9).

Grancharova, A. and Johansen, T. A. (2005). *Survey of Explicit Approaches to Constrained Optimal Control*. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

Grieder, P., Parrillo, P. A., and Morari, M. (2003). Robust receding horizon control - analysis and synthesis. In *Proceedings of the 42nd IEEE Conference on Decision and Control*.

Grieder, P., Wan, Z., Kothare, M., and Morari, M. (2004). Two level model predictive control for the maximum control invariant set. In *Proceedings of the American Control Conference*, pages 1586–1591.

Gritzmann, P. and Klee, V. (1994a). On the complexity of some basic problems in computational convexity: 2. volume and mixed volumes. Technical report, DIMACS.

Gritzmann, P. and Klee, V. (1994b). On the complexity of some basic problems in computational convexity: I. containment problems. *Discrete Mathematics*.

Gritzmann, P. and Sturmfels, B. (1993). Minkowski addition of polytopes: computational complexity and applications to Göbner bases. *SIAM J. Disc. Math.*, 6(2):246–269.

Gutman, P. and Cwikel, M. (1986). Admissible sets and feedback control for discrete-time linear dynamical systems with bounded controls and states. *IEEE Transactions on Automatic Control*, AC-31(4).

Hjelle, Ø. and Dæhlen, M. (2006). *Triangulations and Applications*. Mathematics and Visualization. Springer.

Hovd, M. and Olaru, S. (2010). Piecewise quadratic Lyapunov functions for stability verification of approximate explicit MPC. *Modeling, Identification and Control*, 31(2):45–53.

Hovd, M., Scibilia, F., Maciejowski, J. M., and Olaru, S. (2009). Verifying stability of approximate explicit MPC. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 6345–6350.

Hovland, S., Willcox, K. E., and Gravdhal, J. (2008). Explicit MPC for large-scale systems via model reduction. *AIAA J. Guidance, Control and Dynamics*, 31(4).

Ikonen, E. and Najim, K. (2002). *Advanced process identification and control*. Control Engineering Series. Marcel Dekker, Inc.

Johansen, T. A. (2003). Reduced explicit constrained linear quadratic regulators. *IEEE Transactions on Automatic Control*, 48(5):823–828.

Johansen, T. A. (2004). Approximate explicit receding horizon control of constrained nonlinear systems. *Automatica*, 40:293–300.

Johansen, T. A., Fossen, T. I., and Tøndel, P. (2005). Efficient optimal constrained control allocation via multi-parametric programming. *AIAA J. Guidance, Control and Dynamics*, 28(3):506–515.

Johansen, T. A. and Grancharova, A. (2003). Approximate explicit constrained linear model predictive control via orthogonal search tree. *IEEE Transactions on Automatic Control*, 48:810–815.

Jones, C. and Morari, M. (2008). The double description method for the approximation of explicit MPC control laws. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 4724–4730.

Jones, C. N., Kerrigan, E. C., and Maciejowski, J. M. (2004). Equality set projection: A new algorithm for the projection of polytopes in halfspace representation. Technical report, Department of Engineering, Cambridge University.

Jones, C. N., Kerrigan, E. C., and Maciejowski, J. M. (2008). On polyhedral projection and parametric programming. *Journal of Optimization Theory and Applications*, 138:207–220.

Jones, C. N. and Morari, M. (2009). Approximate explicit MPC using bilevel optimization. In *Proceedings of the European Control Conference*, pages 2396–2401.

Keerthi, S. S. and Gilbert, E. G. (1988). 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*, 57:265–293.

Kerrigan, E. C. and Maciejowski, J. M. (2000). Invariant sets for constrained nonlinear discrete-time systems with application to feasibility in model predictive control. In *Proceedings of the 39th IEEE Conference on Decision and Control*.

Khachiyan, L., Boros, E., Borys, K., Elbassioni, K., and Gurvich, V. (2008). Generating all vertices of a polyhedron is hard. *Discrete and Computational Geometry*, 39((1-3)):174–190.

Khalil, H. (2000). *Nonlinear Systems*. Prentice Hall, third edition.

Kolmanovsky, I. and Gilbert, E. G. (1995). Maximal output admissible sets for discrete-time systems with disturbance inputs. In *Proceedings of the American Control Conference*, pages 1995–1999.

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

Kouvaritakis, B., Rossiter, J. A., and Schuurmans, J. (2000). Efficient robust predictive control. *IEEE Transactions on Automatic Control*, 45(8):1545–1549.

Kvasnica, M., Grieder, P., and Baoti, M. (2004). Multi-parametric toolbox. http://control.ee.ethz.ch/∼mpt/.

Kvasnica, M., Grieder, P., Baotic, M., and Christophersen, F. J. (2006). *Multi-Parametric Toolbox (MPT) documentation*. Swiss Federal Institute of Technology, http://control.ee.ethz.ch/∼mpt/.

Ljung, L. (2006). *System Identification: Theory for the User*. Prentice Hall Information and System Sciences Series, second edition.

Lopez, M. A. and Reisner, S. (2002). Linear time approximation of 3D convex polytopes. *Computational Geometry*, 23:291–301.

Maciejowski, J. M. (2002). *Predictive control with constraints*. Prentice Hall.

Matousek, J. (2002). *Lectures on Discrete Geometry*, volume 212 of *GTM*. Springer.

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

Motzkin, T. S., Raiffa, H., Thompson, G. L., and Thrall, R. M. (1953). The double description method. *H.W. Kuhn and A.W.Tucker, editors, Contributions to theory of games, Princeton University Press, Princeton, RI*, 2.

Mount, D. M. (2002). Computational geometry. Lecture Notes. Department of Computer Science, University of Maryland.

Mücke, E. P., Saias, I., and Zhu, B. (1999). Fast randomized point location without preprocessing in two- and three-dimensional Delaunay triangulations. *Computational Geometry*, 12:63–83.

Muske, K. R. and Rawlings, J. B. (1993). Model predictive control with linear models. *AIChE Journal*, 39(2):262–287.

Naidu, D. S. (2003). *Optimal Control Systems*. CRC Press LLC.

Nam, N. H., Olaru, S., and Hovd, M. (2010). Patchy approximate explicit model predictive control. In *International Conference on Control, Automation and Systems*.

Nicolao, G. D., Magni, L., and Scattolini, R. (1996). On the robustness of receding-horizon control with terminal constraints. *IEEE Transactions on Automatic Control*, 41(3):451–453.

Nienhuys, H.-W. and van der Stappen, A. F. (2003). Maintaining mesh connectivity using a simplex-based data structure. Technical Report UU-CS-2003-018, Institute of Information and Computing Sciences, Utrecht University.

Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer Series in Operations Research And Financial Engineering. Springer Verlag, second edition.

Pistikopoulos, E. N. (2009). Perspectives in multiparametric programming and explicit model predictive control. *AIChE Journal*, 55(8):1918–1925.

Pluymers, B., Rossiter, J. A., Suykens, J., and Moor, B. D. (2005a). A simple algorithm for robust MPC. In *Proceedings of the 16th IFAC World Congress*.

Pluymers, B., Rossiter, J. A., Suykens, J. A. K., and Moor, B. D. (2005b). The efficient computation of polyhedral invariant sets for linear systems with polytopic uncertainty. *Proceedings of the American Control Conference*.

Qin, S. J. and Badgwell, T. A. (2003). A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764.

Rao, V. G. and Bernstein, D. S. (2001). Naive control of the double integrator. *IEEE Control Systems Magazine*, 21:86–97.

Rawlings, J. B. and Mayne, D. Q. (2009). *Model predictive control: theory and design*. Nob Hill Publishing, LLC.

Rawlings, J. B. and Muske, K. R. (1993). Stability of constrained receding horizon control. *IEEE Transactions on Automatic Control*, 38(10):723–757.

Reisner, S., Schutt, C., and Werner, E. (2001). Dropping a vertex or a facet from a convex polytope. *Forum Math.*, 13:359–378.

Rossiter, J. A. and Grieder, P. (2005). Using interpolation to improve efficiency of multiparametric predictive control. *Automatica*, 41(4):637–643.

Rossiter, J. A., Kouvaritakis, B., and Gossner, J. R. (1995). Mixed objective constrained stable generalized predictive control. *Control Theory and Applications, IEE Proceedings*, 142(4):286–294.

Rossiter, J. A., Pluymers, B., Suykens, J., and Moor, B. D. (2005). A multi parametric quadratic programming solution to robust predictive control. In *Proceedings of the 16th IFAC World Congress*.

Scibilia, F., Bitmead, R. R., Olaru, S., and Hovd, M. (2009a). Maximal robust feasible sets for constrained linear systems controlled by piecewise affine feedback laws. In *The 7th IEEE International Conference on Control and Automation*.

Scibilia, F. and Hovd, M. (2009). Multi-rate moving horizon estimation with erroneous infrequent measurements recovery. In *Preprints of the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*.

Scibilia, F., Hovd, M., and Bitmead, R. R. (2008). Stabilization of gas-lift oil wells using topside measurements. In *Proceedings of the 17th IFAC World Congress*.

Scibilia, F., Hovd, M., and Olaru, S. (2010a). An algorithm for approximate explicit model predictive control via Delaunay tessellations. *European Journal of Control*, (submitted).

Scibilia, F., Olaru, S., and Hovd, M. (2009b). Approximate explicit linear MPC via Delaunay tessellation. In *Proceedings of the European Control Conference*, pages 2833–2838.

Scibilia, F., Olaru, S., and Hovd, M. (2010b). On feasible sets for MPC and their approximations. *Automatica*, (accepted for publication).

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

Seidel, R. (1988). Constrained Delaunay triangulation and Voronoi diagrams with obstacles. Technical Report 260, Inst. for Information Processing, Graz, Austria.

Simon, D. (2006). *Optimal state estimation - Kalman, H infinity, and nonlinear approaches*. Wiley-Interscience.

Spjøtvold, J., Rakovic, S. V., Tøndel, P., and Johansen, T. A. (2006). Utilizing reachability analysis in point location problems. In *Proceedings of the 45th IEEE Conference on Decision and Control*.

Su, P. and Drysdale, R. L. S. (1995). A comparison of sequential Delaunay triangulation algorithms. In *Proceedings of the 11th Annual Symposium on Computational Geometry*, pages 61–70.

Sznaier, M. and Damborg, M. (1987). Suboptimal control of linear systems with state and control inequality constraints. In *Proceedings of the 26th IEEE Conference on Decision and Control*, pages 761–762.

Tiwary, H. R. (2008a). On computing the shadows and slices of polytopes. *CoRR*, abs/0804.4150.

Tiwary, H. R. (2008b). On the hardness of computing intersection, union and Minkowski sum of polytopes. *Discrete and Computational Geometry*, 40:469–479.

Tøndel, P., Johansen, T. A., and Bemporad, A. (2003a). An algorithm for multi-parametric quadratic programming and explicit MPC solutions. *Automatica*, 39(3):489–497.

Tøndel, P., Johansen, T. A., and Bemporad, A. (2003b). Evaluation of piecewise affine control via binary search tree. *Automatica*, 39(5):945–950.

van den Boom, T. J. J. and Haverkamp, B. R. J. (2000). Towards a state-space polytopic uncertainty description using subspace model identification techniques. In *Proceedings of the American Control Conference*, pages 1807–1811.

Yan, J. and Bitmead, R. R. (2004). Incorporating state estimation into model predictive control and its application to network traffic control. *Automatica*, 41:595–604.

Zhu, B. (2006). Voronoi diagram and Delaunay triangulation: Applications and challenges in bioinformatics. In *Proceedings of the 3rd International Symposium on Voronoi Diagrams in Science and Engineering*.

Appendix A

Secondary Papers

This addendum comprises two published papers about two particular problems treated during the doctoral studies, but only marginally related to the main theme of the thesis.
The analyses of performance, feasibility and stability of most MPC schemes treat the controller as a full-state-feedback strategy (Mayne et al. (2000)). However, the state variables are not always fully measured, and in these cases a state estimation approach has to be adopted to obtain the state from the measurements.
State estimation is an important subject of research in engineering and mathematics. For a comprehensive discussion, the interested reader is referred to textbooks like Simon (2006), or the "classic" Anderson and Moore (1979).
The inclusion of state estimation into the MPC formulation problem has been considered in the literature, like for example in Yan and Bitmead (2004). For a detailed discussion on the argument the reader is referred for instance to Rawlings and Mayne (2009), a recent and comprehensive textbook on MPC.
The two works in this chapter deal with state estimation, but not with the explicit goal to be used in MPC approaches.

The work in Section A.1 regards the moving horizon state estimation (MHE), which is considered to be the dual problem in the state estimation area of the MPC problem. MHE is a Bayesian estimator which gives the state estimate as the maximizers of the conditional probability density function of the state evolution given the measurement evolution. For practical online implementations the whole measurement evolution cannot be considered explicitly in the optimization problem. The information from the measurements not explicitly considered are taken into account by a function, called the *arrival cost*, that compresses the measurement data. Many chemical plants rely on manual laboratory analyses to obtain key measurements. Such analyses are prone to errors which lead to faulty measurements. Depending on how the arrival cost is determined, the effect of a faulty measurement on the MHE may persist even after the faulty value is discovered and corrected. The paper describes a way for the MHE to quickly recover from such faulty analyses.

The work in Section A.2 considers a particular system in the oil industry: gas-lifted oil well. Gas-lift is a technique to maintain, or to increase, the production from oil wells characterized by low reservoir pressure. A negative aspect of this technique is that it can induce severe production flow oscillations, known as *casing-heading instability*. Conventional linear controllers can deal with this instability. However, these controllers use state variables which are, in general, not available for measurement. Hence, the state needs to be estimated from the available measurements. The state estimation solution proposed is the use of a nonlinear observer tailored on the particular system under consideration.

A.1 Multi-Rate MHE with Erroneous Infrequent Measurements Recovery

This section contains the paper “Multi-Rate Moving Horizon Estimation with Erroneous Infrequent Measurements Recovery" as it appears in the Preprints of the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, held in Barcelona, Spain, during the days June 30 to July 3, 2009.

Remark 11. It is interesting to note that explicit solutions to moving horizon state estimation have also been proposed in the literature (Darby and Nikolaou (2007)).

Preprints of the 7th IFAC Symposium on
Fault Detection, Supervision and Safety of Technical Processes
Barcelona, Spain, June 30 - July 3, 2009

Multi-Rate Moving Horizon Estimation with Erroneous Infrequent Measurements Recovery

Francesco Scibilia * **Morten Hovd** *

* *Norwegian University of Science and Technology, Department of Engineering Cybernetics, N-7491 Trondheim, Norway (e-mail: [francesco.scibilia][morten.hovd]@itk.ntnu.no).*

Abstract: Moving horizon estimation (MHE) represents a desirable approach in processes monitoring and/or control. MHE allows to take into account model uncertainties and unknown disturbances, to consider additional insight about the process in form of inequality constraints, and suggests a straightforward way to include infrequently occurring measurements, as the process history over a user defined horizon is utilized. The problem addressed in this paper is relevant in chemical processes, but may be relevant also in other application areas. In such processes it is common to have measurements at different sampling rates and with time delay. Often an operator is called to collect some of the infrequent measurements, and to insert them into the estimator. Industrial practitioners report that it is not rare for a wrong value to be inserted for such a manual measurement, leading to significant errors in the state and parameter estimates. Recovering from such errors can take hours. We propose a strategy to recover quickly from faulty infrequent measurements. Since nonlinear models are considered, to overcome the non-convex optimization problem resulting from the MHE, in the paper we consider an efficient formulation based on successive linearization.

1. INTRODUCTION

State estimators are often used to monitor and/or control important process states that cannot be measured directly. In many cases the measurements used in the estimation are available at multiple rates; this is a common situation in chemical processes where some measurements, such as temperatures and flowrates, are easily available online; some measurements, such as concentrations, are measured infrequently; and other measurements, such as average molecular weights, are made by using analysis methods that inherently take some time, and therefore they are available infrequently and with time delays (Tatiraju and Soroush [1997]). Moreover, in many real situations an operator is called to collect some of the infrequent measurements, and to insert them into the estimator. Industrial practitioners report that it is not rare for a wrong value to be inserted for such a manual measurement, leading to significant errors in the state and parameter estimates. Recovering from such errors can take hours. A challenge in these processes is to design estimators that can use both online and infrequent measurements effectively, that are robust against model uncertainties and unknown disturbances, and that can provide a way to recover from a faulty measurement. Different types of multi-rate state estimators have been used, as for examples extended Kalman filters (KF) (Elicabe et al. [1995]), Nonlinear State Estimators (Zambare and Soroush [2002]), Moving Horizon Estimators (MHE) (Kraemer et al. [2005]). However, the problem of a faulty infrequent measurement has not been considered.
MHEs constitute one of the most advanced and interesting approaches in estimation (Rao [2000], Muske and Rawlings [1994], Muske et al. [1993], Tenny [2002], Findeisen [1997]). MHEs allow easily to take into account additional insight about the process in form of inequality constraints, and moreover they suggest a straightforward way to include the infrequently occurring measurements, as the process history over a user defined horizon is utilized. When a nonlinear model is considered, the MHE results in a non-convex program and the implementation gives rise to a lot of computational difficulties related to the expense and reliability of solving the non-convex program online. The same issue arises in the dual control problem, the Model Predictive Control (known also as moving horizon control). Model predictive control (MPC) is a model-based control approach that uses a model of the plant in order to determine the optimal control actions respecting some linear constraints (Camacho and Bordons [2003], Tenny [2002]). At each sampling time, starting from the current state, an open-loop optimal control problem is solved over a finite horizon. The first element of the optimal control sequence is applied to the system. At the next time step, the computation is repeated starting from the new state and over the shifted horizon. A simple way to deal with nonlinear models in MPC is to perform successive linearization about the predicted trajectories that would be obtained if the extension to the current time of the previously computed optimal control sequence were used. In the paper we show how the same idea can be brought to MHE.
The paper provides a state-of-the-art of the MHE problem (sections 2 and 3), and explains exhaustively how MHE can be efficiently implemented using the successive linearization approach (section 4). In section 5 the background given in the previous sections is used to present a

technique to recover from erroneous infrequent measurements, which is the main contribution of the paper. A Continuous Stirred Tank Reactor (CSTR) is considered in section 6 to illustrate the proposed strategy. Section 7 concludes the paper.

2. MOVING HORIZON ESTIMATION

Consider the process described by the nonlinear discrete time system

$$x_{t+1} = f(x_t, u_t) + \omega_t \tag{1a}$$

$$y_t = g(x_t) + v_t \tag{1b}$$

where $x_t \in \mathbb{R}^n$ is the state vector, $u_t \in \mathbb{R}^q$ is the vector of inputs, $y_t \in \mathbb{R}^m$ is the process measurements vector, $\omega_t \in \mathbb{R}^n$ represents modeling uncertainties and unmeasurable disturbances, $v_t \in \mathbb{R}^m$ is the process measurements noise and $t = 0, 1, 2, \ldots$.

One strategy for solving the state estimation problem for the system (1) is to formulate it from the perspective of probability theory. Consider the conditional probability density function of the state evolution given the process measurements: $p(x_0, x_1, ..., x_T \mid y_0, y_1, ..., y_{T-1})$. An optimal solution then would be a state sequence that maximizes the conditional probability density function, that is the maximum a posteriori Bayesian (MAP) estimate

$$\begin{aligned}&\left\{\hat{x}_{0|T-1}, \hat{x}_{1|T-1}, ..., \hat{x}_{T|T-1}\right\} \in \\ &\arg \max_{x_0, x_1, ..., x_T} p(x_0, x_1, ..., x_T \mid y_0, y_1, ..., y_{T-1})\end{aligned} \tag{2}$$

where $\hat{x}_{k|T-1}$ is the optimal state estimate at time k given measurements up to time $T-1$.

The first estimator based on the maximization of the probability density function was the Batch Least Square Estimator (BLSE) (Jazwinski [1970]), known also as Full Information problem (FI) (Rao [2000]). Under the common assumption that the initial state is $N\left(P_0, \hat{x}_{0|-1}\right)$, ω is $N(Q, 0)$ and v is $N(R, 0)$, where $\hat{x}_{0|-1}$ is the initial guess of the state and P_0, Q and R are covariance matrices, the maximization of the density function leads to the optimization problem

$$\min_{x_0, \{\omega_k\}_{k=0}^{T-1}} \Phi_T(x_0, \{\omega_k\}) \tag{3}$$

s. t.

$$x_{k+1} = f(x_k, u_k) + \omega_k \tag{4a}$$

$$y_k = g(x_k) + v_k \tag{4b}$$

$$x_k \in \mathbb{X},\ \omega_k \in \mathbb{W},\ v_k \in \mathbb{V} \tag{4c}$$

$$k = 0, ..., T-1$$

where the cost function is

$$\Phi_T(x_0, \{\omega_k\}) = \sum_{k=0}^{T-1} \|v_k\|^2_{R^{-1}} + \|\omega_k\|^2_{Q^{-1}} + \|x_0 - \hat{x}_0^-\|^2_{P_0^{-1}} \tag{5}$$

We use the notation $\|x\|^2_A = x'Ax$. The matrices Q and R are the tuning parameters, and in addition to their statistical interpretation, the matrix Q can be seen as a measure of confidence in the model equations, the matrix R can be seen as a measure of confidence in the process data. The sets $\mathbb{W}$, $\mathbb{X}$ and $\mathbb{V}$ are closed and convex, and usually they are finite dimensional polyhedral sets

$$\begin{aligned}\mathbb{X} &= \{x_k \in \mathbb{R}^n | D_x x_k \le d_x\} \\ \mathbb{W} &= \{\omega_k \in \mathbb{R}^q | D_\omega \omega_k \le d_\omega\} \\ \mathbb{V} &= \{v_k \in \mathbb{R}^m | D_v v_k \le d_v\}\end{aligned} \tag{6}$$

The ability of imposing the constraints (4c) is the main advantage of the optimization based estimators. They can be used to improve the descriptions of the random variables x_k, ω_k and v_k [1], taking into account in the estimation additional insight of the process not expressed in the model (1).

The solution of (3) at time T is the pair $\left(\hat{x}_{0|T-1}, \{\hat{\omega}_k\}_{k=0}^{T-1}\right)$, which, when used as data in (1a), yields the estimation sequence $\left[\hat{x}_{0|T-1}\ \hat{x}_{1|T-1}\ ...\ \hat{x}_{T|T-1}\right]$. Note that the estimate $\hat{x}_{k|T-1}$, for $k = 0, 1, ..., T-1$, are filtered values, whereas for $k = T$ are predicted values. This because, in the formulation considered, at time T we have measurements only up to time $T-1$.

Due to the nonlinear model (4a-4b), problem (3) requires the solution of a nonlinear mathematical program, a quite computationally demanding problem. Regardless of the mathematical problem complexity, solving the state estimation problem online is practically impossible because the size of the problem grows without bound as more process measurements are collected. For practical online implementation a fixed size optimization problem is required, and then a way to compress the measurement data is needed. The approach proposed in Muske et al. [1993] is a recursive formulation of the BLSE on a moving fixed horizon N, where the effect of the data older than $T-N$ are summarized with a function $Z_{T-N}(x_{T-N})$. The estimator so obtained is known as Moving Horizon Estimator (MHE):

$$\min_{x_{T-N}, \{\omega_k\}_{k=T-N}^{T-1}} \Phi_T(x_{T-N}, \{\omega_k\}) \tag{7}$$

s.t. (4a-4c) are satisfied for all $k = T-N, ..., T-1$, where

$$\Phi_T(x_{T-N}, \{\omega_k\}) = \sum_{k=T-N}^{T-1} \| v_k\|^2_{R^{-1}} + \|\omega_k\|^2_{Q^{-1}} + Z_{T-N}(x_{T-N}) \tag{8}$$

With the moving horizon approach, only the last N process measurements are accounted explicitly at each time step, the remaining process measurements are taken into account using the function $Z_{T-N}(x_{T-N})$, known as arrival cost. The arrival cost summarizes the effect of the data $[y_0, ..., y_{T-N-1}]$ on the state x_{T-N}, thereby allowing to fix the dimension of the optimization problem. In probabilistic terms, the arrival cost can be seen as an equivalent statistic for the probability density function $p(x_{T-N}|y_0, ..., y_{T-N-1})$.

If we consider a linear process model and unconstrained estimation, an optimal arrival cost is the Riccati equation arising from Kalman filtering

$$\begin{aligned}&Z_{T-N}(x_{T-N}) = \\ &\|x_{T-N} - \hat{x}_{T-N|T-N-1}\|^2_{P^{-1}_{T-N|T-N-1}} + \Phi^*_{T-N}\end{aligned} \tag{9}$$

[1] The use of constraint on v_k should be considered carefully in practical implementations because it may amplify the effect of spurious measurements.

where $\hat{x}_{T-N|T-N-1}$ is the optimal estimate at time $T-N$ given the measurements $[y_0, ..., y_{T-N-1}]$, and $P_{T-N|T-N-1}$ is the covariance matrix obtained using the Kalman filter (KF) covaivance update formula (Anderson and Moore [1979]), subject to initial condition P_0. The term Φ^*_{T-N} denotes the optimal cost at time $T-N$. For unconstrained linear estimation, Muske and Rawlings [1994] showed that MHE and KF are identical for any horizon N.

Unfortunately, an algebraic expression for the arrival cost does not exist in a tractable form when either constraints are present or when the model is nonlinear. In these cases, an approximate algebraic expression needs to be used instead.

One approach is to use a first-order Taylor series approximation of the model around the estimated trajectory and approximate the arrival cost with an extended Kalman filter covariance update formula (Muske et al. [1993], Muske and Rawlings [1994]). This formulation is also known as filtering MHE.

Findeisen [1997] showed that the filtering formulation can result in oscillatory or cyclic behavior, which does not result for the comparable Kalman filter. Then, the solution proposed was to condition the estimate on the smoothed value $\hat{x}_{T-N|T-2}$ rater than on the predicted value $\hat{x}_{T-N|T-N-1}$. Since the smoothed value is based on more information than the predicted value, in the arrival cost a smoothed covariance matrix update formula (Anderson and Moore [1979]) needs to be used.

An issue with the smoothing update is that the data $[y_{T-N}, ..., y_{T-2}]$ are used twice in the optimization, and this, other than being statistically meaningless, turns N into a tuning parameter of the problem. To avoid the estimator to use twice the data, Rao [2000] developed a smoothing update scheme for linear systems which removes the information of the re-used measurements from the cost function:

$$Z_{T-N}(x_{T-N}) = \|x_{T-N} - \hat{x}_{T-N|T-2}\|^2_{P^{-1}_{T-N|T-2}} + \Phi^*_{T-1} - \|\mathrm{y}_{T-1} - \Theta_{T-1}x_{T-N}\|^2_{\mathcal{W}^{-1}_{T-1}} - \Phi^*_{T-N} \tag{10}$$

where $\hat{x}_{T-N|T-2}$ is the smoothed estimate, $P_{T-N|T-2}$ is the smoothed covariance matrix, $\mathrm{y}_{T-1} = [y'_{T-N}\ y'_{T-N+1}\ ...\ y'_{T-2}]'$, and the matrices Θ_{T-1} and $\mathcal{W}_{T-1}$ follows from properties of linear Gaussian difference equations.

Tenny [2002] extended the scheme to nonlinear systems approximating the nonlinear model as a time-varying discrete time system. This formulation is known as smoothing MHE. It was shown that with linear systems and unconstrained estimation, the filtering MHE and the smoothing MHE are equivalent.

3. MULTIRATE MOVING HORIZON ESTIMATION

The MHE formulation allows for the direct inclusion of additional infrequent measurements in the considered horizon (Kraemer et al. [2005]). Infrequent measurements can be seen as slow rate measurements with time delay.

Considering for clarity only two measurements vectors, one available at a fast sampling rate and the other at a slow sampling rate and with delay, the measurement vector can be written as

$$y_k = \begin{cases} g^F(x_k) & \text{if } k \neq l \\ \begin{bmatrix} g^F(x_k) \\ g^S(x_{k-t_d}) \end{bmatrix} & \text{if } k = l \end{cases} \tag{11}$$

where k indicates the fast sampling rate; $l = L\lfloor k/L \rfloor$, L being the number of steps between two slow sampling points (a slow sampling point coincides with a fast sampling point) and $\lfloor . \rfloor$ the nearest smaller integer value; t_d the slow measurement delay. The slow sampling rate and the measurement delay are integer multiplies of the fast sampling rate.

As discussed in Kraemer et al. [2005], there are two possible approaches to multirate estimation: fixed structure estimation, where a zero-order hold is used to extrapolate the estimation error of the slow measurement at the fast sample rate, and a variable structure estimation, where the estimation error is considered according to the availability of the measurements. It was shown that the fixed structure appears preferable as it is less susceptible to noise in the slow measurements.

The multirate MHE with fixed structure is

$$\min_{x_{T-N}, \{\omega_k\}_{k=T-N}^{T-1}} \Phi_T(x_{T-N}, \{\omega_k\}) \tag{12}$$

s.t.

$$\begin{gathered} x_{k+1} = f(x_k, u_k) + \omega_k \\ y_k = \begin{bmatrix} g^F(x_k) \\ g^S(x_{\bar{k}}) \end{bmatrix} + v_k \\ \omega_k \in \mathbb{W},\ x_k \in \mathbb{X},\ v_k \in \mathbb{V} \\ \bar{k} = \begin{cases} L(\lfloor k/L \rfloor - 1) & \text{if } k \geq \lfloor T/L \rfloor \text{ and } T < L\lfloor T/L \rfloor + t_d \\ L\lfloor k/L \rfloor & \text{otherwise} \end{cases} \end{gathered}$$

In the definition of $\bar{k}$ it is also taken into account the case $N \leq t_d$, even if it is convenient to have an horizon at least as long as the measurement delay. The measurement vector y_k has to be considered as composed by two sub-vectors: $y^F_k \in \mathbb{R}^{m^F}$ relative to the fast measurements, $y^S_k \in \mathbb{R}^{m^S}$ relative to the slow measurements, where $m = m^S + m^F$. Analogously has to be done for v_k.

4. IMPLEMENTATION USING SUCCESSIVE LINEARIZATION

The nonlinear model used in the MHE formulation makes the resulting optimization problem a nonconvex program, which gives rise to a lot of computational difficulties related to the expense and reliability of solving the nonconvex program online.

Successive linearization is a successful approach used in model predictive control to tackle the same problem. In this section we show how the same approach is applied to obtain an efficient MHE formulation (Tenny [2002]).

Taylor first order approximation is used to linearize the nonlinear model. Consider a system $x_{k+1} = f(x_k, u_k) + \omega_k$ and a point $\bar{x}$ (since u_k is know, it can be considered as a constant), where $f(x_k, u_k)$ is a nonlinear function differentiable in x_k. The Taylor first order approximation of the system about the point $\bar{x}$ is $x_{k+1} \approx A_{\bar{x}}x_k + b_{\bar{x}} + \omega_k$ where $A_{\bar{x}} = \frac{\partial f}{\partial x}|_{x=\bar{x}}$ and $b_{\bar{x}} = f(\bar{x}) - A_{\bar{x}}\bar{x}$.

The cost function (8) can be written as

$$\Phi_T = \|\mathrm{v}_T\|^2_{R^{-1}_T} + \|\mathrm{w}_T\|^2_{\mathrm{Q}^{-1}_T} + Z_{T-N}(x_{T-N}) \tag{13}$$

where $\mathrm{v}_T = \left[v'_{T-N} \dots v'_{T-1}\right]'$, $\mathrm{w}_T = \left[\omega'_{T-N} \dots \omega'_{T-1}\right]'$, $\mathrm{R}_T^{-1} = \mathrm{blkdiag}(R^{-1}, N)$ [2] and $\mathrm{Q}_T^{-1} = \mathrm{blkdiag}(Q^{-1}, N)$.
From the measurement equation we have $v_k = y_k - g(x_k)$, where in the case of multirate measurement g has to be interpreted as the composition of g^F and g^S. For simplicity of explanation, suppose g is linear, i.e., $v_k = y_k - Cx_k$, $C \in \mathbb{R}^{m\times n}$. This is not restrictive since the case of a nonlinear function g can be brought to the linear case using Taylor approximation.
Then we have

$$\|\mathrm{v}_T\|^2_{\mathrm{R}_T^{-1}} = \|\mathrm{y}_T - \mathrm{C}_T\mathrm{x}_T\|^2_{\mathrm{R}_T^{-1}} \tag{14}$$

where $\mathrm{y}_T = \left[y'_{T-N} \dots y'_{T-1}\right]'$, $\mathrm{x}_T = \left[x'_{T-N} \dots x'_{T-1}\right]'$ and $\mathrm{C}_T = \mathrm{blkdiag}(C, N)$.
Consider now the estimation sequence from the previous time step, i.e., $\left[\hat{x}_{T-N-1|T-2}\ \hat{x}_{T-N|T-2} \dots \hat{x}_{T-1|T-2}\right]$. Then the nonlinear state evolution of the system (1a) can be approximated by the time-varing linear system

$$x_{k+1} \approx A_k x_k + b_k + \omega_k \tag{15}$$

where $A_k = \frac{\partial f}{\partial x}|_{x=\hat{x}_{k|T-2}}$, $b_k = f(\hat{x}_{k|T-2}) - A_k\hat{x}_{k|T-2}$, for all $k = T-N-1, ..., T-1$.
Repeated use of (15) and organizing so to write all the optimization variables in the vector $\mathrm{z}_T = \left[x'_{T-N}\ \omega'_{T-N}\ \omega'_{T-N+1} \dots \omega'_{T-1}\right]'$, gives

$$\mathrm{x}_T = \mathrm{A}_T\mathrm{z}_T + \mathrm{b}_T \tag{16}$$

where
$\mathrm{A}_T =$

$$\begin{bmatrix} I & 0 & 0 & 0 & \cdots & 0 \\ A_{T-N} & I & 0 & 0 & \cdots & 0 \\ A_{T-N+1}A_{T-N} & A_{T-N+1} & I & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & & \vdots \\ \prod_{k=T-N}^{T-2} A_k & \prod_{k=T-N+1}^{T-2} A_k & \prod_{k=T-N+2}^{T-2} A_k & \cdots & & I\ 0 \end{bmatrix}$$

$$\mathrm{b}_T = \mathrm{A}_T \left[0\ b'_{T-N}\ b'_{T-N+1} \dots b'_{T-2}\ 0\right]'$$

We indicate with I the identity matrix and with 0 the zero matrix of dimension suitable to where it is used.
Using the linearized model (15), we can write an algebraic approximation of the arrival cost using the smoothing update formula (10)

$$\begin{aligned} Z_{T-N}(x_{T-N}) = {} & \|x_{T-N} - \hat{x}_{T-N|T-2}\|^2_{P^{-1}_{T-N|T-2}} - \\ & \|\mathrm{y}_{T-1} - \Theta_{T-1}x_{T-N} - F_{T-1}\|^2_{\mathcal{W}^{-1}_{T-1}} \end{aligned} \tag{17}$$

where the terms Φ^*_{T-1} and Φ^*_{T-N} are removed because they do not influence the optimization. The value $\hat{x}_{T-N|T-2}$ is the smoothed estimate from the previous step. $P_{T-N|T-2}$ is the smoothed covariance matrix and it is obtained first propagating forward the covariance $P_{T-N|T-N-1}$ using the well known Kalman filter formulas

$$P_{k|k-1} = Q + A_{k-1}P_{k-1|k-1}A'_{k-1}$$

$$P_{k|k} = P_{k|k-1} - P_{k|k-1}C'\left(R + C'P_{k|k-1}C\right)^{-1}CP_{k|k-1}$$

and then using the backward Riccati equation to compute the smoothed covariance

$$P_{k|j} = P_{k|k} + P_{k|k}A'_kP_{k+1|k}\left(P_{k+1|j} - P_{k+1|k}\right)P^{-1}_{k+1|k}A_kP_{k|k}$$

starting from $P_{T-2|T-2}$.
The second term in (17) avoids to reuse twice the information relative to $[y_{T-N}\ y_{T-N+1} \dots y_{T-2}]$, and it is characterized by

$$F_{T-1} = \begin{bmatrix} 0 \\ Cb_{T-N} \\ C\left(A_{T-N+1}b_{T-N} + b_{T-N+1}\right) \\ \vdots \\ C\sum_{j=T-N+1}^{T-3}\prod_{k=T-N+1}^{j} A_k b_{k-1} + Cb_{T-3} \end{bmatrix}$$

$$\Theta_{T-1} = \begin{bmatrix} C \\ CA_{T-N} \\ \vdots \\ C\prod_{k=T-N}^{T-3} A_k \end{bmatrix}$$

$$\mathcal{W}_{T-1} = R_{T-1} + C_{T-1}\mathrm{W}_{T-1}C'_{T-1}$$

where $\mathrm{R}_{T-1} = \mathrm{blkdiag}(R, N-1)$, $\mathrm{C}_{T-1} = \mathrm{blkdiag}(C, N-1)$ and
$\mathrm{W}_{T-1} =$

$$\begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & Q & \cdots & Q\prod_{k=T-N+1}^{T-3} A'_k \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \prod_{k=T-N+1}^{T-3} A_kQ & \cdots & Q + \sum_{j=T-N+1}^{T-3}\left\|\prod_{k=T-N+1}^{j} A_k\right\|^2_Q \end{bmatrix}$$

Considering constraints of the form (6), it is trivial to write them in terms of the vector z_T, obtaining the compact form

$$\mathrm{D}_T\mathrm{z}_T \le \mathrm{d}_T. \tag{18}$$

Then, an approximate solution of problem (12) can be obtained solving the quadratic program

$$\begin{aligned} \min_{\mathrm{z}_T} & \ \frac{1}{2}\mathrm{z}'_T\mathrm{H}_T\mathrm{z}_T + \mathrm{f}'_T\mathrm{z}_T \\ \text{s.t. } & \mathrm{D}_T\mathrm{z}_T \le \mathrm{d}_T \end{aligned} \tag{19}$$

where

$$\mathrm{H}_T = \mathrm{A}'_T\mathrm{C}'_T\mathrm{R}_T^{-1}\mathrm{C}_T\mathrm{A}_T + \bar{\mathrm{Q}}_T^{-1} + \bar{P}_T^{-1} + \bar{\Theta}'_{T-1}\mathcal{W}^{-1}_{T-1}\bar{\Theta}_{T-1}$$

$$\begin{aligned} \mathrm{f}_T = {} & -2\mathrm{y}'_T\mathrm{R}_T^{-1}\mathrm{C}_T\mathrm{A}_T - 2\bar{\mathrm{z}}'_T\bar{P}_T^{-1} + 2\mathrm{b}'_T\mathrm{C}'_T\mathrm{R}_T^{-1}\mathrm{C}_T\mathrm{A}_T - \\ & 2\mathrm{y}'_{T-1}\mathcal{W}^{-1}_{T-1}\bar{\Theta}_{T-1} + 2F'_{T-1}\mathcal{W}^{-1}_{T-1}\bar{\Theta}_{T-1} \end{aligned}$$

and the terms not influencing the optimization are neglected. The matrices $\bar{\mathrm{Q}}_T^{-1} = \mathrm{diag}(0, \mathrm{Q}_T^{-1})$ [3], $\bar{P}_T = \mathrm{diag}(P_{T-N|T-2}, 0)$ and $\bar{\Theta}_{T-1} = [\Theta_{T-1}\ 0]$ are introduced to be consistent with the definition of the optimization vector z_T.
The matrices H_T, f_T, D_T and d_T in the optimization problem (19) are time-varying due to the time-varying model (15), thus at each time step they need to be computed. Assuming that an algebraic form of $\frac{\partial f}{\partial x}$ is available, this operation amounts to the evaluation of several algebraic equations, and generally it is far less demanding than solving a non-convex optimization problem.

[2] blkdiag(A, n) indicates the n times block diagonal concatenation of a matrix A.

[3] diag(A, B) indicates the diagonal concatenation of two matrices A and B.

5. RECOVERY STRATEGY FROM ERRONEOUS INFREQUENT MEASUREMENTS

Some measurements, such as average molecular weights, are made by using analysis methods that inherently take some time, and therefore they are available infrequently and with time delays. In many real situations, an operator is called to collect the infrequent measurements, and to insert them into the estimator. Cases when a wrong measurement value is inserted by the operator are not rare, and it constitutes a problem since it affects the reliability of the state estimate for a certain time (the time needed by the estimator to compensate the fault with new correct measurements). The operator often realizes the error. The aim of this paper is to offer a strategy to recover from erroneous infrequent measurements once the error has been discovered.

Let us consider the situation represented in Fig.1. At time step $T = 6$ a slow measurement value y_5^S becomes available and is inserted into the MHE (for simplicity we ignore delays here). Suppose that a wrong value $\tilde{y}_5^S$ is inserted instead. Consequently, the state estimate $\left[\hat{x}_{0|5}\ \hat{x}_{1|5}\ ...\ \hat{x}_{6|5}\right]$, $\left[\hat{x}_{1|6}\ \hat{x}_{2|6}\ ...\ \hat{x}_{7|6}\right]$, ..., are affected by the wrong value, and without any action, depending from the particular process, a certain number of new measurements will be needed to compensate the fault and obtain again reliable estimate. Suppose now that at time $T = 9$ the wrong measurement is realized. Since we are still inside the horizon of the MHE, it makes sense to replace it with the correct value y_5^S. Unfortunately, this will not result in a reliable estimation as one may expect, indeed the effect of the erroneous measurement will still propagate in the current estimation through the smoothed update [4]. Moreover, since we are using successive linearization, the effect of the past fault may be even more severe.
The solution we propose is to use the last reliable estimate for the smoothed update and the successive linearization. Reasonably, the last reliable estimate is the estimate computed before the wrong measurement was inserted into the MHE. Thus, in the case of Fig.1, the smoothed update will be done using $\hat{x}_{3|4}$ and the successive linearization using $\left[\hat{x}_{2|4}\ \hat{x}_{3|4}\ ...\ \hat{x}_{8|4}\right]$. Note that we use predicted values in place of the missing filtered values for the successive linearization.
Since we do not know a priori if a certain infrequent measurement will be faulty, our strategy supposes to store the last estimate obtained before each new infrequent measurement is considered. As long as an infrequent measurement, y_k^S, remains within the estimation horizon, i.e. $k \leq T - N$, in case of recovery action the update of the arrival cost will be done using a smoothed estimate (in the case $k = T - N$ the smoothed update will coincide with the filtered update). For $k > T - N$ the infrequent measurement is beyond the horizon, and the update will be a predicted update, i.e. a predicted value and a predicted covariance will be used in (9). We suggest to discard the stored estimate associated to an infrequent measurement once the measurement goes beyond the horizon. We motivate this choice considering that all the data beyond the horizon are taken into account via the arrival cost, and any erroneous infrequent measurement would fade. Thus, the amount of data needing to be stored is limited.

[4] This does not happen with the filtering formulation.

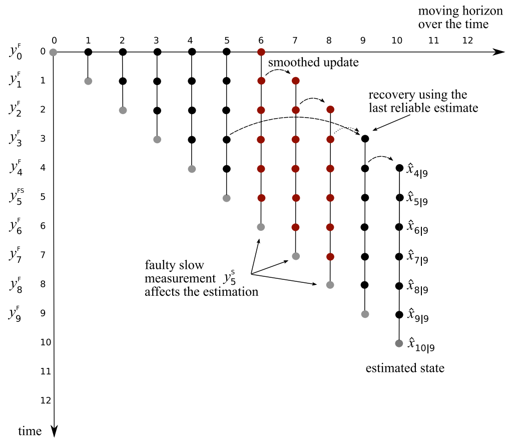


Fig. 1. Concept of the smoothed moving horizon estimation with recovery strategy for erroneous infrequent measurement. For clarity delays are ignored.

6. APPLICATION TO A CSTR

A Continuous Stirred Tank Reactor (CSTR)(Soroush and Kravaris [1992]) is considered in this section. In the CSTR the irreversible reactions $A \overset{k_1}{\rightarrow} U_1$, $A \overset{k_2}{\rightarrow} U_2$, $A \overset{k_d}{\rightarrow} D$ take places, where U_1 and U_2 are undesired side products, and D is the desired product. The reaction rate constants depend on the temperature T^c as $k_1 = Z_1 \exp\left(-\frac{E_{a_1}}{RT^c}\right)$, $k_2 = Z_2 \exp\left(-\frac{E_{a_2}}{RT^c}\right)$ and $k_d = Z_d \exp\left(-\frac{E_{a_d}}{RT^c}\right)$. Energy and species mass balances for the reactor, under standard assumptions, give the reactor model equations

$$\frac{dC_a}{dt} = R_a + \frac{C_{A_i} - C_A}{\tau} \tag{20a}$$

$$\frac{dT^c}{dt} = \frac{R_H}{\rho c} + \frac{T_i^c - T^c}{\tau} + \frac{Q_r}{\rho c V} \tag{20b}$$

$$\frac{dC_D}{dt} = R_D - \frac{C_D}{\tau} \tag{20c}$$

where the rate expressions are $R_A = -k_1 C_A^{n_1} - k_2 C_A^{n_2} - k_d C_A^{n_d}$, $R_H = -\Delta H_1 k_1 C_A^{n_1} - \Delta H_2 k_2 C_A^{n_2} - \Delta H_d k_d C_A^{n_d}$ and $R_D = k_d C_A^{n_d}$.
In Table 1 are given the parameters of the reactor, the operating conditions considered in the simulations and the steady state of interest.
The continuous time equations (20) were discretized using Euler's method with a sampling period $\Delta t = 10s$. The temperature T^c is measured frequently every $10s$; the concentrations C_A and C_D are measured infrequently. For simplicity we suppose to not have delay ($t_d = 0$). The horizon chosen for the MHE is $N = 300s$.
In Fig.2 and Fig.3 we consider the case of an erroneous value of the concentration C_A inserted at time $t_e = 750s$: the correct value is $y_{C_A}(t_e) = 1.42$ kmol m^{-3}, the value inserted erroneously is $\tilde{y}_{C_A}(t_e) = 2.42$ kmol m^{-3}. At time

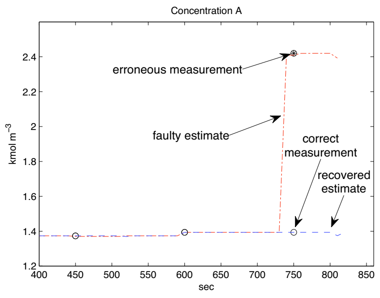


Fig. 2. Recovery from the erroneous measurement at $t = 750s$. The circles represent the infrequent measurements.

$t = 820s$ the error is realized, and the correct value $y_{C_A}(t_e)$ is replaced. Fig.2 shows how the estimate is recovered using the strategy proposed. The entire estimation scenario simulated is not affected by the erroneous measurement (Fig.3).

Table 1. Parameters, Operating Conditions and Steady State of the CSTR

$R = 8.345$	kJ kmol^{-1} K^{-1}
$Z_1 = 2.00 \cdot 10^3$	m^6 kmol^{-2} s^{-1}
$Z_2 = 3.40 \cdot 10^6$	m$^{-1.5}$ kmol$^{-0.5}$ s^{-1}
$Z_d = 2.63 \cdot 10^5$	s^{-1}
$E_{a_1} = 4.90 \cdot 10^4$	kJ kmol^{-1}
$E_{a_2} = 6.50 \cdot 10^4$	kJ kmol^{-1}
$E_{a_d} = 5.70 \cdot 10^4$	kJ kmol^{-1}
$-\Delta H_1 = 4.50 \cdot 10^4$	kJ kmol^{-1}
$-\Delta H_2 = 5.00 \cdot 10^4$	kJ kmol^{-1}
$-\Delta H_d = 6.00 \cdot 10^4$	kJ kmol^{-1}
$n_1 = 3.00 \cdot 10^0$	
$n_2 = 5.00 \cdot 10^{-1}$	
$n_d = 1.00 \cdot 10^0$	
$\rho = 1.00 \cdot 10^3$	kg m^{-3}
$c = 4.20 \cdot 10^0$	kJ kg m^{-1} K^{-1}
$V = 1.00 \cdot 10^{-2}$	m^3
$\tau = 3.00 \cdot 10^2$	s
$T_i^c = 2.952 \cdot 10^2$	K
$C_{A_i} = 1.00 \cdot 10^1$	kmol m^{-3}
$C_{A0} = 1.00 \cdot 10^{-1}$	kmol m^{-3}
$C_{D0} = 0.00 \cdot 10^0$	kmol m^{-3}
$T_0^c = 2.952 \cdot 10^2$	K
$C_{A_{SS}} = 1.320 \cdot 10^0$	kmol m^{-3}
$T_{SS}^c = 4.00 \cdot 10^2$	K
$C_{D_{SS}} = 4.000 \cdot 10^0$	kmol m^{-3}
$Q_{r_{SS}} = -1.030 \cdot 10^0$	kJ s^{-1}

7. CONCLUSIONS

The paper reviewed the filtering and smoothing MHE formulations, illustrating explicitly an efficient implementation of the smoothing MHE based on the successive linearization approach. A particular smoothing formulation where the last reliable measurements are used for the smoothing update and the successive linearization was the basis for the strategy proposed to recover from erroneous infrequent measurements.
The CSTR considered suited the purpose of exemplifying the results obtained.

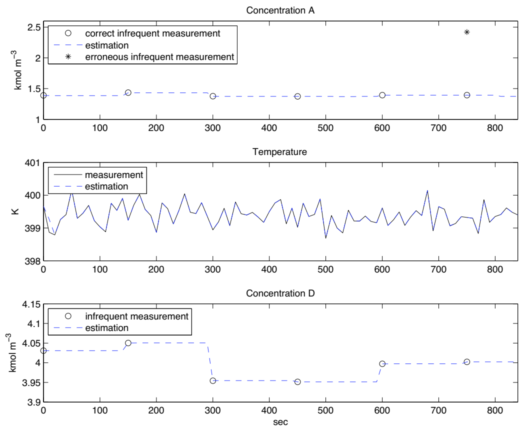


Fig. 3. Multirate moving horizon state estimation with recovery from the erroneous infrequent measurement.

REFERENCES

B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice Hall, 1979.

E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer, second edition, 2003.

G. E. Elicabe, E. Ozdeger, C. Georgakis, and C. Cordeiro. On-line estimation of reaction rates in semicontinuous reactors. *Ind. Eng. Chem. Res.*, 1995.

P. K. Findeisen. Moving horizon state estimation of discrete time systems. Master's thesis, University of Wisconsin-Madison, 1997.

A. H. Jazwinski. *Stochastic Process and Filtering Theory*. Academic Press, New York and London, 1970.

S. Kraemer, R. Gesthuisen, and S. Engell. Fixed structure multirate state estimation. In *American Control Conference*, 2005.

K. R. Muske and J. B. Rawlings. Nonlinear moving horizon state estimation. *NATO ASI Series, Kluwer Academic*, 293:349–365, 1994.

K. R. Muske, J. B. Rawlings, and J. H. Lee. Receding horizon recursive state estimation. *Proceedings of the American Control Conference*, pages 900–904, 1993.

C. V. Rao. *Moving Horizon Strategies for Constrained Monitoring and Control of Nonlinear Discrete-Time Systems*. PhD thesis, University of Wisconsin-Madison (USA), 2000.

M. Soroush and C. Kravaris. Discrete-time nonlinear controller synthesis by input/output linearization. *AIChE Journal*, 40:980–992, 1992.

S. Tatiraju and M. Soroush. Nonlinear state estimation in a polymerization reactor. *Ind. Eng. Chem. Res.*, 1997.

M. J. Tenny. *Computational Strategies for Nonlinear Model Predictive Control*. PhD thesis, University of Wisconsin-Madison (USA), 2002.

N. Zambare and M. Soroush. Multi-rate nonlinear state estimation in a polymerization reactor: a real-time study. *Proceedings of the American Control Conference*, 2002.

A.2 Stabilization of Gas-Lift Oil Wells Using Topside Measurements

This section contains the paper “Stabilization of Gas-Lift Oil Wells Using Topside Measurements” as it appears in the Proceedings of the 17th World Congress of the International Federation of Automatic Control (IFAC) held in Seoul, South Korea, during the days July 6-11, 2008.

Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Stabilization of gas-lift oil wells using topside measurements

Francesco Scibilia [*] **Morten Hovd** [*] **Robert R. Bitmead** [**]

[*] *Department of Engineering Cybernetics, Norwegian University of Science and Technology, N-7491 Trondheim, Norway (e-mail: [francesco.scibilia],[morten.hovd]@itk.ntnu.no).*
[**] *Department of Mechanical and Aerospace Engineering, University of California, San Diego, 9500 Gilman Drive, La Jolla, CA 92093-0411, USA (e-mail: rbitmead@ucsd.edu)*

Abstract: Highly oscillatory flow regimes that can occur in gas-lift oil wells have been successfully treated using conventional linear control. However, these control systems rely on downhole pressure measurements which are unreliable or even unavailable in some cases. In this paper we propose a solution based on a high gain observer for the state of the process. The estimates are used to compute the downhole pressure, that is the controlled variable considered in the feedback control. Moreover, we propose an estimator to extend a nonlinear observer already presented in the literature, and then we compare the performances. The key feature of the solution proposed is its simplicity and that it relies only on measurements easily obtainable from the top of the single well, and thus it is immediately applicable to multiple-well systems where, since there is often one common outflow manifold, it would be hard to see from the outflow measurements which well is operating in an oscillatory regime.

1. INTRODUCTION

Oil wells with highly oscillatory flow constitute a significant problem in the petroleum industry. This is the case, for instance, for oil wells on mature fields, where artificial lift techniques are used to increase tail-end production. Gas lift is one of the most widely used technologies to maintain, or to increase, the production from wells characterized by low reservoir pressure.
With gas lift, gas is injected into the tubing, as close as possible to the bottom of the well, and mixed with the fluid from the reservoir (Fig. 1). The gas reduces the density of the fluid in the tubing, which reduces the downhole pressure, and thereby increases the production from the reservoir. The lift gas is routed from the surface into the annulus, the volume between the casing and the tubing, and enters the tubing through a unidirectional valve that does not permit backflows.
A negative aspect of this technique is that gas lift can induce severe production flow oscillations. The oscillations caused by the dynamic interaction between injection gas in the casing and multiphase fluid (oil/gas mixture) in the tubing are a phenomenon known as *casing-heading instability*. This instability can be explained as follows. Consider a situation where there is no (or low) flow in the tubing. The bottom well pressure is high due to the weight of the fluid column in the tubing. Gas is then inserted in the annulus, but because of the high bottom hole pressure, initially it does not enter the tubing, the injection valve stays closed. The gas starts to compress in the annulus, and after some time it gets enough pressure to open the injection valve and to start to enter in the tubing. As gas enters the tubing the density of the fluid, and consequently the downhole pressure, decreases, accelerating the inflow of lift gas and increasing the production of oil. As gas continues to enter the tubing, the pressure in the annulus falls until the liquid in the tubing causes the injection valve to close, hence the tubing starts to fill with liquid and the annulus to fill with gas. Since no gas is injected into the tubing the production decreases again to the natural production of the well, which might be zero. A new cycle starts when the pressure in the annulus becomes high enough to penetrate the valve.
The fluctuating flow typically has an oscillation period of a few hours and is distinctively different from short-term oscillation caused by hydrodynamic slugging.
The casing-heading instability introduces two production-related challenges: average production is lower compared to a stable flow regime, and the highly oscillatory flow puts strain on downstream equipment. Fig. 2 shows a conceptual gas-lift production curve. The produced oil rate is a function of the flow rate of the gas injected into the well. Maximizing the performance of a gas-lifted well can be summarized as maximizing the oil production by keeping gas injected in the tubing at a certain level (decided by topside production limitations) that may be in the unstable region. In maximizing the oil production it is desired to keep the flow stable, to mantain a high processing ability topside and to have higher production capacities, as can be seen in Fig. 2.
Efforts have been exerted both in academia and industry to find optimal solutions based on control theory (Eikrem et al. [2002], Eikrem et al. [2004], Aamo et al. [2004], Eikrem et al. [2006], Havre and Dalsmo [2002], Skofteland and Godhavn [2003]).
An extended Kalman filter (Eikrem et al. [2004]) and a nonlinear observer (Aamo et al. [2004]) have been used to estimate the state of the system, and then to use them to

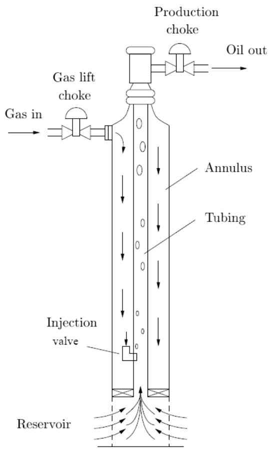


Fig. 1. A gas lifted oil well scheme.

compute the downhole pressure needed to close the control loop that stabilizes the system.

In this paper we propose a simpler solution based on a high gain observer (HGO). The measurements used are only the pressure of gas in the annulus, the pressure of the fluid at the top of the tubing and the density at the top of the tubing. The measurement of the flow through the production choke is not required, so this solution can be easily applied to multiple-well systems, where usually there is a common manifold where all the wells are connected. In these systems in case of slugging, the measurement of the total flow would not be informative about which well in the system is operating in the unstable regime.

The nonlinear observer (NLO) designed in Aamo et al. [2004] was shown to be exponentially fast, but has the assumption that one of the states is measured. We remove this assumption using an estimator extracted from the structure of the HGO. We provide also a stability analysis of the estimator. Then, the performances of the HGO is compared with the one of the combination estimator-NLO.

The paper is organized as follows: in Section 2 we present the mathematical model of the process; in Section 3 we design the observer, the estimator, and show open-loop simulation graphs; in Section 4 is presented an output feedback stabilization scheme combining the observer with a proportional integral (PI) control of the estimated down-hole pressure, and; Section 5 presents final remarks.

2. MATHEMATICAL MODEL

Commonly in the petroleum industry, the process described in Section 1 is simulated by the transient multi-phase simulator OLGA 2000 (Scandpower AS), that constitutes the state-of-the-art available nowadays.

The OLGA 2000 model developed for the gas lift well is highly accurate taking into account many aspects of the real system, and therefore is complicated and not suitable for control design purposes. Here we use a simplified model due to Eikrem et al. [2002].

The process is modelled by three states: x_1 the mass of gas in the annulus; x_2 the mass of gas in the tubing; x_3 the mass of oil in the tubing. Looking at Fig. 1, we have from mass balances

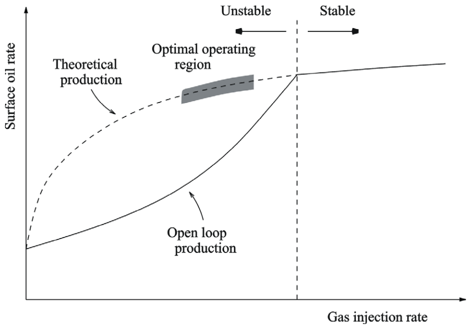


Fig. 2. Oil production as function of gas injection rate. The dotted line is the production calculated by steady state simulations assuming stable operation. The solid line is generated by dynamic simulations.

$$\dot{x}_1 = w_{gc} - w_{iv}, \tag{1}$$
$$\dot{x}_2 = w_{iv} - w_{pg}, \tag{2}$$
$$\dot{x}_3 = w_{ro} - w_{po}, \tag{3}$$

where w_{gc} is the mass flow rate of lift gas into the annulus, considered constant; w_{iv} is the mass flow rate of lift gas from the annulus into the tubing; w_{pg} is the mass flow rate of gas through the production choke; w_{ro} is the oil mass flow rate from the reservoir into the tubing; and w_{po} is the mass flow rate of produced oil through the production choke.

The flows are modelled by

$$w_{gc} = constant, \tag{4}$$
$$w_{iv} = C_{iv}\sqrt{\rho_{ai}\max\{0, p_{ai} - p_{wi}\}}, \tag{5}$$
$$w_{pc} = C_{pc}\sqrt{\rho_m \max\{0, p_{wh} - p_s\}}u, \tag{6}$$
$$w_{pg} = \frac{x_2}{x_2 + x_3} w_{pc}, \tag{7}$$
$$w_{po} = \frac{x_3}{x_2 + x_3} w_{pc}, \tag{8}$$
$$w_{ro} = C_r\left(p_r - p_{wb}\right). \tag{9}$$

C_{iv}, C_{pc} and C_r are constants, u is the production choke opening ($u(t) \in [0,1]$), ρ_{ai} is the density of gas in the annulus at the injection point, ρ_m is the density of the oil-gas mixture at the top of the tubing, p_{ai} is the pressure in the annulus at the injection point, p_{wi} is the pressure in the tubing at the gas injection point, p_{wh} is the pressure at the well head, p_s is the pressure in the separator, p_r is the pressure in the reservoir, and p_{wb} is the pressure at the well bore.

The separator pressure, p_s, is assumed to be held constant by a control system. The reservoir pressure, p_r, is assumed to be slowly varying and therefore is treated as constant. Note that the flow rates through the production valve and the injection valve are restricted to be positive.

The densities are modelled as follows

$$\rho_{ai} = \frac{M}{RT_a} p_{ai}, \tag{10}$$

$$\rho_m = \frac{x_2 + x_3}{L_w A_w}, \tag{11}$$

and the pressures as follows

$$p_{ai} = \left(\frac{RT_a}{V_a M} + \frac{gL_a}{V_a}\right) x_1, \tag{12}$$

$$p_{wh} = \frac{RT_w}{M} \frac{x_2}{L_w A_w - v_o x_3}, \tag{13}$$

$$p_{wi} = p_{wh} + \frac{g}{A_w}\left(x_2 + x_3\right), \tag{14}$$

$$p_{wb} = p_{wi} + \rho_o g L_r. \tag{15}$$

M is the molar weight of the gas, R is the gas constant, T_a is the temperature in the annulus, T_w is the temperature in the tubing, V_a is the volume of the annulus; L_a is the length of the annulus; L_w is the length of the tubing, A_w is the cross-sectional area of the tubing above the injection point, L_r is the length from the reservoir to the gas injection point, A_r is the cross-sectional area of the tubing below the injection point, g is the gravity constant, ρ_o is the density of the oil, and v_o is the specific volume of the oil. The oil is considered incompressible, so $\rho_o = 1/v_o$. The molar weight of the gas, M, is assumed constant, and the temperatures, T_a and T_w, are assumed slowly varying and therefore treated as constants.

The dynamics of the simplified model has been compared to those given by the OLGA 2000 multiphase simulator in Imsland [2002] and found to be in satisfactory agreement. It should be noted, however, that the aim of the simplified model is just to capture the casing-heading instability, and that a number of other instabilities that may occur in gas-lift oil wells are not captured as well, as for instance tubing-heading instability, tubing-reservoir interactions, hydrodynamic slugging.

3. STATE ESTIMATION

In practice, the measurements downhole in the tubing are to be considered quite unreliable because of the harsh conditions in which the sensors have to operate. Considering also that the maintenance of those sensors is basically impossible, sometimes downhole measurements are even not available at all.
In this paper we assume that only well-top measurements are available, and in particular the pressure in the annulus, that gives $y_1(t) = p_{ai}(t)$, the pressure at the top of the tubing, $y_2(t) = p_{wh}(t)$, and the density at the top of the tubing, $y_3(t) = \rho_m(t)$.

3.1 Observer

The HGO used has a particularly simple structure since it is only a copy of the simplified model, together with the correction terms. The observer uses the available process measurements for the correction of the state estimates in the simplified model.
From (12) we obtain

$$x_1 = \left(\frac{RT_a}{V_a M} + \frac{gL_a}{V_a}\right)^{-1} y_1, \tag{16}$$

from (13)

$$x_2 = \frac{M\left(L_w A_w - v_o \hat{x}_3\right)}{RT_w} y_2 \tag{17}$$

and from (11)

$$x_3 = L_w A_w y_3 - \hat{x}_2 \tag{18}$$

needed for the correction terms.
The observer equations are then

$$\dot{\hat{x}}_1 = w_{gc} - \hat{w}_{iv} + K_1\left(x_1 - \hat{x}_1\right) \tag{19}$$

$$\dot{\hat{x}}_2 = \hat{w}_{iw} - \hat{w}_{pg} + K_2\left(x_2 - \hat{x}_2\right) \tag{20}$$

$$\dot{\hat{x}}_3 = \hat{w}_{ro} - \hat{w}_{po} + K_3\left(x_3 - \hat{x}_3\right) \tag{21}$$

where K_1, K_2 and K_3 are positive constant gains, and $\hat{w}_{iv}$, $\hat{w}_{pg}$, $\hat{w}_{ro}$ and $\hat{w}_{po}$ have the same structure of (5)-(15) where instead of the states x_1, x_2 and x_3 we have the estimates $\hat{x}_1$, $\hat{x}_2$ and $\hat{x}_3$ respectively. Since (13) and (11) contain both x_2 and x_3, in (17) and (18) we use the estimates $\hat{x}_3$ and $\hat{x}_2$ instead.
At the time of writing this paper, the stability of the observer proposed is supported only by simulation results. Even if the simulations show that the observer is exponentially converging to the real states, the non smoothness of the state equations (due to the max functions and the square root terms) does not allow an immediate proof of stability. In works such as Hammouri et al. [2002], Gautier et al. [1992] the stability of high gain observers for a class of nonlinear systems has been analyzed and conditions have been given. Such results are promising, and efforts are in train to extend them to classes of nonlinear systems like the gas-lifted oil well.

3.2 Estimator for the mass of gas in the annulus

The NLO designed and analyzed in Aamo et al. [2004] estimates the state x_2 and x_3 under the assumption that the state x_1 is measured. It was shown that the NLO is exponentially fast. In this paper we use the equation (19) of the HGO to extend the NLO providing an estimator for the state x_1 based on a well-top measurement.
Considering (1) and (19), the error, $\tilde{x}_1 = x_1 - \hat{x}_1$, is governed by

$$\dot{\tilde{x}}_1 = -w_{iv} + \hat{w}_{iv} - K_1 \tilde{x}_1 \tag{22}$$

Since the mass is an inherently positive quantity and that the system is modeled by mass balances, we have

$$w_{iv}(x) \geq 0 \quad \forall x \geq 0, \tag{23}$$

$$\hat{w}_{iv} \leq C_{iv}\sqrt{\frac{M}{RT_a}\left(\frac{RT_a}{V_a M} + \frac{gL_a}{V_a}\right)\hat{x}_1} \quad \forall \hat{x} \geq 0. \tag{24}$$

Taking the Lyapunov function candidate $V = \frac{1}{2}\tilde{x}_1^2$ we have

$$\dot{V} = \tilde{x}_1 \dot{\tilde{x}}_1 \tag{25}$$

and using (23), (24) and $\tilde{x}_1 = x_1 - \hat{x}_1$

$$\dot{V} \leq C x_1 \tilde{x}_1 - \left(C + K_1\right)\tilde{x}_1^2 \tag{26}$$

where $C = C_{iv}\sqrt{\frac{M}{RT_a}\left(\frac{RT_a}{V_a M} + \frac{gL_a}{V_a}\right)}$ is a positive constant. Since x_1 is bounded, we can write $x_1 \leq \delta_1$, where δ_1 is

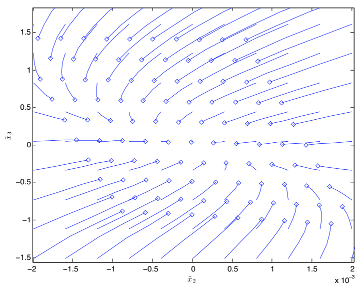


Fig. 3. Phase portrait ($u = 0.4$).

a constant. Using input-to-state stability (Khalil [2000] paragraph 4.9) we have

$$\dot{V} \leq 0 \quad \forall \tilde{x}_1 \geq \frac{C\delta_1}{(C+K_1)\,\theta} = \rho(\delta_1). \tag{27}$$

where $0 < \theta < 1$ and the function $\rho(\delta_1)$ belongs to class κ. This shows that (22) is ISS with respect to x_1, that means it is always possible to make the observer to converge exponentially fast toward the real state and to keep the error as small as desired changing the value of the gain K_1. This is a coarse result due to the assumption (23) and (24) that allowed to make (22) independent from $x_2, \hat{x}_2, x_3, \hat{x}_3$. Actually the simulations show that the error exponentially converges to 0. Anyway it gives a stability proof of the estimator for x_1, and using this in connection with the NLO forms an exponentially fast observer using only well-top measurements.
Moreover, it is possible to see the HGO as a cascade interconnection of two systems: the estimator for x_1 and the sub-observer composed by (20)-(21). The error dynamics of the sub-observer are governed by the second order system $\dot{\tilde{x}}_2 = \dot{x}_2 - \dot{\hat{x}}_2$, $\dot{\tilde{x}}_3 = \dot{x}_3 - \dot{\hat{x}}_3$. This can be considered autonomous if we fix the input u and consider x_1 given by the estimator ($x_1 = \hat{x}_1$). The qualitative behavior of such system can be easily visualized by a phase portrait in the phase plane. Considering several inputs u ($u(t) \in [0,1]$), it has been seen that the origin is a locally asymptotically stable equilibrium point (Fig. 3 shows the case $u = 0.4$).

3.3 Open-loop simulations

The numerical coefficients used for the simulations are taken from Eikrem et al. [2006] and refer to a laboratory installation where compressed air is used as the lift gas and water as the produced fluid. The production tube measures $18m$ in height and has an inner diameter of $20mm$.
In the simulations in this paper, gas is fed into the annulus at a constant rate of $w_{gc} = 0.1 \times 10^{-3} kg/s$. All the simulations are implemented in Matlab. The initial values equal steady state conditions. Values for the HGO correction gains that make the estimates converge in $3sec$ were easily found after a few tries: $K_1 = 1$, $K_2 = 1$ and $K_3 = 1$.
Fig. 4 shows that the states estimated by the HGO converge exponentially fast to the real states. Fig. 5 shows the downhole pressure calculated from the states estimates compared to the value obtained from the simulated system.

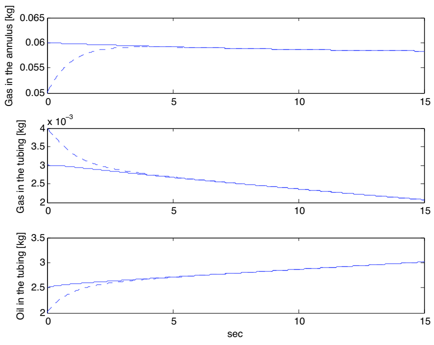


Fig. 4. States of the system. The full line are the states simulated, the dashed line are the states estimated.

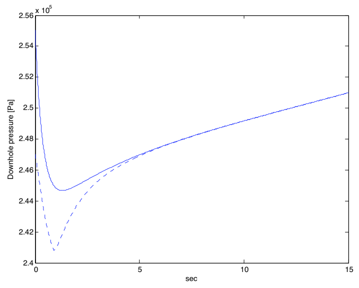


Fig. 5. Downhole pressure. The full line is the pressure simulated, the dashed line is the pressure estimated.

From Fig. 6 it is possible to see how raising the opening of the production choke from $u = 0.2$ to $u = 0.8$ (switching at $t = 200sec$) causes severe slugging in the production.
The HGO was compared with the NLO proposed in Aamo et al. [2004] in combination with the estimator proposed in Subsection *3.2*. The NLO is an exponentially fast nonlinear observer, but it is characterized by a structure more complicated than the one of the HGO. In Fig. 7 the two observers are compared, and it can be seen that the convergence is extremely quick for both (notice the time scale). The tuning of the HGO gains to obtain this result was quite straightforward ($K_1 = 20$, $K_2 = 15$ and $K_3 = 15$), thanks to its simple structure. The same cannot be said for the NLO, that required quite some time to well tune its gains.

4. FEEDBACK STABILIZING CONTROL

It can be seen from simulations that a higher rate of injection gas will stabilize the well, but not at an optimal operating point. A fixed choke opening will also stabilize the well, provided the opening of the choke is reduced until the flow from the well is stable. The reason why an increased amount of lift gas and/or a reduced choke opening gives stable flow is that the flow in the tubing changes from gravitation dominant to friction dominant flow. An improved production solution is to stabilize the well system in the unstable region with feedback control.

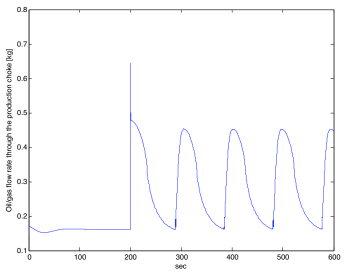


Fig. 6. Mixture oil/gas flow rate through the production choke.

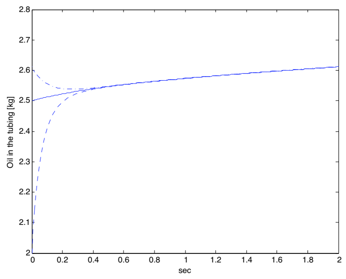


Fig. 7. Mass of oil in the tubing. The solid line is the state simulated, the dashed line is the state estimate with the HGO proposed, the dashdoted line is the state estimated with the NLO.

4.1 Controller

In Eikrem et al. [2006] it has been shown that casing-heading instability can be eliminated by stabilizing the downhole pressure using a PI control:

$$u = K_p \left(p_{wb} - p^*_{wb}\right) + K_i \int \left(p_{wb} - p^*_{wb}\right) dt \qquad (28)$$

where p_{wb} is the downhole pressure and p^*_{wb} is its desired set point, chosen usually by the operator. The means of actuation is the production choke ($u(t) \in [0, 1]$). However the downhole pressure is not an easy measurement to obtain, due to the harsh condition in which the pressure sensor has to operate. In addition, high failure rate of these sensors is reported by oil companies, and their maintenance causes costs and problems with the production of oil.
In this paper we propose an alternative to the downhole pressure measurement, and that is to replace p_{wb} with its estimate $\hat{p}_{wb}$.
The pressure $\hat{p}_{wb}$ can be obtained from (15) by using the states estimated with the observer described in the previous Section. The control structure is shown in Fig. 8.

4.2 Closed-loop simulations

The set point is chosen as $p^*_{wb} = 2.64Pa$. The controller was tuned using a combination of process knowledge and iterative simulations, and found $K_p = -0.3 \cdot 10^{-5}$ and $K_i = -0.006 \cdot 10^{-5}$.

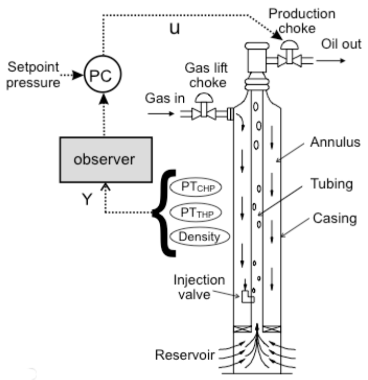


Fig. 8. Control structure for stabilization of a gas-lift well, by controlling the estimated downhole pressure.

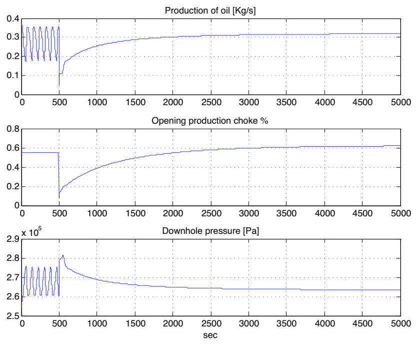


Fig. 9. Production of oil. Opening of the production choke. Downhole pressure.

Fig. 9 represents the following simulated scenario: at the beginning the gas-lift oil well is simulated in open loop with a 55% choke opening. The initial values equal steady state conditions. At time $t = 500sec$ the controller is connected to the system. After the control loop has been closed, the oscillations are quickly stabilized, even if it takes about $50min$ ($3000sec$) before the system is brought to the desired setpoint. This is roughly the time taken to build up the pressure in the annulus.
It can be seen also how the controller gently opens the production choke from 55% to 62%, this stabilizes the production of oil eliminating the casing-heading instability. Note that also the production is increased.
The downhole pressure is stabilized to $2.64Pa$.

The case of noisy measurements was also considered. Since the density is the measurement that can be more subject to uncertainty, we assume to have y_3 corrupted by white noise (zero mean, variance $100kg/m^3$ corresponding to 10% of the nominal value, Fig. 10). In Fig. 11 it can be noted that the controller successfully operates on the production choke valve so as to eliminate the oscillations in the oil production.

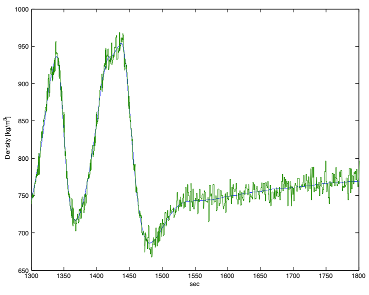


Fig. 10. Density measurement plus the uncertainty on the measurement.

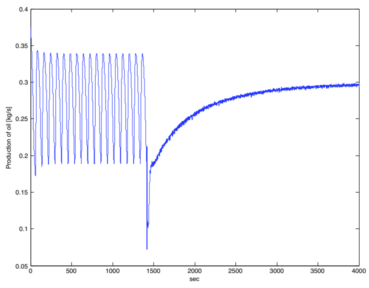


Fig. 11. Stabilization of the oil production using the downhole pressure obtained with noisy measurements.

In industry, gain scheduling is often used to adapt the gains of the PI controller as the operating point of the production valve changes. This increases the performance, but in some cases gain scheduling may also be necessary to keep the system stable since the gain values computed for a certain steady state choke opening might not have sufficient control authority to stabilize the casing-heading instability for higher steady state choke openings, as was shown in Eikrem et al. [2006]. Gain scheduling can be used also with the control structure proposed in this paper to have a better stabilizing controller in the all ranges of production choke opening, implementing hysteresis to prevent frequent change due to noise in the pressure estimate.

5. CONCLUSION

In this paper the problem of casing-heading instability that can occur in gas-lifted oil wells was considered. Casing-heading instability causes highly oscillatory oil flow rate, leading to lower production and lower processing capacity. The solution proposed was the use of a closed-loop control. The control structure presented uses the opening of the production choke as the manipulated variable and the downhole pressure as controlled variable.
Since measurements downhole in the tubing are quite unreliable, we proposed a high gain observer to estimate the states using only well-top measurements, and then using these estimates to reconstruct the downhole pressure needed. Moreover, using part of the HGO we designed an estimator for the mass of gas in the annulus and used it to extend the exponentially fast NLO proposed in Aamo et al. [2004].
The performance of the HGO was demonstrated in simulations and compared with the combination $x_1$estimator-NLO. It was seen that for basically the same performance, the HGO presents a simpler tuning capability.
The control structure proposed can also be used in a straightforward fashion as a backup strategy: it is possible to switch from a control structure based on the measured downhole pressure to the structure based on well-top measurements in case of sensor failure.
Even if the simulations showed that the observer converges exponentially fast, the stability is not yet theoretically supported. The significant nonlinearity of the model equations makes the problem nontrivial. The stability of high gain observers for a class of nonlinear systems has been analyzed in the literature, and conditions have been given. An extension of the analysis to classes of nonlinear systems like the gas-lifted oil well system represents ongoing research.

ACKNOWLEDGEMENTS

The first author wish to thank O. M. Aamo for his help with the mathematical model, and H. B. Siahaan for all the suggestions given.

REFERENCES

O.M. Aamo, G.O. Eikrem, H.B. Siahaan, and B.A. Foss. Observer design for multiphase flow in vertical pipes with gas-lift - theory and experiments. *Journal of Process Control*, 2004.

G.O. Eikrem, B.A. Foss, L. Imsland, B. Hu, and M. Golan. Stabilization of gas lifted wells. In *Proceedings of the 15th IFAC World Congress*, 2002.

G.O. Eikrem, L. Imsland, and B.A. Foss. Stabilization of gas lifted wells based on state estimation. In *IFAC*, 2004.

G.O. Eikrem, O.M. Aamo, and B.A. Foss. On instability in gas-lift wells and schemes for stabilization by automatic control. *SPE paper no. 101502*, 2006.

J.P. Gautier, H. Hammouri, and S. Othman. A simple observer for nonlinear systems - application to bioreactors. In *IEEE Transactions on Automatic Control*, 1992.

H. Hammouri, B. Targui, and F. Armanet. High gain observer based on a triangular structure. *International Journal of Robust and Nonlinear Control*, 2002.

K. Havre and M. Dalsmo. Active feedback control as the solution to severe slugging. *SPE paper no. 71540*, 2002.

L. Imsland. *Topics in nonlinear control: Output feedback stabilization and control of positive systems.* PhD thesis, NTNU, 2002.

H.K. Khalil. *Nonlinear Systems.* Prentice Hall, third edition, 2000.

G. Skofteland and J.M. Godhavn. Suppression of slugs in multiphase flow lines by active use of topside choke - field experience and experimental results. In *Proceedings of Multiphase'03*, 2003.