Constrained Optimal Control

An Application to Semiactive Suspension Systems

Preliminary Thesis

November 2005

Tina Paschedag

Course of Technical Cybernetics
Matriculation Nr.: 164120

Otto-von-Guericke University Magdeburg
Institute of Automation Technology

University of Cagliari
Department of Electrical and Electronical Engineering

Abstract

This thesis compares three different control design methods with regard to their application to a quarter car semiactive suspension model. The design methods investigated are called *optimal gain switching*, *discontinuous variable structure control* and *explicit model predictive control*. All of the three divide the state space into several regions and assign one linear/affine feedback subcontroller to each of these regions. The partition of the state space is computed off-line. During the on-line phase, the controller switches between the subcontrollers according to the current state.
The investigated methods aim to approximate optimal control laws and therefore can be called *suboptimal*.

Contents

List of Figures

Abbreviations

(C)FTOC	(constrained) finite time optimal control
(C)ITOC	(constrained) infinite time optimal control
LP	linear programming
LQ	linear quadratic
LQR	linear quadratic regulator
(e)MPC	(explicit) model predictive control
MPT	model predictive toolbox
OGS	optimal gain switching
PWA	piecewise affine
PPWA	piecewise affine on polyhedra
QP	quadratic programming
(d)VSC	(discontinuous) variable structure control

Chapter 1

Introduction

In this thesis we consider *linear systems with constraints*, which are probably the most important class in practical control applications and, thus studied a lot in the past. It is well accepted that for these systems, in general, stability and good performance can only be achieved with a non-linear control law.
We investigated three different approaches to design a non-linear controller for these type of systems, namely *optimal gain switching* (OGS), *discontinuous variable structure control* (dVSC) and *explicit model predictive control* (eMPC). All of these three methods consist of an off-line and an on-line phase. Their off-line phases divide the state space into several regions and assign linear subcontrollers in the case of OGS and dVSC or affine subcontrollers in the case of eMPC to these regions. During the on-line phase the controller switches between these subcontrollers according to the current state.
The methods presented in this thesis all intend to approximate *time optimal control laws*, and therefore can be called suboptimal control methods.

In order to compare the three design methods we applied them to a quarter-car suspension system. Therefore, we considered an *active* and a *semiactive* model of suspension systems. The design of active suspensions for road vehicles aims to optimize the performance of the vehicle with regard to comfort, road holding and rideability. In an active suspension the interaction between vehicle body, the so-called sprung mass, and wheel (nonsprung mass) is regulated by an actuator of variable length. The actuator is usually hydraulically controlled and applies between body and wheel a force that represents the control action generally determined with an optimization procedure.
In contrast to active suspensions, passive suspensions consist of dampers and springs and therefore the interaction between body and wheel is determined by their elastic constants and damping coefficients, thus is constant.
Active suspensions have a better performance than passive suspensions, but they are much more complex and cost-intensive. As a viable alternative to a purely active suspension system, the use of semiactive suspensions has been investigated a lot in the past. Such a system consists of a spring whose stiffness is constant and of a damper whose characteristic coefficient f is adjustable within an interval $[f_{\min}, f_{\max}]$ controlling the opening of a valve. The value f is determined such that an active control considered as target is approximated

as close as possible.

This thesis is structured in the following way: In Chapter 2 we recall some of the basic concepts in control theory, which are necessary to understand the presented design techniques. The quarter-car suspension model both for the active and the semiactive suspension is introduced in Chapter 3. Afterwards, the concepts of optimal gain switching, discontinuous variable structure control and explicit model predictive control are illustrated in Chapters 4, 5 and 6, respectively. Finally we present the simulation results comparing the different control design concepts in Chapter 7.

Chapter 2

Principles of Modeling and Control

This chapter presents some of the most important aspects of modeling and control theory which are necessary to understand the given problem and the different approaches to design a suitable controller.

2.1 Some Basic Concepts in Control

The control problem can in general terms be formulated as follows:

> **The Control Problem**
> Given a system $\mathcal{S}$, with measured signals y, determine a feasible control input u, so that a controlled variable z follows as closely as possible a reference signal (or setpoint) r, despite influence of disturbances w, measurement errors n, and variations in the system.

The problem is typically solved by letting u be automatically generated from y and r by a controller (or regulator) ($\mathcal{R}$).
The control problem as formulated leads to a number of issues. One is to describe the properties of the system ($\mathcal{S}$) and the disturbances (w). Another is to construct methods to calculate the regulator ($\mathcal{R}$) for wide classes of system descriptions. Mathematically, this means that $\mathcal{S}$ is described by a linear or nonlinear differential equation or by a difference equation in continuous and discrete time, respectively.

System

By a system we mean an object that is driven by a number of inputs (external signals) $u(t)$, $-\infty < t < \infty$ and as a response produces a number of outputs $y(t)$, $-\infty < t < \infty$.
From a mathematical point of view, a system is a mapping (a function) from the input u to the output y

$$u \xrightarrow{\mathcal{S}} y$$

or

$$y = \mathcal{S}(u) \tag{2.1}$$

Note that the mapping (2.1) is from the entire input $u(t)$, $-\infty < t < \infty$ to the entire output $y(t)$, $-\infty < t < \infty$. The value of the output at time t_1, i.e. $y(t_1)$, could thus very well depend on the values of the signal u at all time points t, $-\infty < t < \infty$.[1] Starting from the general description (2.1) we shall define a number of useful concepts for systems. The system $\mathcal{S}$ is

1.
   - *causal* if for every time point t_1, $y(t_1)$ only depends on $u(t)$, $-\infty < t < t_1$;
   - *non-causal* otherwise;

2.
   - *static* if for every time point t_1, $y(t_1)$ only depends on $u(t)$ for $t = t_1$;
   - *dynamic* otherwise;

3.
   - *time discrete* if u and y are defined only for a countable number of time points, kept strictly apart by a smallest, positive time distance:

     $$(y(t), u(t)) \qquad t = t_k,\ k = 0, \pm 1, \pm 2, \ldots$$

   - *time continuous* if u and y are defined for all real t over an interval or the whole of the time axis;

4.
   - *SISO (single-input–single-output)* if, for every time point t_1, $u(t_1)$ and $y(t_1)$ are scalars (real numbers);
   - *MIMO* otherwise;

5.
   - *time invariant* if the mapping (2.1) does not depend on the absolute time (i.e., if u is shifted by τ time units, then the corresponding output will also be shifted by τ time units);
   - *time variant* otherwise;

6.
   - *linear* if $\mathcal{S}$ is a linear mapping, i.e.:

     $$\mathcal{S}(\alpha_1 u_1 + \alpha_2 u_2) = \alpha_1 \mathcal{S}(u_1) + \alpha_2 \mathcal{S}(u_2); \tag{2.2}$$

   - *nonlinear* otherwise.

Let us consider some typical systems that are part of the control problem.

- *The Control Object* ("the plant") is the system to be controlled. It has
  - Inputs: A *control input* (u) which is used by us to affect the system, and a *disturbance* (w) that also affects the system.

[1] For a dynamical system one must normally define an initial state for the output to be uniquely defined. Since the initial state in our formalism is defined at $-\infty$, its influence will have "died out" at any finite time point for a wide class of systems.

  - Outputs: A *controlled variable* (z) which is the variable that should be controlled to desired values, and a *measured output* (y) which contains all measured signals from the plant. The signal y may include some disturbances as well.

- *The Controller or Regulator* is also a system with
  - Inputs: *The Reference signal or Setpoint* (r), which is the desired value of the controlled variable z. The measured output from the controlled object, y, is also an input to the controller.
  - Outputs: The controller output is the control signal u to the controlled object.

- *The Closed Loop System* is the system obtained when the control loop is closed. This system has
  - Inputs: Reference signal and disturbance signal.
  - Outputs: *The Control Error* ($e = r - z$), the difference between the desired and actual values of the controlled variable. Moreover, it is natural to consider also the control signal u from the controller as an output from the closed loop system, since it may be necessary to study its properties as a function of the inputs.

In the following sections of this chapter we give an introduction on how to describe systems like the plant and the controller. In a first step we illustrate modeling of damping systems, which may be described by ordinary linear differential equations. Later on we recapitulate some fundamentals of control theory such as the state space model, system properties like stability, controllability, observability and at the end we present two fundamental methods of the control unit design, namely pole placement and the linear quadratic regulator.

2.2 Modeling of Mechanical Systems

2.2.1 Introduction

By a mathematical model we mean a description of the system (here we concentrate on the plant) where relationships between the model's variables and the signals are expressed in mathematical terms. Model building naturally leads to differential or difference equations for continuous time and discrete time models respectively. In this section we will describe how to derive the differential equations which describe mechanical systems, e.g. suspension systems. Later on we will discuss the formal, mathematical aspects of such equations.

2.2.2 Describing Mechanical Systems

There are three elements that make up simple mechanical systems:

1. spring elements
2. damper elements

3. mass or inertia elements

The word "spring" may bring to mind a coil of a steel wire, but the notion of a spring element is much more general in the modeling of mechanical systems. Anything the exhibits significant "stretch" or deformation under an applied load may be regarded as a spring. Damper elements are used to represent frictional effects. In lumped-element models, mass elements simply represent the mass of some portion of the physical system, centered at a point.

Translational Springs

Formally the spring element is a mechanical element that links two endpoints in a physical system and has some functional relationship between the relative displacement of the two endpoints and the force transmitted through the element. For the general spring element, we can say that the force is given by a function of the "deformation", the extension or relative displacement of the two endpoints. Furthermore, we assume that the ideal spring element is both massless and frictionless.
The force-deformation relation for a linear spring is known as *Hooke's law*, often expressed as

$$F_s = kx \tag{2.3}$$

where F_s is the force transmitted through the spring, k is a constant describing the stiffness of the spring and x is the extension of the one endpoint while the other one is fixed. Note that the force is considered positive when directed opposite to the displacement. If the assumption that one endpoint is fixed is not true any longer the relationship between the force and the deformation can be generalized like this applying the relative extension:

$$F_s = k(x_{near} - x_{far}) \tag{2.4}$$

where x_{near} represents the displacement of the endpoint at which the force is being determined and x_{far} represents the other endpoint.

Translational Dampers

Analog to the spring, the damper element links two points in a physical system and has a functional relationship between the relative velocity of the two endpoints and the force transmitted through the element. This damping force, that depends on the velocity of the two endpoints, is given by

$$F_d = c\,\frac{d}{dt}(x_{near} - x_{far}) = c(\dot{x}_{near} - \dot{x}_{near}) \tag{2.5}$$

where F_d is the force and c is the damping coefficient.

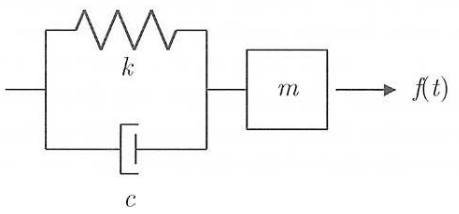

Figure 2.1: A spring-mass-damper system with an applied force [6]

Mass Elements in Translational Motion

The force that is effective on the mass is related to the second time derivative of the displacement of the mass with respect to a "fixed" frame of reference (within the context of the theory of relativity, we would refer to an unaccelerated or *inertial* frame of reference). The relation between force and acceleration is known as **Newton's Second Law**:

$$F = ma = m\frac{d^2x}{dt^2} = m\ddot{x} \tag{2.6}$$

where F is the force which is effective on the mass m and $a = \frac{d^2x}{dt^2}$ is the acceleration of the latter.

The Interrelationship between Forces in different Elements in a System

Let us consider a specific configuration of a spring, a mass and a damper, as shown in Fig. 2.1. Newton's second law states that the sum of the forces acting on a body is equal to the mass of the body times the body's acceleration with respect to a fixed frame of reference. A common technique for approaching mechanical problems and applying Newton's second law involves the use of free-body diagrams (FBD's). A box is drawn around some portion of the system which is then considered as "free body", and the forces acting on this body are summed (vectorially). The resulting sum is then equated to the product of the mass inside the drawn box, and the acceleration of this mass. This is illustrated in Fig. 2.2. Drawing the box around the mass and applying Newton's second law, we have

$$\sum F = ma \tag{2.7}$$

Because the spring and the damper elements give rise to forces that oppose motion, summing them vectorially yields

$$f(t) - F_s - F_d = ma \tag{2.8}$$

or, in terms of their respective constitutive equations,

$$f(t) - k(x_2 - x_1) - c(\dot{x}_2 - \dot{x}_1) = m\ddot{x}_2 \tag{2.9}$$

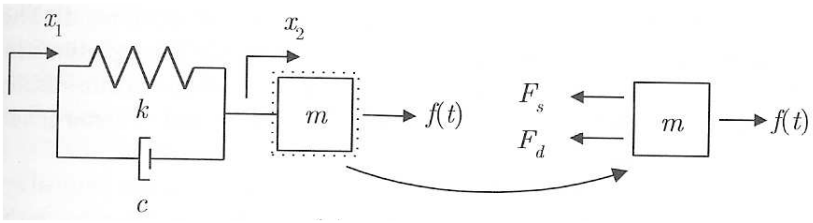

Figure 2.2: Newton's second law applied using the FBD approach [6]

where $f(t)$ is an external force acting on the system. Rearranging the above yields the governing equation for the spring-mass-damper system of Fig. 2.2:

$$m\ddot{x}_2 + c\dot{x}_2 + kx_2 = c\dot{x}_1 + kx_1 + f(t) \tag{2.10}$$

Note that there is another method called *d'Alembert's principle* which draws the "box" around a massless node. Because of this fact the sum of the forces acting on the node is always zero.

2.3 Models of Dynamic Systems

2.3.1 The Concept of State and State Space Models

The system's state

As almost all controlled objects are dynamic systems, their output depends on all earlier input values. This leads to the fact that it is not enough to know $u(t)$ for $t \geq t_0$ in order to be able to calculate the output $y(t)$ for $t \geq t_0$. We need information about the system and therefore we define the state of a system.

Definition 1. (The system's state) With the state of a system we denote an amount of information such that with this state and the knowledge of $u(t)$, $t \geq t_0$, we can calculate $y(t)$, $t \geq t_0$.

This definition is well in line with the everyday meaning of the word "state".
It is also obvious from the definition of state that this concept plays a major role in the simulation of dynamic systems. The state is exactly the information that has to be stored and updated during the simulation in order to be able to calculate the output.

The State Space Model in general

In Section 2.2.2 we have shown how the physical modeling of a damping system yields a differential equation. After the modeling it is advantageous to convert the system's model

into another form, called state space form, for which exist powerful tools for simulation and controller design. Therefore we start from a general ordinary differential equation

$$g\left(\tilde{x}^{(n)}(t), \tilde{x}^{(n-1)}(t), \ldots, \tilde{x}(t), u^{(m-1)}(t), \ldots, u(t)\right) = 0 \tag{2.11}$$

where

$$\tilde{x}^{(k)}(t) = \frac{d^k}{dt^k}\tilde{x}(t)$$

and $g(\,\cdot\,, \cdot, \ldots, \cdot\,)$ is an arbitrary, vector-valued, nonlinear function.
As we now want to obtain the state space model for the system, it is necessary to transform the differential equation of order n into a system of n first order differential equations by introducing a number of internal variables as follows

$$x_1(t) = \tilde{x}(t), \ldots, x_n(t) = \tilde{x}^{(n-1)}$$

Introducing a vector notation

$$x(t) = \begin{pmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{pmatrix}$$

we can write the system of first-order differential equations as

$$\dot{x}(t) = f(x(t), u(t)) \tag{2.12}$$

With $\dot{x}(t)$ we denote the first derivative with respect to time. In (2.12) $f(x, u)$ is a vector function with n components. The outputs of the model can then be calculated from the internal variables $x_i(t)$ and the inputs $u_i(t)$:

$$y(t) = h(x(t), u(t)) \tag{2.13}$$

For the system consisting of (2.12) and (2.13) the vector $x(t)$ is the *state vector* of the model and its components $x_i(t)$ are *state variables*. The dimension of $x(t)$ n is called the *model order*.
In conclusion the general state space model (continuous time) is

$$\dot{x}(t) = f(x(t), u(t)) \tag{2.14a}$$
$$y(t) = h(x(t), u(t)) \tag{2.14b}$$

where the state $x(t)$ is an n-dimensional column vector, the input $u(t)$ is an m-dimensional column vector and the output $y(t)$ is a p-dimensional column vector. In an analogous mode the state space model in discrete time can be written as

$$x(t_{k+1}) = f(x(t_k), u(t_k)) \qquad k = 0, 1, 2, \ldots \tag{2.15a}$$
$$y(t_k) = h(x(t_k), u(t_k)) \tag{2.15b}$$

where the state at time t_k $x(t_k)$ is an n-dimensional column vector, the input at time t_k $u(t_k)$ is an m-dimensional column vector and the output at time t_k $y(t_k)$ is a p-dimensional column vector.

Linear State Space Models

If the models (2.14) and (2.15) are linear the functions $f(x,u)$ and $h(x,u)$ are linear in x and u, i.e.:

$$f(x,u) = Ax + Bu \tag{2.16a}$$
$$h(x,u) = Cx + Du \tag{2.16b}$$

Here the matrices have the following dimensions

$$A : n \times n \qquad B : n \times m$$
$$C : p \times n \qquad D : p \times m$$

Considering linear, time invariant models the state space model in continuous time is represented by

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.17a}$$
$$y(t) = Cx(t) + Du(t) \tag{2.17b}$$

with its corresponding discrete time linear model given by

$$x((k+1)T) = Fx(kT) + Gu(kT) \tag{2.18a}$$
$$y(kt) = Cx(kT) + Du(kT) \tag{2.18b}$$

Here we assume that the signals are measured at equidistant times $(0, T, 2T, \ldots)$, separated by the *sampling interval* T.
It is possible to compute a state space model in discrete time like (2.18) that is related directly to the state space model in continuous time (2.17). If the input u is piecewise constant according to

$$u(t) = u(kT), \qquad kT \le t < (k+1)T$$

the matrices F and G in (2.18a) are given by

$$F = e^{AT}, \qquad G = \int_0^T e^{A\tau} B \, d\tau \tag{2.19}$$

In the next paragraph we will introduce transfer functions and their connection to state space models.

2.3.2 Transfer Functions

Another concept to analyze the relationship of system inputs and outputs is to study the transfer behavior. Applying Laplace transform to (2.15) assuming zero initial conditions we get

$$sX(s) = AX(s) + BU(s) \tag{2.20a}$$
$$Y(s) = CX(s) + DU(s) \tag{2.20b}$$

Eliminating the state $X(s)$ from (2.20) yields

$$Y(s) = G(s)U(s) \tag{2.21}$$

where $G(s)$ is a $p \times m$ matrix called the *transfer function*. The system (2.17) corresponds to a transfer function $G(s)$

$$G(s) = C(sI - A)^{-1}B + D \tag{2.22}$$

If u and y are scalars ($p = m = 1$), $G(s)$ is a rational function:

$$G(s) = \frac{b_0 s^n + b_1 s^{n-1} + \cdots + b_n}{s^n + a_1 s^{n-1} + \cdots + a_n} \tag{2.23}$$

The values of s for which $G(s) = 0$ are called *zeros*, while values of s for which the denominator of $G(s)$ equals 0 are called *poles*.
Normally the poles of G are identical to the eigenvalues of the matrix A in (2.17). Some eigenvalues may, however, correspond to dynamics that cannot be excited or observed from the input-output behavior. Such eigenvalues are not poles of the transfer function of the system because they are canceled during the computation of $G(s)$.
The poles and zeros play an important role for the stability analysis of the system that will be discussed later on.
If in the discrete time case u and y have the z transforms $U(z)$ and $Y(z)$, respectively, the input-output behavior will be determined by

$$Y(z) = G(z)U(z) \tag{2.24}$$

where $G(z)$ is a $p \times m$ matrix called the (discrete time) transfer function. Given (2.18) $G(z)$ results from

$$G(z) = C(zI - A)^{-1}B + D \tag{2.25}$$

The definition of poles and zeros of the transfer function remains the same as in the continuous time case. The complex variables z and s are related by the equation

$$z = e^{Ts} \tag{2.26}$$

on the basis of which it is possible to map poles from the s-plane into the z-plane.

2.3.3 Stability

Stability is fundamental for control systems and there are a number of different stability concepts, of which we will recall some in this section. We will focus on input-output stability and Lyapunov stability. In a first step we will look at stability for linear systems, later on in this section we will extend the stability concept to non-linear systems.

Input-Output Stability

To give a definition of input-output stability we first define the concept of *gain* for a linear mapping $y = Ax$. The norm of the mapping A is defined as how much larger the norm of y can be, compared to the norm of x:

$$|A| = \sup_{x \neq 0} \frac{|y|}{|x|} = \sup_{x \neq 0} \frac{|Ax|}{|x|}$$

We may interpret $|A|$ as the "gain" of the mapping.
With the *input-output stability* is meant that an input with bounded norm must lead to an output with bounded norm. In terms of the general concept of *gain* the definition is simply:

Definition 2. (Input-Output Stability) A system is **input-output stable** if it has a finite gain.

As we already mentioned in Subsection 2.3.2 stability of the system depends on the location of the poles of the transfer function in the complex plane. We repeat the following well known theorem for continuous time systems without proving it.

Theorem 1. ***(Input-Output Stability of Linear Systems)*** *A linear, time invariant system is input-output stable if and only if its poles lie in the left half plane, not including the imaginary axis.*

With (2.26) the left half plane can be mapped into the unit circle, so that a discrete time system is input-output stable if and only if its poles lie inside the unit circle.

Lyapunov stability

Lyapunov stability is the stability of the system's solutions which are solutions to differential equations. It is defined in the following way

Definition 3. Let $x^*(t)$ be a solution of $\dot{x}(t) = f(x(t))$ with the initial state $x^*(0)$. This solution is said to be **stable** if for each $\varepsilon > 0$ there is a δ such that $|x^*(0) - x(0)| < \delta$ implies that $|x^*(t) - x(t)| < \varepsilon$ for all $t > 0$. ($x(t)$ is the solution that belongs to the initial state $x(0)$.) It is said to be **asymptotically stable** if it is **stable** and there exists a δ such that $|x^*(0) - x(0)| < \delta$ implies that $|x^*(t) - x(t)| \to 0$ as $t \to \infty$. A solution that is not stable is called **unstable**.

Assuming the initial time $t_0 = 0$ the solution of (2.17) is

$$x(t) = e^{At}x(0) + \int_0^t e^{At}Bu(\tau)d\tau \tag{2.27}$$

Comparing two different solutions with the same input but different initial values it is easy to find that stability does neither depend on the input nor on the initial value. The stability is thus a *system property* of linear systems. We can therefore talk about *asymptotically stable systems* instead of asymptotically stable solutions. The stability of linear systems can be analysed with the following criterion:

Theorem 2. *A linear system in state space form (2.17) is* ***asymptotically stable*** *if and only if all eigenvalues of the matrix A lie inside the left half plane. If the system is stable, all the eigenvalues are inside the left half plane or on the imaginary axis.*

If the system matrix A has an eigenvalue on the imaginary axis the system can be either stable or unstable (but never asymptotically stable). It can be shown that the stability then depends on the number of linearly independent eigenvectors to the corresponding eigenvalue. If the number is equal to the multiplicity of the eigenvalue, the system is stable, otherwise not.
Analogously to the input-output stability the criterion can be used for discrete time systems if one replaces 'left half plane' by 'unit circle'.

Non-linear Systems

Considering non-linear systems it is often sufficient to consider an *equilibrium* instead of explicit solutions to the system's equations. For a system described by (2.14a) an equilibrium is given by constant vectors u_0 and x_0 such that

$$f(x_0, u_0) = 0 \tag{2.28}$$

The equilibrium is also sometimes called *singular point* or *stationary point*. Nonlinear systems can have several distinct equilibria.
Stability, instability and asymptotic stability were defined in Definition 3 which can be easily applied to equilibria.
Suppose we study the system

$$\dot{x} = f(x(t), u_0) \tag{2.29a}$$

$$\text{where } f(x_0, u_0) = 0 \tag{2.29b}$$

with the equilibrium (x_0, u_0). Further, assume that there is a function V with the properties

$$V(x_0) = 0; \quad V(x) > 0,\ x \neq x_0; \quad V_x(x)f(x) \leq 0 \tag{2.30}$$

(Here V_x is the row vector $(\partial V/\partial x_1, \ldots, \partial V/\partial x_n)$.) V can be interpreted as a generalized distance from x to the point x_0. This generalized distance decreases for all solutions of $\dot{x} = f(x)$, since

$$\frac{d}{dt}V(x(t)) = V_x(x(t))\dot{x}(t) = V_x(x(t))f(x(t)) \leq 0$$

A function V satisfying (2.30) in some neighborhood of the equilibrium x_0 is called a (local) *Lyapunov function*.
By adding some requirements to the Lyapunov function properties we get the following stability test.

Theorem 3. *An equilibrium x_0 of the system $\dot{x} = f(x)$ is globally asymptotically stable if there exists a function V, satisfying (2.30) for all values of x, and in addition satisfying*

$$V_x(x)f(x) < 0,\ x \neq x_0 \tag{2.31a}$$

$$V(x) \to \infty, |x| \to \infty \tag{2.31b}$$

In many cases it is not easy to verify the inequality in (2.31a), therefore exists an extension of Theorem 3.

Theorem 4. *An equilibrium x_0 of the system $\dot{x} = f(x)$ is globally asymptotically stable if it is possible to find a function V, which*

1. *satisfies (2.30) for all values of x*
2. *satisfies (2.31b)*
3. *has the property that no solution of the differential equation (except $x(t) = x_0$) lies entirely in the set $V_x(x)f(x) = 0$*

In the theorems we have looked at so far, the properties (2.30) have been valid in the whole state space and we have been able to prove global asymptotic stability. The situation is not often this favorable. Instead, V satisfies (2.30) only for the x-values belonging to some subset N of the state space. In this case exists another theorem to decide wether an equilibrium is stable or not.

Theorem 5. *Assume that there exists a function V and a positive number d such that (2.30) is satisfied for the system (2.29a) in the set*

$$M_d = x : V(x) \leq d$$

Then a solution starting in the interior of M_d remains there. If, in addition, no solutions (except the equilibrium x_0) remain in the subset of M_d where $V_x(x)f(x) = 0$, then all solutions starting in the interior of M_d will converge to x_0.

2.3.4 Controllability and Observability

The concepts of controllability and observability describe how state variables in the state space are influenced by inputs and how they show up in the output. They are also important for the understanding of what happens when factors are canceled in the transfer function as well as for the control synthesis with state feedback.

Definition 4. (Controllability) The state x^* is said to be **controllable** if there is an input that in finite time gives the state x^* from the initial state $x(0) = 0$. The system is said to be controllable if all states are controllable.

Definition 5. (Observability) The state $x^* \neq 0$ is said to be **unobservable** if, when $u(t) = 0$, $t \geq 0$ and $x(0) = x^*$, the output is $y(t) \equiv 0$, $t \geq 0$. The system is said to be **observable** if it lacks unobservable states.

The criteria for controllability and observability can be seen in the literature. An important usage of the controllability and the observability concepts is to describe when the eigenvalues of the matrix A can be modified using feedback. We will look at this more in detail in Section 2.4.

Stabilizability and Detectability

It can be shown that if a system is not controllable, the controllable modes (eigenvalues) can be modified, but the uncontrollable cannot. If a certain mode is unstable (the definition will be given in Subsection 2.3.3) it is particularly interesting to know if it is controllable, and thereby can be modified. The following concepts therefore are useful.

Definition 6. (Stabilizability, Detectability) A system (A, B, C) (see (2.16)) is said to be **stabilizable** if there exists a matrix K, so that $A - BK$ is stable. It is said to be **detectable** if there exists a matrix L, so that $A - LC$ is stable.

A controllable system is obviously always stabilizable, just as an observable system is always detectable. These definitions are important for the concepts of state feedback.

2.4 Control Design Methods

In this section we introduce two design concepts for the controller which are fundamental and play an important role for the different approaches we will compare later on. In this section we assume that the state is completely measurable, the state space model we consider is thus:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.32a}$$
$$y(t) = x(t) \tag{2.32b}$$

For such systems a constant state feedback is often a satisfying control design:

$$u(t) = -Kx(t) \tag{2.33}$$

where K is a $m \times n$-matrix. The two methods described below are two different ways to determine the matrix K in (2.33).

2.4.1 Pole placement

Considering a system with constant state feedback, i.e. applying (2.33) to (2.32a) we get

$$\dot{x}(t) = (A - BK)x(t) \tag{2.34}$$

Here the eigenvalues of the matrix $A - BK$ are the poles of the closed loop system. Under these conditions there exists an important theorem.

Theorem 6. *If the pair (A, B) is controllable, the eigenvalues of the matrix $A - BK$ can be assigned arbitrarily by choosing an appropriate matrix K.*

For system with a scalar input signal, K is thus a vector, its elements can be found by comparing coefficients of the characteristic polynomial of the matrix $A - BK$ and the polynomial of the designated eigenvalues.
Even if the state is not entirely known, it is possible to construct a state estimate $\hat{x}$ such that $u(t) = -K\hat{x}$ retains similar pole assignment and closed-loop properties. One can achieve this by designing a state estimator (or observer).
This technique can be applied both to continuous and discrete time models.

2.4.2 Linear Quadratic Regulator

A linear quadratic regulator (LQR) is called *optimal* because the control vector u_{opt} is chosen in such a way that it minimizes or maximizes a function called the *performance index*. We consider again the system (2.32), for which we choose the following quadratic performance index

$$J = \int_0^\infty (x(t)^T Q x(t) + u(t)^T R u(t))\, dt \tag{2.35}$$

where

$$\begin{array}{lll} x^T Q x \geq 0 & \forall\, x \,\epsilon\, \mathbb{R}^n & Q \text{ positive semidefinite} \\ x^T R x > 0 & \forall\, x \,\epsilon\, \mathbb{R}^n & R \text{ positive definite} \end{array}$$

It can be seen in the literature that the optimal control law u^* corresponding to the given performance index can be computed by solving a Riccati-Equation.

Theorem 7. *If the pair (A, B) of (2.32a) is stabilizable, the optimal control law for the performance index (2.35) is*

$$u^* = -R^{-1} B^T P\, x(t) \tag{2.36}$$

where the $n \times n$ matrix P solves the algebraic Riccati-Equation

$$PA + AP - PBR^{7-1}B^T P + Q = 0 \tag{2.37}$$

The closed loop system is then determined by

$$\dot{x}(t) = (\underbrace{A - BR^{-1}B^T P}_{A_{cl}})x(t) \tag{2.38}$$

In the following we want to analyze the closed loop stability of the optimal control system. Therefore we recall a property of positive semidefinite matrices:

Remark 1. Every symmetric positive semidefinite matrix $Q \,\epsilon\, \mathbb{R}^{n\times n}$ can be factorized

$$Q = EE^T$$

The following theorem is a criterion for the stability of an LQR-system.

Theorem 8. *If the pair (E^T, A) is detectable the solution P of (2.37) is unique and all eigenvalues of A_{cl} are located in the left half plane.*

Thus, if the pair (E^T, A) is detectable, the stability of the closed loop system is assured.

Chapter 3

The Quarter-Car Suspension Model

In this chapter we introduce the idea of semiactive suspension systems and describe the dynamical models of suspension systems on the basis of which we compare the three different control approaches later on.

3.1 Introduction

The design of active suspensions for road vehicles aims to optimize the performance of the vehicle with regard to comfort, road holding and rideability.
In an active suspension there are no passive elements, such as dampers and springs. The interaction between vehicle body and wheel is regulated by an actuator of variable length. The actuator is usually hydraulically controlled and applies between the body and the wheel a force that represents the control action generally determined with an optimization procedure.
Active suspension systems have better performance than passive suspensions. However, active suspensions are rather complex, since they require several components such as actuators, servovalves, high-pressure tanks for the control fluid, sensors for detecting the system's state, etc. The associated power, that must be provided by the vehicle engine, may reach several 10 kW depending on the required performance. Furthermore, these suspension systems have a high cost.
As a variable alternative to a purely active suspension system, the use of semiactive suspensions has been considered by many researchers. Semiactive suspension systems consist of a spring whose stiffness is constant and of a damper whose characteristic coefficient f can be changed within an interval $[f_{\min}, f_{\max}]$ controlling the opening of a valve. The time required to update f is less than 10^{-2} s.
A semiactive suspension is a valid engineering solution when it can reasonably approximate the performance of the control law of an active suspension. In fact, a semiactive suspension requires a low power controller that can be easily realized at a lower cost than that of a fully active one. In general, a semiactive suspension design consists of two phases:

- Determine a suitable active law u_{act} to be considered as a target.
- Design the semiactive suspension so that its control law u_{sem} approximates the target law as close as possible.

In the following chapters we will present different algorithms to obtain the target control law u_{act}. At each sampling instant k the controller should select the damper coefficient f in the set of $[f_{\min}, f_{\max}]$ so as to minimize $(u_{\text{act}}(k) - u_{\text{sem}}(k))^2$.

3.2 Dynamical Model of the Suspension System

We consider two different dynamical models of a quarter-car suspension system. The first one is a two-degrees of freedom fourth-order model as in [11]. The second-model that is taken under consideration is a one-degree of freedom second order model that neglects the dynamics of the tire [7].
Since the reduced model does not describe the interaction of the tire with the suspended mass and the ground, it cannot be used to evaluate features like road holding and ride-ability. But it is possible to give a geometrical representation of the computed regions of the two-dimensional system in the state space, which is an important aspect in terms of the comparability of the different design techniques. This is the main reason why we deal with two different models.
In general the state space model of the considered systems are represented by the following linear system:

$$\dot{x}(t) = A\,x(t) + B\,u(t) + L\,x_0(t) \tag{3.1a}$$
$$y(t) = C\,x(t) \tag{3.1b}$$

where $x(t) = [x_1(t)\ x_2(t)\ x_3(t)\ x_4(t)]^T$ is the state, $x_0(t)$ is a disturbance and $u(t)$ is the control force. Since in physical systems all variables are bounded the control force that has to be found in the following chapters is subject to the constraint:

$$|u(t)| \leq u_{\max} \tag{3.2}$$

The physical meaning of the states and the structure of the constant matrices A, B, C and L will be specified below.

3.2.1 The Two-Degrees of Freedom Model

For the two-degrees of freedom model, which is depicted in Figure 3.1, we utilize the following notation:

M_w is the nonsprung mass consisting of the wheel and its moving parts;

M_s is the sprung mass, i.e. the part of the whole body mass and the load mass pertaining to only one wheel;

$x_1(t)$ is the nonsprung mass displacement at time t with respect to a fixed reference;

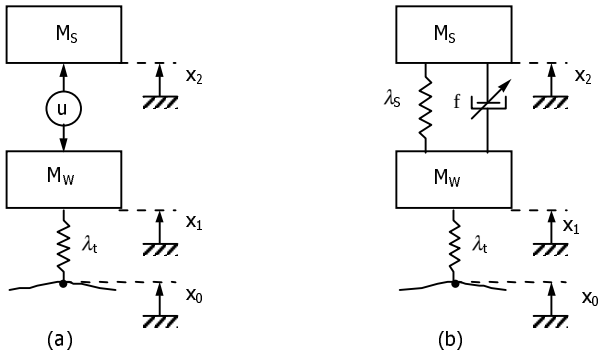

Figure 3.1: Model of the two-degrees of freedom model (a) active (b) semiactive suspension

$x_2(t)$ is the sprung mass displacement at time t with respect to a fixed reference;

$x_3(t) = \dot{x}_1(t)$ is the velocity of the nonsprung mass at time t;

$x_4(t) = \dot{x}_2(t)$ is the velocity of the sprung mass at time t;

$u_{\mathbf{act}}(t)$ is the active control force at time t;

$x_0(t)$ is the function representing the disturbance, which simulates the longitudinal profile of the road;

λ_t is the elastic constant of the tire, whose damping characteristics have been neglected. This is in line with almost all researchers who have investigated synthesis of active suspensions for motor vehicles as the tire damping is minimal;

λ_s is the elastic constant of the spring of the semiactive suspension;

$f(t)$ is the adjustable damper coefficient of the semiactive suspension at time t.

In the linear equation (3.1a) the constant matrices A, B and L of the fourth-order model have the following structure:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{\lambda_t}{M_w} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 0 \\ -\frac{1}{M_w} \\ \frac{1}{M_s} \end{bmatrix} \qquad L = \begin{bmatrix} 0 \\ 0 \\ \frac{\lambda_t}{M_w} \\ 0 \end{bmatrix}$$

The disturbance $x_0(t)$ representing the longitudinal road profile, which also depends on the vehicle speed, is assumed to be stochastic and may be characterized by its power spectral density (PSD) distribution function. In our case the road roughness characteristics are expressed by a signal whose PSD distribution function is

$$\Phi(\omega) = \frac{cV}{\omega^2 + \alpha^2 V^2} \tag{3.3}$$

where $c = (\sigma^2/\pi)\alpha$. Here σ^2 denotes the road roughness variance and V the vehicle speed, whereas the coefficients c and α depend on the type of the road's surface.
The signal $x_0(t)$, whose PSD is given by (3.3), may be obtained as the output of a linear filter expressed by the differential equation

$$\dot{x}_0(t) = -\alpha V x_0(t) + w(t). \tag{3.4}$$

The control law we will design in the following chapters requires the knowledge of the system's state x. Since not every component of $x(t)$ is directly measurable, we reconstruct the state through an appropriate state observer. To do this, we choose a suitable matrix C for the output equation (3.1b). If we assume

$$C = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

which corresponds to measuring the suspension deformation and the sprung mass velocity, the system (3.1) is completely observable and completely controllable.
Since both the concepts of OGS and explicit MPT make use of a discrete time state space model, we choose a sampling interval T and discretize the model (3.1)

$$x(k+1) = Gx(k) + Hu(k) + Ww(k) \tag{3.6}$$

where

$$G = e^{AT}, \quad H = \left(\int_0^T e^{A\tau} d\tau \right) B, \quad W = \left(\int_0^T e^{A\tau} d\tau \right) L.$$

It is well known [20] that a system that is observable and controllable in the absence of sampling maintains these properties after the introduction of sampling if and only if, for every eigenvalue of A for the continuous time control system, the relationship

$$Re\{\lambda_i\} = Re\{\lambda_j\} \tag{3.7}$$

implies

$$Im\{\lambda_i - \lambda_j\} \neq \frac{2n\pi}{T}, \quad n = \pm 1, \pm 2, \ldots \tag{3.8}$$

The problem at hand results in the following set of eigenvalues:

$$\left\{ 0, 0, \sqrt{-\frac{\lambda_t}{M_w}}, -\sqrt{-\frac{\lambda_t}{M_w}} \right\}.$$

Under these conditions it is necessary to choose a sampling period T, such that:

$$T \neq n\pi \sqrt{\frac{M_w}{\lambda_t}}. \tag{3.9}$$

As we now want to show how to obtain the semiactive control law u_{sem} once determined the active target law u_{act}, we consider the sampled model.
The effect of the semiactive suspension which is composed of a spring and a damper with an adjustable damper coefficient (refer to Fig. 3.1(b)) leads to the following semiactive control law:

$$u_{\text{sem}}(k) = -\underbrace{[-\lambda_s \quad \lambda_s \quad -f(k) \quad f(k)]}_{K} x(k) \tag{3.10}$$

Note that, as f may vary, $u_{\text{sem}}(k)$ is both a function of f and of $x(k)$.
In general, f can only take values in a real set $[f_{\min}, f_{\max}]$. We propose to choose at each step k the value of $f(k)$ to minimize the difference

$$F[f, x(k)] = (u_{\text{act}}(k) - u_{\text{sem}}(k))^2. \tag{3.11}$$

Let us first assume $x_3(k) \neq x_4(k)$, then the value $f^*(k)$ such that $F[f^*(k), x(k)] = 0$ is

$$f^*(k) = -\frac{u_{\text{act}}(k) + \lambda_s \, \Delta x(k)}{\Delta v(k)} \tag{3.12}$$

where $\Delta x(k) = x_2(k) - x_1(k)$ is the suspension deformation and $\Delta v(k) = x_4(k) - x_3(k)$ is its rate of change.
As the admissible values of f lie in the interval $[f_{\min}, f_{\max}]$ the adjusted damper coefficient becomes

$$f(k) = \min_{f \in [f_{\min}, f_{\max}]} \arg F[f, x(k)] = \begin{cases} f_{\max} & \text{if} \quad f^*(k) > f_{\max} \\ f^*(k) & \text{if} \quad f^*(k) \in [f_{\min}, f_{\max}] \\ f_{\min} & \text{if} \quad f^*(k) < f_{\min} \end{cases} \tag{3.13}$$

When $x_3(k) = x_4(k)$, regardless to the values of f, the damper does not give any contribution to $u_{\text{sem}}(k)$. Thus, in this case we assume $f(k) = f_{\max}$, which we choose also as the initial value for the damper coefficient $f(0) = f_{\max}$.

3.2.2 The One-Degree of Freedom Model

For the one-degree of freedom model of the suspension system, which is schematized in Figure 3.2, we introduce the following notation according to the previous model:

M_s is the sprung mass;

$x_1(t)$ is the sprung mass displacement at time t with respect to a fixed reference;

$x_2(t) = \dot{x}_1(t)$ is the velocity of the sprung mass at time t;

$u_{\mathbf{act}}(t)$ is the active control force at time t;

λ_s is the elastic constant of the spring of the semiactive suspension;

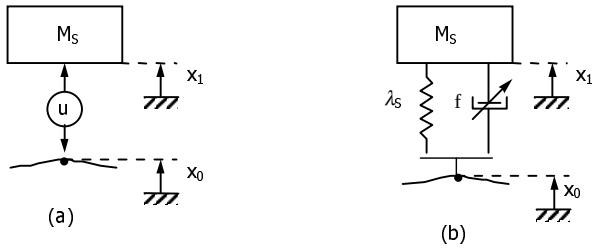

Figure 3.2: Model of the one degree of freedom model (a) active suspension (b) semiactive suspension

$f(t)$ is the adjustable damper coefficient of the semiactive suspension at time t.

The matrices A and B of the state space model (3.1a) have the following structure:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1/M_s \end{bmatrix}$$

We will simulate disturbances caused by the road profile only with the two-degrees of freedom model, therefore we do not specify the matrix L here. Like for the four dimensional model we sample the system (3.1)

$$x(k+1) = Gx(k) + Hu(k). \tag{3.14}$$

Analogously to the fourth order model, the effect of the suspension depicted in Fig. 3.2(b) is equivalent to that of a control force

$$u_{\text{sem}}(k) = -[\lambda_s \quad f(k)]\, x(k) \tag{3.15}$$

Minimizing $(u_{\text{act}}(k) - u_{\text{sem}}(k))^2$ under the assumption $x_2(k) \neq 0$, in this case results in a damper coefficient

$$f^*(k) = -\frac{u_{\text{act}} - \lambda_s\, x_1(k)}{x_2(k)} \tag{3.16}$$

As the damper coefficient has to be chosen out of the set $[f_{\min}, f_{\max}]$, $f(k)$ is determined considering (3.13).

3.2.3 Parameter Values

In this section we explain the choices we made for the various parameters of the suspension system for the simulations we will present later on. Since we want to reproduce some of the simulation results shown in [11], we choose the same parameters.
The masses and elastic constants are assumed: $M_w = 28.58$ Kg, M_s=288.90 Kg, $\lambda_t = 155900$ N/m, $\lambda_s = 14345$ N/m.

With the disturbance x_0 we want to describe an asphalt road profile and therefore we assume $\alpha = 0.15$ m^{-1}, $V = 30$ m/s and $\sigma^2 = 9$ mm^2.
As already mentioned above the choice of the sampling interval T needs some warrant. We have assumed $T = 0.01$ s, that is to say a sampling frequency equal to $\omega_s = 2\pi/T \simeq 6 \cdot 10^2$ rad/s. This is essentially due to the following reasons:

- The bandwidth of the passive suspension system described by (3.1a) is $\omega_b < 2 \cdot 10^2$ rad/s. A sampling frequency of $\omega_s \simeq 6 \cdot 10^2$ rad/s is in good agreement with Shannon's theorem [20] that requires $\omega_s > 2\omega_b$.

- This choice of sampling interval is consistent with eq. (3.9). In fact, $2\pi \left(\frac{M_s}{\lambda_t}\right)^{0.5} \simeq 8.5 \cdot 10^2$ s and thus we can be sure that the system will maintain the properties of stabilizability and observability.

- To change f the controller must change the opening of the damper valve. Present technologies impose a limit of about 10^2 Hz on the updating frequency of the damper coefficient.

We have taken $u_{\max} = 3000$ N that is slightly less than the total weight resting on one wheel. A control force of higher magnitude may cause loss of contact between wheel and road. Furthermore, this constraint also limits the acceleration of the sprung mass and this is a necessary condition for the comfort of passengers. Finally we have assumed $f(k) \in [800, 3000]$ Ns/m.
The performance of the semiactive suspension design will be presented and discussed in Chapter 7.

Chapter 4

Optimal Gain Switching

4.1 Introduction

The approach to design a controller for linear sampled-data systems with bounded control variables we present in this chapter was first introduced by Yoshida [22] and applied to semiactive suspension systems in [11]. For this kind of problems it makes use of a Linear-Quadratic (LQ) optimal controller which minimizes a quadratic cost function of the system's state and control under no constraint, thus the optimal control law is a linear state feedback and hence, relatively simple in its analysis and realization. The trade-off between the optimal control law and the control constraint can be made by suitably selecting the weighting factors in the quadratic cost function. Because of the linearity of the system, the control variable becomes small as the state approximates its equilibrium point (usually the origin). To achieve a more effective control, that means a fast controller, it is necessary to keep the control variable close to its maximal allowable value. Use of a *variable feedback gain* enables us to design such a controller.
Therefore the design method that was developed by Yoshida yields a controller having a variable feedback gain using the performance index represented by a quadratic function of the state. The idea can be briefly outlined considering two phases:

off-line: choose a set of LQ optimal feedback gains corresponding to a sequence of increasing weights on the state in the usual quadratic cost function and, for each gain, find a linear region, i.e. a set of initial conditions such that the control law is satisfactory subject to the control constraint

on-line: at each sampling instant, apply the highest gain whose linear region includes the current state.

Since this procedure implies that the state penalty becomes progressively higher as the state approaches the origin, a lower-gain linear control is used far from the origin, while a higher-gain control near the origin does not violate the constraint on the control variable. As at every sampling instant the optimal feedback gain according to the present state is selected we also call this procedure *optimal gain switching* (OGS).

4.2 The controller design

In this section in a first step we will provide the theoretical background on which the OGS approach is based. In a second step we will present an algorithm to use the variable feedback gain method to determine a control law for a linear sampled-data system.

4.2.1 Theoretical Background

For the following system

$$x(k+1) = Gx(k) + Hu(k) \tag{4.1}$$

we aim at determining a suitable control law. We consider as target the control law $u^*(\,\cdot\,)$ that minimizes a performance index of the form:

$$J = \sum_{k=0}^{\infty} x(k) Q\, x(k), \tag{4.2}$$

(with Q positive semidefinite) under the constraint

$$|u(k)| \leq u_{max} \quad (k \geq 0) \tag{4.3}$$

It can be seen in the literature that the optimal solution $u^*(\,\cdot\,)$ does not correspond in general to a feedback control law and furthermore its computation is quite burdensome. The procedure proposed by Yoshida therefore approximates the optimal control law $u^*(\,\cdot\,)$ by switching among feedback control laws whose gains can be computed as solution of a family of LQR problems.
To determine the OGS control law u_{OGS} in a first step we consider a family of performance indexes

$$J_\rho = \sum_{k=0}^{\infty} [\,\rho\, x^T(k) Q\, x(k) + r u^2(k)], \quad \rho > 0,\ r > 0 \tag{4.4}$$

The resulting linear quadratic control law is the state feedback expressed by

$$u_\rho(k) = -K_\rho\, x(k) \tag{4.5}$$

where the gain matrix K_ρ is obtained by solving an algebraic Riccati equation. The closed loop system is then represented by

$$x(k+1) = \hat{G}_\rho\, x(k) \tag{4.6}$$

where $\hat{G}_\rho = G - HK_\rho$.
For a given value of ρ it is possible to compute a *linear region* Γ_ρ in the state space such that for any point x_0 within this region the following equation holds:

$$|u_\rho(k)| \equiv |K_\rho \hat{G}_\rho^k\, x_0| \leq u_{max}, \quad (k \geq 0) \tag{4.7}$$

Thus, considering the system (4.1) without disturbances, feedback law u_ρ and an initial state $x_0 \in \Gamma_\rho$ we can be sure that in its future evolution the value of the control input will always satisfy the constraint (4.3).
To construct the linear regions Γ_ρ we define the following vectors

$$z(k) = (\hat{G}_\rho^T)^k K_\rho^T, \quad k = 0, 1, \ldots \tag{4.8}$$

and the sets

$$L(k) = \left\{ x_0 \,\middle|\, |\, u(k)\,| = |\, z^T(k)\, x_0\,| \le u_{max} \right\} \tag{4.9}$$

$$S(j) = \bigcap_{k=0}^{j} L(k) \tag{4.10}$$

It is evident that $S(j)$ is a monotonically nonincreasing set and its limit is Γ_ρ. Since $L(k)$ is a set bounded by two parallel hypersurfaces, Γ_ρ is a polyhedral convex set. Let $\tilde{S}(j)$ be the convex combination of $\pm\, z(k)$ $(k = 0 \sim j)$, i.e.

$$\begin{aligned} \tilde{S}(j) &= C\{z(k), \quad k = 0, \ldots, j\} \\ &= \left\{ z \,\middle|\, z = \sum_{k=0}^{j} \alpha(k)\, z(k),\ \sum_{k=0}^{j} |\alpha(k)| \le u_{max} \right\} \end{aligned} \tag{4.11}$$

Since

$$\begin{aligned} S(j) &= \left\{ x \,\middle|\, |z^T\, x| \le u_{max},\ z \in \tilde{S}(j) \right\} \\ \tilde{S}(j) &= \left\{ z \,\middle|\, |x^T\, z| \le u_{max},\ x \in S(j) \right\} \end{aligned}$$

$\tilde{S}(j)$ and $S(j)$ are called the dual region of $S(j)$ and $\tilde{S}(j)$, respectively. The region $\tilde{S}(j)$ is also a monotonically noninceasing set and the dual region of its limit is Γ_ρ. The following theorem is proven in [22]:

Theorem 9. *The linear region Γ_ρ can be described by*

$$\Gamma_\rho = S(j_0) \tag{4.12}$$

where

$$j_0 = \min \left\{ j \,\middle|\, z(j+1) \in \tilde{S}(j) \right\} \tag{4.13}$$

To decide wether an initial point lies in the linear region or not it is sufficient to verify (4.7) for the first $j_0 + 1$ sampling instants. The value of j_0 is a function of ρ and can be computed by solving a linear programming problem.
To check wether a point belongs to the linear region or not we introduce the following matrix-notation

$$Z_\rho = \begin{bmatrix} K_\rho \\ K_\rho \hat{G}_\rho \\ \vdots \\ K_\rho \hat{G}_\rho^{j_0} \end{bmatrix}$$

Thus an initial point $x_0 \in \Gamma_\rho$ if and only if $-u_{\max} \le Z_\rho x_0 \le u_{\max}$.

4.2.2 Design algorithm

As we already outlined in the Introduction the design procedure consists of an off-line and an on-line phase, which will be specified below.

Algorithm 1

off-line

Step 1: Choose a finite set of m values for ρ, namely $\{\rho_1, \ldots, \rho_m\}$. Determine the corresponding gain matrices K_ρ for each ρ by solving an algebraic Riccati equation.

Step 2: Construct the linear region Γ_ρ, therefore calculate the matrices

$$Z_\rho = \begin{bmatrix} K_\rho \\ K_\rho \hat{G}_\rho \\ \vdots \\ K_\rho \hat{G}_\rho^{j_0} \end{bmatrix} \tag{4.14}$$

where j_0 satisfies (4.13).

on-line

Step 1: Assume $v = 0$ and $k = 0$.

Step 2: Sample the state $x(k)$.

Step 3: Determine the largest number v such that

$$v = \max\{\, i \mid x(k) \in \Gamma_{\rho_i},\ i = 0, \ldots, m\}. \tag{4.15}$$

and set $\rho(k) = \rho_v$. The condition $x(k) \in \Gamma_\rho$ is true, if and only if the following inequalities hold:

$$-u_{\max} \leq Z_\rho x_0 \leq u_{\max} \tag{4.16}$$

Step 4: Apply the control force according to

$$u_{\mathrm{OGS}}(k) = -K_{\rho_v}, \tag{4.17}$$

set $k = k + 1$, and return to **on-line step 2**.

It has been shown by Yoshida that if no disturbance is acting on the system, $\rho(k)$ is a nondecreasing function of k. In this case at a sampling instant k it is sufficient in **on-line step 3** to replace (4.15) by

$$v = \max\{\, i \mid x(k) \in \Gamma_{\rho_i},\ i = v, \ldots, m\} \tag{4.18}$$

The selection of the weighting coefficients needs some further comments. A good choice of the values ρ_i in the **off-line step 1** may influence the performance of the OGS law. As m

i	1	2	3	4	5	6	7	8	9	10
ρ_i	0.01	0.1	0.5	1	4	20	50	100	1000	10^5
j_{0_i}	46	28	19	19	17	16	16	15	15	15

Table 4.1: Weight coefficients ρ and relative indices j_ρ

increases, the performance index J_ρ decreases, but the procedure becomes computationally more intensive.
The weighting coefficient ρ_1 should be determined such that the linear region Γ_{ρ_1} contains all the initial conditions of interest. The weighting coefficient ρ_m should be selected such that the region Γ_{ρ_m} covers small disturbances or very small system noises. The coefficients $\rho_2, \ldots, \rho_{m-1}$ should be chosen taking into account the size of the linear region Γ_i. Once ρ_1, ρ_m and m are determined, there are ρ_i such that the ratios of the norm between two adjacent gains are constant, i.e.,

$$\frac{\|K_{\rho_i}\|}{\|K_{\rho_{i-1}}\|} = \left(\frac{\|K_{\rho_m}\|}{\|K_{\rho_1}\|}\right)^{\frac{1}{m}} \tag{4.19}$$

Further, we want to comment on the computational complexity of the OGS control law. The most burdensome part of this procedure is the off-line phase, where the vectors z_i are computed. During the on-line phase, it is necessary at each sampling instant k to compute at most m matrix products $Z_\rho x(k)$. It has been shown in [11] that an appropriate choice of the sampling time T ensures that the time needed for on-line computations does not exceed T itself.

4.3 OGS for the suspension system

We have applied the OGS design procedure to the suspension system which was introduced in chapter 3. In chapter 7 we will reproduce some of the simulation results presented in [11] considering the fourth order model schematized in Fig. 3.1. Therefore we assume Q and r in eq. (4.4) like this

$$Q = \begin{bmatrix} 11 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}; \quad r = 0.8 \cdot 10^{-9}$$

These weights lead to a good performance in terms of road holding and passenger's comfort. Another important aspect of the design procedure is the choice of the weighting coefficients $\{\rho_1, \rho_2, \ldots, \rho_m\}$. We assumed $m = 10$ as it seems a good trade-off between computational efficiency and performance. The chosen values are shown in Table 4.1. It can be noted that in this case j_0 is a nonincreasing function.

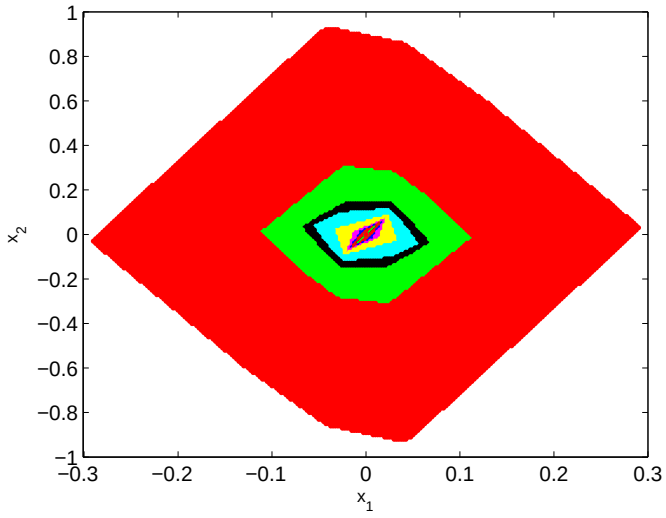

Figure 4.1: Cut through the linear regions Γ_ρ for the fourth-order suspension system at $x_3 = x_4 = 0$

The Linear Regions Γ_ρ

The linear regions Γ_{ρ_i}, which we will also call "Yoshida regions" in the following, are determined during the off-line phase of the design procedure. Choosing the set $\{\rho_1, \rho_2, \ldots, \rho_m\}$ as presented beforehand, the linear regions will always be nested. The linear regions according to the ρ_i in Table 4.1 are illustrated in Figure 4.1, which shows a cut through $x_3 = x_4 = 0$ of the linear regions for the fourth-order two-degrees of freedom model of the suspension system.

Chapter 5

Discontinuous Variable Structure Control

5.1 Introduction

Variable Structure Control (VSC) Systems are characterized by a suite of feedback control laws and a decision rule (the switching function or selection strategy) and can be regarded as a combination of subsystems where each subsystem has a fixed control structure and is valid for specified regions of system behavior. The advantage is its ability to combine useful properties of each of the composite structures of the system. Furthermore, the system may be designed to possess new properties not present in any of the composite structures alone. The initial ideas for discontinuous VSC, which will be considered here, were proposed by Kiendl and Schneider [14] and a suitable design method was presented by Adamy [1]. The only contribution to these concepts in English is a survey written by Adamy [2].
Similar to the OGS method the variable structure controller depending on the system's state either switches between a finite number of linear subcontrollers (discontinuous VSC) or changes the controller parameters continuously (soft VSC) with the objective to obtain a better performance in terms of shorter settling times avoiding violation of control signal constraints.
The discontinuous VSC method makes use of a set of nested, positively invariant sets each with a dedicated linear controller. During the regulation cycle, the trajectory runs from a positively invariant region in the state space into the next smaller one simultaneously activating the assigned controller.
Here we briefly outline the general structure of the dVSC we have implemented to the semiactive suspension model. Later on we will also comment on the extension to soft VSC.
Consider the linear plant in continuous time

$$\dot{x} = Ax + Bu \tag{5.1}$$

under the control signal constraint

$$|u| \leq u_{\max}. \tag{5.2}$$

The controller

$$u = \mathscr{F}(x, p) \tag{5.3}$$

where $\mathscr{F}$ is a general operator, depends on the system's state x and a selection parameter p, that is computed by a selection strategy or supervisor, i.e.,

$$p = S(x), \tag{5.4}$$

defined by a discontinuous function S. The selection strategy switches between a finite number k of different subcontrollers.
As already mentioned, the objectives of such dVSC are often improved settling times, in the case of VSC lacking sliding modes[1] or robustness in the case of sliding mode controls. However, their disadvantages are the discontinuities occurring in their control signals u and the high-frequency switching, that often reduces actuator lives.
Soft VSC have a continuous set of subcontrollers and thus guarantee a smooth control signal. In this case, we can generalize the selection strategy of (5.4)

$$S(x, p^{(n)}, \ldots, p) = 0, \tag{5.5}$$

which also includes dynamic behavior and implicit equations. Soft VSC allows achieving settling times close to those of time-optimal controls, but in contrast to these soft VSC requires much less effort for the design process and can be implemented more easily.
In the next section we illustrate the design process of dVSC according to [2].

5.2 Discontinuous VSC employing nested Lyapunov functions

Since the direct method of Lyapunov's stability theory is essential to this control approach we recall some principles from chapter 2, (see page 13). We repeat the following theorem.

Theorem 10. *Let the differential equation $\dot{x} = f(x)$ with a continuous function f having an equilibrium state $x = 0$. If exists a function $v(x)$ having continuous partial derivatives and if*

$$v(0) = 0, \tag{5.6a}$$
$$v(x) > 0, \quad x \neq 0, \tag{5.6b}$$
$$\dot{v}(x) < 0, \quad x \neq 0, \tag{5.6c}$$

then the equilibrium $x = 0$ will be asymptotically stable and $v(x)$ will be called a ***Lyapunov function****.*

[1] The purpose of a sliding mode controller is to drive the plant's state trajectory onto a predetermined surface in the state space (*sliding surface*) and to maintain it on this surface subsequently. Ideally, once it has reached the sliding surface the state trajectory "slides" along this surface into the origin.

For stable linear systems $\dot{x} = Ax$ it will always be possible to compute a quadratic Lyapunov function

$$v(x) = x^T R\, x \tag{5.7}$$

where R is a positive-definite matrix solving the so-called *Lyapunov equation*:

$$A^T R + RA = -Q \tag{5.8}$$

for an arbitrary positive-definite matrix Q. If there is a Lyapunov function $v(x)$ for a system $\dot{x} = f(x)$ and

$$G = \{x \mid v(x) < c\} \tag{5.9}$$

is bounded, then G is a positively invariant set that is also called *Lyapunov region*, i.e. trajectories that start therein will never leave it.
We will illustrate the idea of dVSC below and therefore consider again system (5.1) under the constraint (5.2) and the selection strategy (5.4). Furthermore, we consider only bounded sets $X_0 \subset \mathbb{R}^n$ of possible initial vectors $x(t = 0)$, since $X_0 = \mathbb{R}^n$ is usually not of practical interest. The three major elements of the dVSC design procedure are:

(D1) Choose a family of k linear state controllers $u = -K_p\, x$ leading to stable control loops

$$\dot{x} = (A - BK_p)\, x = \hat{A}_p\, x, \quad p = 1, \ldots, k \tag{5.10}$$

whose response times decrease with increasing index p.

(D2) According to each control loop (5.10) construct a Lyapunov region

$$G_p = \{x \mid v_p(x) < c_p\} \tag{5.11}$$

where c_p determines the size of G_p. Moreover, G_p should be such that all $x \in G_p$ satisfy the constraint $|u| = |K_p\, x| \leq u_{\max}$.

(D3) The Lyapunov regions should be nested one inside the other in accordance with

$$G_{p+1} \subset G_p, \quad p = 1, \ldots, k-1 \tag{5.12}$$

with an increasing index p.

Analogously to the OGS design process the VSC method consists of an off-line and an on-line phase. The three steps that were mentioned above and will be explained more precisely below represent the off-line phase. During the on-line phase the controller determines the smallest Lyapunov region that contains the current system's state and activates the subcontroller belonging to this region. Upon the trajectory's entry into a smaller region, the controller switches to the next assigned subcontroller.

In the first step the subcontrollers' matrices K_p are determined utilizing pole placement (refer to section 2.4.1, see page 15), such that the n eigenvalues $\lambda_{p,j}$ of $\hat{A}_p$ conform to

$$\lambda_{p+1,j} = h\,\lambda_{p,j}, \quad h > 1 \tag{5.13}$$

and lead to a stable closed loop, i.e. $Re\{\lambda_p\} < 0$. These controllers thus accelerate the control system's behavior, while simultaneously causing a similar behavior, since the eigenvalue configuration remains the same.
In a second step the Lyapunov regions are constructed employing quadratic Lyapunov functions $v_p(x) = x^T R_p\,x$. The matrix R_p is the solution of the Lyapunov equation

$$\hat{A}_p^T R_p + R_p\,\hat{A}_p = -Q_p \tag{5.14}$$

The matrices Q_p have to be positive-definite, $Q_{p+1} = Q_p$ is frequently a reasonable choice. Thus, the Lyapunov regions will be ellipses determined by the matrices R_p. Since the condition $|K_p\,x| \leq u_{\max}$ has to be satisfied for all $x \in G_p$ and should be exploited as good as possible, the c_p in (5.11) are chosen such that the hyperplanes

$$\pm K_p\,x = u_{\max} \tag{5.15}$$

are tangent to the elliptical Lyapunov regions. In order to determine these c_p we solve the optimization problem

$$\begin{aligned} \max_{R_p} \quad & x^T R_p\,x \\ \text{subj. to} \quad & \pm\,K_p\,x = u_{\max} \end{aligned} \tag{5.16}$$

whose solution yields

$$c_p = \frac{u_{\max}^2}{K_p\,R_p^{-1}K_p^T} \tag{5.17}$$

The largest Lyapunov region G_1 has to be determined such that $X_0 \subseteq G_1$, i.e. the first region includes all possible initial states.
Finally, in a third step we verify that all k regions G_p are nested. If all points of interest satisfy

$$\frac{x^T R_p\,x}{c_p} < \frac{x^T R_{p+1}\,x}{c_{p+1}} < 1 \tag{5.18}$$

then $G_{p+1} \subset G_p$ is ensured. To check wether (5.18) is true or not it is sufficient to make sure that the matrices

$$\frac{R_{p+1}}{c_{p+1}} - \frac{R_p}{c_p} \tag{5.19}$$

are positive definite for $p = 1, \ldots, k-1$.

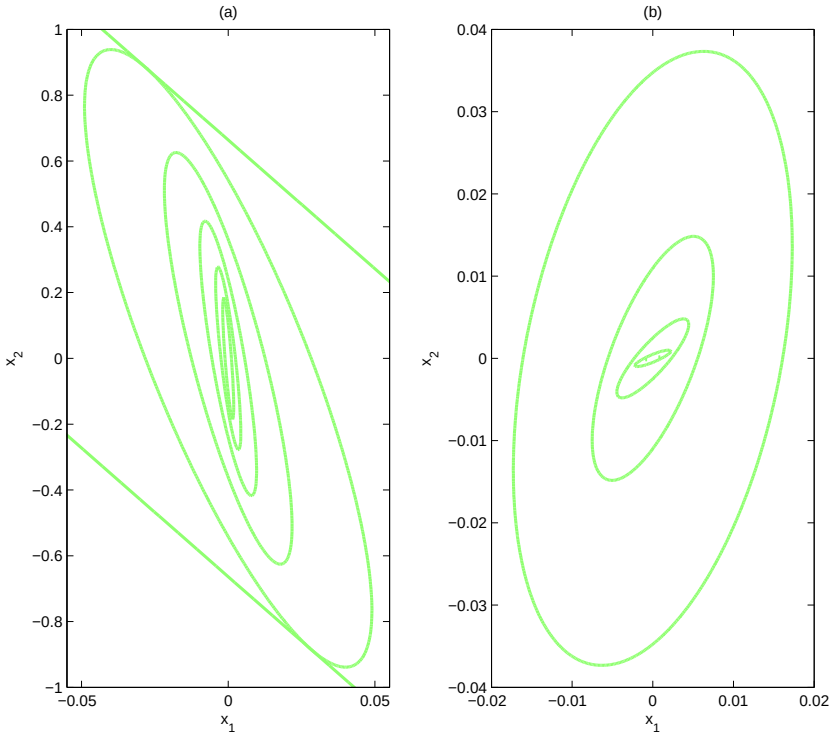

Figure 5.1: Family of nested Lyapunov regions (a) for the second-order suspension model (b) cut through $x_3 = x_4 = 0$ for the fourth-order suspension model; $h = 1.5$

The Lyapunov regions (5.11) for the second-order and the fourth-order suspension system are depicted in Figure 5.1 (a) and (b), respectively. The first set of eigenvalues in (5.13) was chosen according to the closed loop eigenvalues of the OGS for the fourth largest region.
Note that in Fig. 5.1 the axis scales are different, i.e. the Lyapunov regions for the fourth-order model are very small compared to those of the OGS in Figure 4.1. We were not able to enlarge the Lyapunov regions by varying the Q_p in (5.14), moreover the size of the Lyapunov regions seems to be independent from the Q_p. Therefore we illustrate the general idea of soft VSC in the next section, in order to find out wether it could be useful to enlarge the Lyapunov regions.

In addition to the presented concepts Kiendl, Stelzner and Adamy introduced parallelepipeds as an alternative to ellipsoidal Lyapunov regions [15]. Therefore they considered $v(x) = \|Wx\|_\infty$ as an alternative to the quadratic Lyapunov function (5.7), where $\|\cdot\|_\infty$ is the maximum vector norm. We did not implement these concepts as they did not seem to be useful to enlarge the Lyapunov regions significantly.

5.3 Soft VSC employing implicit Lyapunov functions

The concept of soft VSC was introduced by Kiendl and Schneider [14] applying a continuously changing control law by nesting the Lyapunov regions infinitely dense. The control concept consists of three major elements analogue to those of dVSC:

(S1) A continuous family of linear state controllers $u = -K(p)\,x$, which lead to stable control loops

$$\dot{x} = (A - BK(p))\,x \tag{5.20}$$

(S2) According to each control loop (5.20) determine a Lyapunov region $G(p)$ that guarantees $|u| = |K(p)\,x| \leq u_{\max}$ for all $x \in G(p)$.

(S3) The Lyapunov regions $G(p)$ should be nested infinitely dense, i.e. $G(p+\varepsilon) \subset G(p)$ should be satisfied for every small $\varepsilon > 0$, which implies that the size of G(p) will decrease with increasing p.

A suitable design method was developed in [1] and we will present the basic ideas here. The Lyapunov regions $G(p)$ involved may be defined by

$$G(p) = \{x \,|\, g(p,x) < 0\} \tag{5.21}$$

together with a suitable function $g(p,x)$. The control vector $k(p)$ associated with a Lyapunov region $G(p)$ will be activated upon the trajectory's entry into $G(p)$ which will occur whenever $x(t)$ lies on the border

$$\partial G(p) = \{x \,|\, g(p,x) = 0\} \tag{5.22}$$

of $G(p)$. The parameter p and thus the control law $u(p,x)$ are determined for each $x(t)$ during regulation cycles by the implicit equation

$$g(p,x) = 0. \tag{5.23}$$

With regard to the control g must satisfy the following two conditions:

(S4) There must be a unique solution to (5.23) with respect to p.

(S5) The function g should be chosen such that stability is ensured for the closed loop system.

Satisfying condition (S4) is necessary in order to be able to assign one and only one value p to each state vector x. If this condition is not met either (5.23) has no solution or several solutions can be found. In these irregular cases the control approach cannot be realized. However, (S4) holds true if it is possible to express (5.23) like this: $p = p\,(x)$.
Without loss of generality, (S1),(S2) and (S3) can be reformulated such that the size of $G(p)$ decreases as the parameter p decreases and $p = 0$ for $x = 0$. Since p decreases along the system's trajectories, equals 0 in the origin and is always positive

$$v(x) = p \tag{5.24}$$

is a Lyapunov function for the system. Thus we can rewrite (5.20) and (5.23) in the following way:

$$\dot{x} = (A - BK(v))\,x = \hat{A}(v)\,x \tag{5.25}$$
$$0 = g(v, x) \tag{5.26}$$

Introducing implicit Lyapunov functions like (5.26) leads to the question which are the conditions for g to define an implicit Lyapunov function. This theoretical background can be found in [1, 2]. Below we will briefly outline how to obtain the subcontrollers and the selection strategy.
To design a particular control we first assume, without loss of generality, that the linear system (5.1) is in controllable standard form, where

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & & 1 \\ -a_0 & -a_1 & -a_2 & \cdots & -a_{n-1} \end{bmatrix}, \qquad \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

or have been transformed into the above.
We now choose $K(v)$ in (5.25) such that the closed loop eigenvalues λ_i will be shifted onto rays $\lambda_i(v) = \lambda_i(1)v^{-1}$, which start at $\lambda_i(1)$ and run towards negative infinity with decreasing v. The above choice of $K(v)$ leads to faster linear control subsystems (5.25), since v decreases during regulation cycles, which is the main aim of the soft VSC approach. In order to achieve these ray-like eigenvalue paths $\lambda_i(v)$, we need to formulate the control matrix $K(v)$ as follows:

$$K^T(v) = \begin{bmatrix} \hat{a}_0 v^{-n} - a_0 \\ \hat{a}_1 v^{-(n-1)} - a_1 \\ \vdots \\ \hat{a}_{n-1} v^{-1} - a_{n-1} \end{bmatrix} \tag{5.27}$$

where the $\hat{a}_i$ are the coefficients of the characteristic polynomial determined by the closed loop matrix $\hat{A}(v = 1)$. Arranging the plant's coefficients of the characteristic polynomial a_i, that are determined by the matrix A, in a vector

$$a^T = \begin{pmatrix} a_0 & a_1 & \dots & a_i & \dots & a_{n-1} \end{pmatrix} \tag{5.28}$$

and analogously integrate the coefficients $\hat{a}_i$ in a vector

$$\hat{a}^T = \begin{pmatrix} \hat{a}_0 & \hat{a}_1 & \dots & \hat{a}_i & \dots & \hat{a}_{n-1} \end{pmatrix} \tag{5.29}$$

yields a vector notation of the control matrix $K(v)$

$$K^T(v) = D^{-1}(v)\,\hat{a} - a \tag{5.30}$$

where D is the diagonal matrix

$$D(v) = \mathrm{diag}(v^n, \ldots, v^2, v). \tag{5.31}$$

Combining (5.30) and (5.25), the closed loop matrix can be rewritten in the form

$$\hat{A}(v) = \frac{1}{v} D(v)\, \hat{A}_1 D^{-1}(v), \quad \hat{A}_1 = \hat{A}(1). \tag{5.32}$$

In the second design step, we need to choose suitable Lyapunov regions $G(v) = \{x \,|\, g(v,x) < 0\}$ to fulfill condition (S2). We will choose elliptical Lyapunov regions $G(v) = \{x \,|\, x^T R(v) x - 1 < 0\}$ according to the dVSC concept. We multiply the quadratic form by an additional function $e(v)$, where $e(v) > 0$, in order to ensure that the pair of hyperplanes given by $|K(v)x| = u_{\max}$ will be tangent to the elliptic Lyapunov region, and obtain

$$G(v) = \{x \,|\, g(v,x) = e(v)\, x^T R(v)\, x - 1 < 0\}. \tag{5.33}$$

Similarly to the discontinuous case (5.17) the solution to this optimization problem yields

$$e(v) = \frac{K(v) R^{-1}(v) K(v)^T}{u_{\max}^2}. \tag{5.34}$$

Analogously to the dVSC, we have to verify that $X_0 \subseteq G(1)$, to make sure that all initial points of interest are covered by the largest Lyapunov region.
It has been shown by Adamy [2] that the soft VSC approach consists of the control loop (5.25), the control matrix (5.30) and the selection strategy

$$g(v,x) = e(v)\, x^T R(v)\, x - 1 = 0, \tag{5.35}$$

where

$$e(v) = \frac{1}{u_{\max}^2} [a^T D(v) R_1^{-1} D(v) a - 2\hat{a}^T R_1^{-1} D(v) a + \hat{a} R_1^{-1} \hat{a}] \tag{5.36}$$

$$R(v) = D^{-1}(v) R_1 D^{-1}(v) \tag{5.37}$$

and $e(v)$ is a polynomial of order 2^n or less. Computing the parameters of this control involves choosing a suitable vector $\hat{a}$, a matrix R_1 and verifying $X_0 \subseteq G(1)$. The matrix R_1 will be chosen solving the constrained optimization problem

$$\begin{aligned} &\max_{R_1} \frac{1}{\sqrt{e^n(1) \det R_1}} \\ &\text{subj. to} \quad \hat{A}_1^T R_1 + R_1 \hat{A}_1 = -Q_1, \\ &\qquad\quad N R_1 + R_1 N = -S_1, \\ &\qquad\quad \max_{v \in [0,1]} e'(v) \le 0 \end{aligned} \tag{5.38}$$

where Q_1 and S_1 are arbitrary positive-definite matrices. For further explanation concerning the theoretical background of soft VSC refer to [1, 2].
We did not implement the soft VSC, because comparing (5.34) to (5.17) we supposed that the problem which we met for the discontinuous VSC, i.e. that the Lyapunov regions were to small, would occur also for the continuously changing Lyapunov regions of the soft VSC.

Chapter 6

Explicit Model Predictive Control

6.1 Introduction

In this chapter we present the general ideas of explicit model predictive control (eMPC).It has been an extension to model predictive control that has become an accepted standard for complex constrained multivariable control problems for discrete time systems. Applying the MPC approach, at each sampling time, starting at the current state, an open-loop optimal control problem is solved over a finite time horizon. At the next time step, the computation is repeated starting from the new state and over a shifted horizon, leading to a moving horizon policy. The solution relies on a linear dynamic model, respects all input and output constraints and optimizes a quadratic or linear performance index.

When the MPC concept appeared the computations were executed on-line, so that the MPC was only applicable to relatively slow and/or small problems. The explicit MPC approach presented here, moves all the burdensome computations off-line and partitions the state space into polytopic regions, so that during the on-line phase of the control procedure according to the current state the actual subcontroller can be found out of a table. The on-line phase of the eMPC is similar to that one of the other concepts presented above. By introducing the partitioning of the state space, the eMPC can also be applied to problems with faster dynamics like the suspension system.

In this chapter we will illustrate some fundamentals of polytopes theory, multi-parametric programming and its application to constrained finite time optimal control (CFTOC). At the end of this chapter we present several partitions we obtained with the free model predictive toolbox for MATLAB.

6.2 Polytopes Theory

Polytopic (or, more general, polyhedral) sets are an integral part of multi-parametric programming. For this reason we give some of the definitions of polytopes [17]. For more details we refer to the literature [9, 23].

Definition 7. (polyhedron) A convex set $\mathcal{Q} \subseteq \mathbb{R}^n$ given as an intersection of a finite number of closed half-spaces

$$\mathcal{Q} = \{x \in \mathbb{R}^n \,|\, Q^x x \leq Q^c\} \tag{6.1}$$

is called *polyhedron*.

Definition 8. (polytope) A bounded polyhedron $\mathcal{P} \subset \mathbb{R}^n$

$$\mathcal{P} = \{x \in \mathbb{R}^n \,|\, P^x x \leq P^c\}, \tag{6.2}$$

is called *polytope*.

It is obvious from the above definitions that every polytope represents a convex, compact[1] set. We say that a polytope $\mathcal{P} \subset \mathbb{R}^n \,|\, P^x x \leq P^c$ is *full dimensional* if $\exists x \in \mathbb{R}^n : P^x x < P^c$. Furthermore, if $\|(P^x)_i\| = 1$, where $(P^x)_i$ denotes the i-th row of a matrix P^x, we say that the polytope $\mathcal{P}$ is normalized. One of the fundamental properties of a polytope is that it can also be described by its *vertices*

$$\mathcal{P} = \{x \in \mathbb{R}^n \,|\, x = \sum_{i=1}^{v_p} \alpha_i V_p^{(i)},\ 0 \leq \alpha_i \leq 1,\ \sum_{i=1}^{v_p} \alpha_i = 1\} \tag{6.3}$$

where $V_p^{(i)}$ denotes the i-th vertex of $\mathcal{P}$, and v_p is the total number of vertices of $\mathcal{P}$. The half-space representation (6.2) is often referred to as $\mathcal{H}$ as well as the vertex representation (6.3) is denoted by $\mathcal{V}$.

Definition 9. (face) Linear inequality $a'x \leq b$ is called valid for a polyhedron $\mathcal{P}$ if it holds for all $x \in \mathcal{P}$. A subset of a polyhedron is called a face of $\mathcal{P}$ if it is represented as

$$\mathcal{F} = \mathcal{P} \cap \{x \in \mathbb{R}^n \,|\, a'x = b\}, \tag{6.4}$$

for some valid inequality $a'x \leq b$. The faces of a polyhedron $\mathcal{P}$ of dimension 0, 1, (n-2) and (n-1) are called vertices, edges, ridges and facets respectively.

We say that a polytope $\mathcal{P} \subset \mathbb{R}^n,\ \mathcal{P} = \{x \in \mathbb{R}^n \,|\, P^x x \leq P^c\}$ is in a *minimal representation* if a removal of any of the rows in $P^x x \leq P^c$ would change it (i.e. there are no redundant halfspaces). It is straightforward to see that a normalized, full dimensional polytope $\mathcal{P}$ has a unique minimal representation. This fact is very useful in practice. Normalized, full dimensional polytopes in a minimal representation allow us to avoid any ambiguity when comparing them and very often speed up other polytope manipulations.
The following definition is useful when we specify the state space partition of the CFTOC later on.

Definition 10. A collection of sets $R_1, \ldots, R_N$ is a *partition in the broad sense* of a set Θ if

[1] A compact set is bounded and closed.

(i) $\bigcup_{i=1}^{N} R_i = \Theta$,

(ii) $(R_i \setminus \partial R_i) \cap (R_j \setminus \partial R_j) = \emptyset,\ \forall\, i \neq j$

where ∂ denotes the boundary. Moreover $R_1, \ldots, R_N$ is a *polyhedral partition in the broad sense* of a polyhedral set Θ if $R_1, \ldots, R_N$ is a partition in the broad sense of Θ and the R_i's are polyhedral sets.

6.3 Multi-Parametric Quadratic Programming

In this section we investigate multi-parametric quadratic programs (mp-QP) of the form

$$\begin{aligned} J^*(x) = \min_{z} \quad & \{J(z,x) = \tfrac{1}{2} z^T H z\} \\ \text{subj. to} \quad & Gz \leq W + Sx, \end{aligned} \tag{6.5}$$

where $z \in \mathbb{R}^s$ are the *optimization variables*, $x \in \mathbb{R}^n$ is the *vector of parameters*, $H \in \mathbb{R}^{s \times s}$, $H \succ 0$[2], $W \in \mathbb{R}^m$ and $S \in \mathbb{R}^{m \times n}$. The solution to problems of the form (6.5) has been examined a lot recently and here we refer to [4, 5].
Note that the general problem with $J(z,x) = z^T H z + x^T F z$ can always be transformed in the mp-QP (6.5) by using the variable substitution $\tilde{z} \triangleq z + H^{-1} F^T x$.
Given a close polyhedral set $K \subset \mathbb{R}^n$ of parameters,

$$K \triangleq \{x \in \mathbb{R}^n \mid Tx \leq Z\}, \tag{6.6}$$

we denote by $K^* \subseteq K$ the region of parameters $x \in K$ such that the QP (6.5) is feasible and the optimum $J^*(x)$ is finite. For any given $\bar{x} \in K^*$, $J^*(\bar{x})$ denotes the minimum value of the objective function in problem (6.5) for $x = \bar{x}$.
The function $J^* : K^* \to \mathbb{R}$ will denote the function which expresses the dependence on x of the minimum value of the objective function over K^*; $J^*(\cdot)$ will be called value function.
The single-valued function $z^* : K^* \to \mathbb{R}^s$ will describe for any fixed $x \in K^*$ the optimizer $z^*(x)$ related to $J^*(x)$.
We aim at determining the feasible region of parameters $K^* \subseteq K$ and at finding the expression of the valued function $J^*(\cdot)$ and the optimizer function $z^*(\cdot)$. Denote with G_j, S_j, W_j, T_j and Z_j the j-th tow of G, S, W, T and Z respectively. We give the following definition of active and weakly active constraints:

Definition 11. The i-th constraint is **active** at x if $G_i\, z^*(x) - W_i - S_i\, x = 0$, it is **inactive** if $G_i\, z^*(x) - W_i - S_i\, x < 0$. We also define the i-th constraint as **weakly active** if it is active and its corresponding Lagrange multiplier[3] λ_i is zero.

[2] $H \succ 0$ if and only if $x^T Q x > 0 \quad \forall x \in \mathbb{R}^n$.

[3] **Lagrange multipliers** are a method for dealing with constraints in mathematical optimization problems. Suppose the question as given is to find local extremal values of a function of several variables subject to one or more constraints given by setting further functions of the variables to given values. The method introduces a new unknown scalar variable, the Lagrange multiplier, for each constraint; and forms a linear combination involving the multipliers as coefficients. This reduces the constrained problem to an unconstrained problem. It may then be solved, for example by the usual gradient method.

Let $J \triangleq \{1, \ldots, m\}$ be the set of constraint indices. For any $A \subseteq J$, let G_A and S_A be the submatrices of G and S, respectively, consisting of the rows indexed by A.

Definition 12. The optimal partition of J associated with x is the partition $(A(x), NA(x))$

$$\begin{aligned} A(x) &\triangleq \{j \in J \mid G_j\, z^*(x) - S_j\, x = W_j\} \\ NA(x) &\triangleq \{j \in J \mid G_j\, z^*(x) - S_j\, x < W_j\} \end{aligned} \tag{6.7}$$

where $A(x)$ is the *optimal active set* and will be simply referred to as *the set of active constraints at* x.

The multi-parametric analysis uses the concept of *Critical Regions* (CR). For a given $x^* \in K^*$ let $(A, NA) \triangleq (A(x^*), NA(x^*))$, and let

$$CR_A \triangleq \{x \in K \mid A(x) = A\}. \tag{6.8}$$

The set CR_A is the critical region related to the set of active constraints A, i.e., the set of all parameters x such that the constraints indexed by A are active at the optimum of problem (6.5).

Definition 13. For a given set of active constraints A we say that the **Linear Independence Constraint Qualification** (LICQ) holds if the rows of G_A are linearly independent.

In the following the set CR_i denotes the critical region related to the set of active constraints A_i.

6.3.1 Geometric Algorithm for mp-QP

The algorithm to solve the mp-QP (6.5) off-line, that will also be applied to partition the state space off-line in the control problem, consists of two main steps, which can be characterized as follows:

1. Determine the dimension $n' \leq n$ of the smallest affine subspace $\mathcal{K}$ that contains K^*. If $n' < n$, find the equations in x which define $\mathcal{K}$.

2. Determine the partition of K^* into the critical regions CR_i, and find the expression of the functions $J^*(\,\cdot\,)$ and $z^*(\,\cdot\,)$ for each critical region.

Below we give the details of the two steps. The first step is a preliminary one whose goal is to reduce the number of parameters in order to obtain a full-dimensional feasible region of parameters. This eases the second step, which computes the multi-parametric solution and represents the core of the mp-QP algorithm.

Determining the Affine Subspace $\mathcal{K}$

In order to work with a minimal dimension of the parameter vector, the first step aims at finding the affine subspace $\mathcal{K} \subseteq \mathbb{R}^n$ containing the parameters which render (6.5) feasible. A first but simple consideration concerns the column rank r_S of S. Clearly, if $r_S < n,\ n - r_s$ parameters can be eliminated by a simple coordinate transformation in $\mathbb{R}^n$.
Besides this obvious preliminary reduction of the parameter space, there is another case where the number of parameters can be further reduced. Even if the matrix S has full column rank, it is possible that the polyhedron K^* is contained in a subspace of dimension $n' < n$.
Therefore, before solving the mp–QP problem, we need e test for checking the dimension n' of the smallest affine subspace $\mathcal{K}$ that contains K^*. Moreover, when $n' < n$, we need the equations describing $\mathcal{K}$ in $\mathbb{R}^n$. The equations are then used for a change of coordinates in order to reduce the number of parameters from n to n' and to obtain a polyhedron K^* that has full dimension in $\mathbb{R}^{n'}$. For computational details of this step refer to [5].

Determining the Critical Regions

In order to start solving the mp–QP, we need an initial vector x_0 inside the polyhedral set K of parameters over which we want to solve the problem, such that the QP (6.5) is feasible for $x = x_0$. A good choice for x_0 is the center of the largest ball contained in K for which a feasible z exists, determined by solving the LP

$$\begin{aligned} \max_{x,z,\varepsilon} \quad & \varepsilon \\ \text{subj. to} \quad & T_i x + \varepsilon \|T_i\| \leq Z_i, \quad i = 1, \ldots, n_T \\ & Gz - Sx \leq W \end{aligned} \tag{6.9}$$

where n_T is the number of rows T_i of the matrix T. If $\varepsilon \leq 0$, then the QP problem (6.5) is infeasible for all x in the interior of K. Otherwise, we fix $x = x_0$ and solve the QP problem (6.5), in order to obtain the corresponding optimal solution z_0. Such a solution is unique, because $H \succ 0$, and therefore uniquely determines a set of active constraints A_0 out of the constraints in (6.5). In [4, 5] the following theorem is proved:

Theorem 11. *Let $H \succ 0$. Consider a combination of active constraints A_0 and assume that LICQ holds. Then, the optimal z^* and the associated vector of Lagrange multipliers λ^* are uniquely defined functions of x over the critical region CR_0.*

Theorem 11 characterizes the solution only in the neighborhood of a specific x_0, but it does not provide the construction of the set CR_0 where this characterization remains valid. In [4] it is shown that CR_0 is a polyheron in the x-space, and it represents the largest set of $x \in K$ such that the combination of active constraints at the minimizer remains unchanged. Once the critical region CR_0 has been defined, the rest of the space $CR_{\text{rest}} = K - CR_0$ has to be explored and new critical regions generated. The following theorem justifies such a procedure to characterize the rest of the region CR_{rest}.

Region	Inequalities
R_1	$C1 \geq 0,\ x_1 \geq x_1^-,\ x_2^- \leq x_2 \leq x_2^+$
R_2	$C1 \geq 0,\ C2 \leq 0,\ x_2 \leq x_2^+$
R_3	$C2 \leq 0,\ , C3 \geq 0\ x_1 \leq x_1^+,\ x_2 \leq x_2^+$
R_4	$C1 \leq 0,\ C3 \leq 0,\ C4 \geq 0,\ x_1 \leq x_1^+,\ x_2 \geq x_2^-$
R_5	$C1 \leq 0,\ C4 \leq 0,\ C5 \geq 0$

Table 6.1: Definition of the partition $CR_{\text{rest}} \triangleq K \setminus CR_0$

Theorem 12. *Let $Y \subseteq \mathbb{R}^n$ be a polyhedron, and let $CR_0 \triangleq \{x \in Y \mid Ax \leq b\}$ be a polyhedral subset of Y, $CR_0 \neq \emptyset$. Also let*

$$R_i = \left\{ x \in Y : \begin{array}{l} A^i x > b^i \\ A^j x \leq b^j, \forall j < i \end{array} \right\}, \quad i = 1, \ldots, m$$

where $m = \dim(b)$, and let $CR_{rest} \triangleq \bigcup_{i=1}^m R_i$. Then

(i) $CR_{\text{rest}} \cup CR_0 = Y$

(ii) $CR_0 \cap R_i = \emptyset, \quad R_i \cap R_j = \emptyset, \forall\ i \neq j$

*i.e. $\{CR_0, R_1, \ldots, R_m\}$ is a **partition** of Y.*

In order to exemplify the procedure proposed in Theorem 12 for partitioning the set of parameters K, consider the case when only two parameters x_1 and x_2 are present. As shown in Figure 6.1, K is defined by the inequalities

$$\{x_1^- \leq x_1 \leq x_1^+, \quad x_2^- \leq x_2 \leq x_2^+\}$$

and CR_0 by the inequalities

$$\{C1 \leq 0, \ldots, C5 \leq 0\}$$

are affine functions of x. The procedure consists of considering, one by one, the inequalities which define CR_0. Considering, for example, the inequality $C1 \leq 0$, the first set of the rest of the region

$$CR_{\text{rest}} \triangleq K \setminus CR_0$$

is given by

$$R_1 = \{C1 \geq 0,\ x_1 \geq x_1^-,\ x_2^- \leq x_2 \leq x_2^+\},$$

which is obtained by reversing the sign of the inequality $C1 \leq 0$ and removing redundant constraints, which is visualized in Figure 6.1(b).

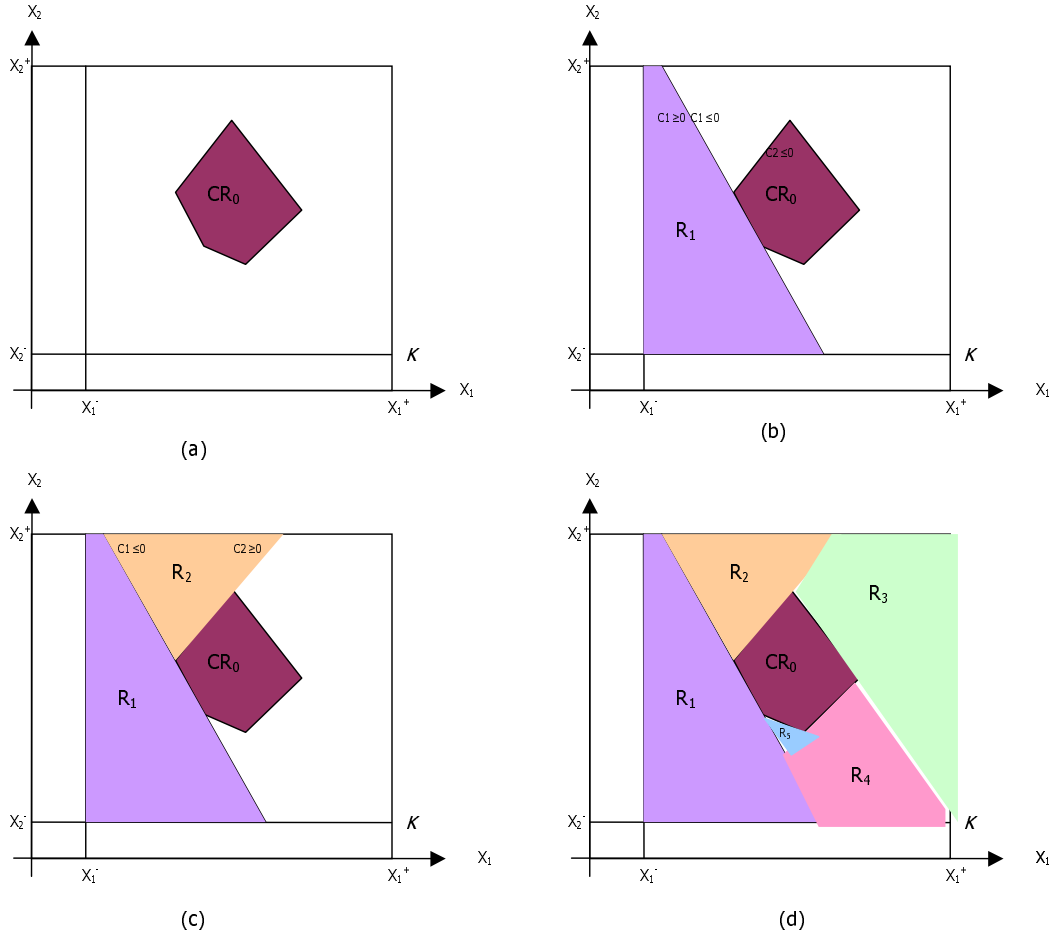

Figure 6.1: Two dimensional example: partition of the rest of the space $CR_{\text{rest}} \triangleq K \setminus CR_0$; (a) set of parameters K and initial region CR_0; (b) partition of CR_{rest}, Step 1; (c) partition of CR_{rest}, Step 2; (d) final partition of CR_{rest}

Thus, by considering the rest of the inequalities, the complete rest of the region is

$$CR_{\text{rest}} = \bigcup_{i=1}^{5} R_i$$

where $R_1, \ldots, R_5$ are given in Table 6.1 and are graphically reported in Figure 6.1(d). Note that the partition strategy suggested in Theorem 12 can also be applied when K is unbounded.

Theorem 12 provides a way of partitioning the non-convex set $K \setminus CR_0$ into polyhedral subsets R_i. For each R_i a new vector x_i is determined by solving the LP (6.9), and, correspondingly, an optimum z_i^*, a set of active constraints A_i, and a critical region CR_i. Theorem 12 is then applied to partition $R_i \setminus CR_i$ into polyhedral subsets, and the algorithm proceeds iteratively.

Note that theorem 12 introduces cuts in the x-space which might spilt critical regions into subsets. Therefore, after the whole x-space has been covered, those polyhedral regions CR_i are determined where the function $z^*(x)$ is the same. If their union is a convex set, it is computed to permit a more compact description of the solution [3].

6.3.2 Continuity and convexity properties

Convexity of the value function $J^*(x)$ and continuity of the solution $z^*(x)$ can be shown as corollaries of the linearity result of theorem 11. This fact together with the convexity of the set of feasible parameters $K^* \subseteq K$ and the piecewise linearity of the solution $z^*(x)$ is proved in the next theorem.

Theorem 13. *Consider the multi-parametric program* (6.5) *and let* $H \succ 0$. *Then, the set of feasible parameters* $K^* \subseteq K$ *is convex. The optimizer* $z^*(x) : K^* \to \mathbb{R}^s$ *is continuous and piecewise affine on polyhedra (PPWA), in particular it is affine in each critical region, and the optimal solution* $J^*(x) : K^* \to \mathbb{R}$ *is continuous, convex and piecewise quadratic on polyhedra.*

Theorem 13 is proven in [4, 5]. Below we will summarize the off-line algorithm for mp–QP.

6.3.3 A summary of the mp-QP Algorithm

Based on the above discussion and results, the main steps of the off-line mp-QP solver are outlined in the following algorithm.

Algorithm 2
Input: Matrices H, G, W, S of problem (6.5) and set K in (6.6)
Output: Multi-parametric solution to (6.5)

1 **Let** $K \subseteq \mathbb{R}^n$ be the set of parameters (states);

2 execute **partition**(K);

3 **end.**

procedure partition(Y)

4 **let** $x_0 \in Y$ and ε the solution to the LP (6.9);

5 **if** $\varepsilon \leq 0$ **then exit**; (no full dimensional) CR is in Y

6 for $x = x_0$, compute the optimal solution (z_0^*, λ_0^*) of the QP (6.5);

7 determine the set of active constraints A_0 when $z = z_0^*,\ x = x_0$, and build $G_{A_0},\ W_{A_0},\ S_{A_0}$;

8 **if** $r = \text{rank}(G_{A_0})$ is less than the number l of rows of G_{A_0}, **then** take a subset of r linearly independent rows, and redefine $G_{A_0},\ W_{A_0},\ S_{A_0}$ accordingly;

9 determine $\lambda_{A_0}^*(x),\ z^*(x)$

10 Characterize the CR

11 Define and partition the rest of the region as in theorem 12;

12 for each new subregion R_i, **partition**(R_i); **end procedure**.

Note that the variables in steps *9* and *10* occur in equations we have not presented in this thesis. For further explanation to these variables refer to the proof of Theorem 11 in [4, 5].
The algorithm explores the set K of parameters recursively: Partition the rest of the region as in Theorem 12 into polyhedral sets R_i, use the same method to partition each set R_i further, and so on. This can be represented as a search tree with a maximum depth equal to the number of combinations of active constraints.

6.4 Constrained Finite Time Optimal Control

For discrete time linear systems it can be proven that the solution to constrained finite time optimal control (CFTOC) problems is a time varying affine feedback control law. We describe how the optimal control law can be efficiently computed by means of multi-parametric quadratic programming for quadratic performance criteria.

6.4.1 Problem formulation

Consider the linear time-invariant system

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{6.10}$$

subject to the constraints

$$Ex(t) + Lu(t) \leq M \tag{6.11}$$

at all time instants $t \geq 0$.[4]
In (6.10)–(6.11), $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, and $y(t) \in \mathbb{R}^p$ are the state, input and output vector respectively.
Define the following cost function

$$J(U_N, x(0)) \triangleq \|Px_N\|_p + \sum_{k=0}^{N-1} \|Qx_k\|_p + \|Ru_k\|_p \tag{6.12}$$

where x_k denotes the state vector at time k obtained by starting from the state $x_0 = x(0)$ and applying to system (6.10) the input sequence $u_0, \ldots, u_{k-1}$. Consider the CFTOC

[4]The results presented here also hold for more general forms of linear constraints, arising, e.g. from constraints on the input rate.

problem

$$\begin{aligned} J^*(x(0)) = \min_{U_N} \quad & J(U_N, x(0)) \\ \text{subj. to} \quad & Ex_k + Lu_k \leq M, \quad k = 0, \ldots, N-1 \\ & x_N \in \mathcal{X}_f \\ & x_{k+1} = Ax_k + Bu_k, \quad k \geq 0 \\ & x_0 = x(0) \end{aligned} \tag{6.13}$$

where N is the time horizon and $\mathcal{X}_f \subseteq \mathbb{R}^n$ is a terminal polyhedral region. In (6.12)–(6.13) we denote with

$$U_N \triangleq [u_0^T, \ldots, u_{N-1}^T]^T \in \mathbb{R}^s,$$

$s \triangleq mN$ the *optimization vector*, with $\|Qx_k\|_p$ the p-norm of the vector x_k weighted with the matrix Q; $p = 1, 2, \infty$ are possible choices. We denote with $\mathcal{X}_j$ the set of states x_j at time j for which (6.12)–(6.13) is feasible, i.e.

$$\mathcal{X}_j = \{x \in \mathbb{R}^n \parallel \exists u(Ex + Lu \leq M \text{ and } Ax + Bu \in \mathcal{X}_{j+1})\}, \; j = 0, \ldots, N-1 \tag{6.14}$$

$$\mathcal{X}_N = \mathcal{X}_f \tag{6.15}$$

In the following we will assume that $Q = Q^T \succeq 0, \; R = R^T \succ 0, \; P \succeq 0$, for $p = 2$. We will also denote with $\mathcal{X}_0 \subseteq \mathbb{R}^n$ the set of initial states $x(0)$ for which the optimal control problem (6.13) is feasible.
Note that we distinguish between the *current* state $x(k)$ of system (6.10) at time k and the variable x_k in the optimization problem (6.13), which is the *predicted* state of system (6.10) at time k obtained by starting from $x_0 = x(0)$ and applying to system (6.10) the input sequence $u_0, \ldots, u_{k-1}$. Analogously, $u(k)$ is the input applied to (6.10) at time k while u_k is the k-th optimization variable of the optimization problem (6.13).
If we set

$$p = 2, \; \{(x, u) \in \mathbb{R}^{n+m} \mid Ex + Lu \leq M\} = \mathbb{R}^{n+m}, \; \mathcal{X}_f = \mathbb{R}^n \tag{6.16}$$

problem (6.13) becomes the standard unconstrained finite optimal control problem whose solution can be expressed through the time varying state feedback control law

$$u^*(k) = K_k x(k), \quad k = 0, \ldots, N-1 \tag{6.17}$$

where the gain matrices K_k are given by the equation

$$K_k = -(B^T P_k + 1B + R)^{-1} B^T P_{k+1} A, \tag{6.18}$$

and where the symmetric positive-semidefinite matrices P_k are given recursively by the algorithm

$$P_N = P \tag{6.19}$$

$$P_k = A^T (P_{k+1} - P_{k+1} B (B^T P_{k+1} B + R)^{-1} B P_{k+1}) A + Q. \tag{6.20}$$

The optimal cost is given by

$$J^*(k) = x^T(0)\, P_0\, x(0). \tag{6.21}$$

If in addition to (6.16) we set $N = +\infty$ and assume that the pair (A, B) is stabilizable and the pair (C, A) is detectable, then problem (6.13)–(6.16) becomes the standard infinite time linear quadratic regulator (LQR) problem whose solution can be expressed as the state feedback control law

$$u^*(k) = Kx(k), \quad k = 0, \ldots, +\infty \tag{6.22}$$

where the gain matrix K is given by

$$K = -(B^T P B + R)^{-1} B^T P_\infty A \tag{6.23}$$

and where P_∞ is the unique solution of the algebraic matrix equation

$$P_\infty = A^T (P_\infty - P_\infty B (B^T P_\infty B + R)^{-1} B P_\infty) A + Q \tag{6.24}$$

within the class of symmetric positive-semidefinite matrices.
In the following we show that the solution to problem (6.13) can again be expressed in feedback form where $u^*(k)$ is a continuous piecewise affine function on polyhedra of the state $x(k)$.

6.4.2 State Feedback Solution of CFTOC

By substituting

$$x_k = A^k x_0 + \sum_{j=0}^{k-1} A^j B u_{k-1-j} \tag{6.25}$$

the optimization problem (6.13) can be rewritten in the form

$$\begin{aligned} J^*(x(0)) = \tfrac{1}{2}\, x^T(0) Y x(0) \quad + \quad \min_{U_N} \quad & \tfrac{1}{2}\, U_N^T H U_N + x^T(0) F U_N \\ \text{subj. to} \quad & G U_N \leq W + E x(0) \end{aligned} \tag{6.26}$$

where $H = H^T \succ 0$, H, F, Y, G, W, E are obtained from P, Q, R, (6.13) and (6.25), and it follows from the previous assumptions that

$$\begin{bmatrix} Y & F^T \\ F & H \end{bmatrix} \succeq 0.$$

Note that the optimizer U_N is independent of the term involving Y in (6.26).
We view $x(0)$ as a vector of parameters and our goal is to solve (6.26) for all values of interest, and to make this dependence *explicit*. Note that the set $\mathcal{X}_0$ is a polyhedron and can be computed by projecting the polyhedron

$$\mathcal{P}_0 = \{(U_N, x(0)) \in \mathbb{R}^{s+n} \mid G U_N \leq W + E x(0)\}$$

on the $x(0)$-space.
Before proceeding further, it is convenient to define

$$z \triangleq U_N + H^{-1}F^T x(0), \tag{6.27}$$

$z \in \mathbb{R}^s$, and to transform (6.26) by completing squares to obtain the equivalent problem

$$\begin{aligned} J_z^*(x(0)) = \quad & \min_z \quad \tfrac{1}{2} z^T H z \\ & \text{subj. to} \quad Gz \leq W + Sx(0) \end{aligned} \tag{6.28}$$

where $S \triangleq E + GH^{-1}F^T$, and $J_z^*(x(0)) = J^*(x(0)) - \frac{1}{2} x^T(0)(Y - FH^{-1}F^T)x(0)$. In the transformed problem the parameter vector $x(0)$ appears only in the right hand side of the constraints.
Problem (6.28) is a multi-parametric quadratic program that can be solved using **Algorithm 2** described in section 6.3.3. Once the multi-parametric problem (6.28) has been solved for a polyhedral set $X \subset \mathbb{R}^n$, the solution $U_N^* = U_N^*$ of CFTOC (6.13) and therefore $u^*(0) = u^*(x(0))$ is available explicitly as a function of the initial state $x(0)$.
Theorem 13 states that the solution $z^*(x(0))$ of the mp-QP problem (6.28) is a continuous and piecewise affine function on polyhedra of x. Clearly the same properties are inherited by the controller. The following Corollaries of Theorem 13 establish the analytical properties of the optimal control law and of the value function.

Corollary 1. *The control law $u^*(0) = f_0(x_0),\ f_0 : \mathbb{R}^n \to \mathbb{R}^m$, obtained as solution to the optimization problem* (6.13) *is continuous and piecewise affine on polyhedra*

$$f_0(x) = F_0^i x + g_0^i \quad \textit{if} \quad x \in CR_0^i, \quad i = 1, \ldots, N_0^r \tag{6.29}$$

where the polyhedral sets $CR_0^i = \{x \in \mathbb{R}^n \mid H_0^i x \leq K_0^i,\ i = 1, \ldots, N_0^r\}$ are a partition in the broad sense of the feasible polyhedron $\mathcal{X}_0$.

Corollary 2. *The value function $J^*(x(0))$ obtained as solution to* (6.13) *is convex and piecewise quadratic on polyhedra.*

The solution to the multi-parametric problem (6.28) provides the state feedback solution $u^*(k) = f_k(x(k))$ of CFTOC (6.13) for $k = 0$ and it also provides the open loop optimal control laws $u^*(k)$ as function of the initial state, i.e., $u^*(k) = u^*(k)(x(0))$. The state feedback PPWA optimal controllers $f_k : x(k) \mapsto u^*(k)$ for $k = 1, \ldots, N$ are computed in the following way. Consider the same CFTOC (6.13) over the shortened horizon $[i, N]$

$$\begin{aligned} \min_{U_{N-i}} \quad & \|Px_N\|_p + \sum_{k=i}^{N-1} \|Qx_k\|_p + \|Ru_k\|_p \\ \text{subj. to} \quad & Ex_k + Lu_k \leq M, \quad k = i, \ldots, N-1 \\ & x_N \in \mathcal{X}_f \\ & x_{k+1} = Ax_k + Bu_k, \quad k \geq 0 \\ & x_i = x(i) \end{aligned} \tag{6.30}$$

where $U_{N-i} \triangleq [u_i^T, \ldots, u_{N-1}^T]^T$. We will denote with $\mathcal{X}_i \subseteq \mathbb{R}^n$ the set of initial states $x(i)$ for which the optimal control problem (6.30) is feasible and with U^*_{N-i} its optimizer. Problem (6.30) can be translated into the mp-QP

$$\begin{array}{ll} \min\limits_{U_{N-i}} & \frac{1}{2} U_{N-i}^T H U_{N-i} + x^T(i) F U_{N-i} \\ \text{subj. to} & G U_{N-i} \leq W + E x(i). \end{array} \tag{6.31}$$

The first component of the multi-parametric solution to (6.31) has the form

$$u^*(i) = f_i(x(i)), \quad \forall x(i) \in \mathcal{X}_i, \tag{6.32}$$

where the control law $f_i : \mathbb{R}^n \to \mathbb{R}^m$, is continuous and PWA

$$f_i(x) = F_i^j x + g_i^j \quad \text{if} \quad x \in CR_i^j,\ j = 1, \ldots, N_i^r \tag{6.33}$$

and where the polyhedral sets $CR_i^j = \{x \in \mathbb{R}^n \mid H_i^j x \leq K_i^j\},\ j = 1, \ldots, N_i^r$ are a partition in the broad sense of the feasible polyhedron $\mathcal{X}_i$. Therefore, the feedback solution $u^*(k) = f_k(x(k)),\ k = 0, \ldots, N-1$ to the CFTOC (6.13) is obtained by solving N mp-QP problems.

6.5 MPC for the semiactive suspension system

In this section we present the partition of the state space for the semiactive suspension system applying the explicit MPC desing method. The simulations were done with MATLAB, for which exists a free toolbox called $\mathbb{MPT}$ [16].
Computing explicit state feedback controllers via multi-parametric programming may easily lead to controllers with prohibitive complexity, both in runtime and solution. There are three aspects which are important in this respect: performance, closed-loop stability and constraint satisfaction. The $\mathbb{MPT}$ toolbox provides several possibilities to compute the controller and the partition of the state space, which are specified below:

Finite Time Optimal Control (FTOC) This method yields the finite time optimal controller (FTOC), i.e. the performance will be N-step optimal but may not be infinite horizon optimal. The complexity of the controller depends strongly on the prediction horizon N, the larger N the more complex the controller. Furthermore, within this method, the $\mathbb{MPT}$ toolbox provides three different modes:

- `probstruct:Tconstraint=0`: No terminal set constraint. The controller will be defined over a superset of the maximum controllable set (i.e. all states, which are controllable), but no guarantees on stability or closed-loop constraint satisfaction can be given. As the prediction horizon N is increased the feasible set of states will converge to the maximum controllable set from "the outside-in", i.e. the controlled set will shrink as N increases.
  Even though closed loop stability and constraint satisfaction are not guaranteed, $\mathbb{MPT}$ provides a function to extract the set of states which satisfy the constraints for all time and another fuuction to analyze these states for stability.

- `probstruct:Tconstraint=1`: Stabilizing terminal set is automatically computed. The resulting controller will guarantee stability and constraint satisfaction for all time, but will only cover a subset of the maximum controllable set of states. By increasing the prediction horizon, the controllable set of states will converge to the maximum controllable set from "the inside-out", i.e. the controlled set will grow larger as N increases.
- `probstruct:Tconstraint=P`: User defined terminal set. Depending on the properties of the set P, any combination of the two cases previously described may occur.

Infinite Time Optimal Control (ITOC) This method yields the infinite time optimal controller, i.e. the best possible performance for the control problem. Asymptotic stability and constraint satisfaction are guaranteed and the maximum controllable set will be covered by the resulting controller. However, the controller's complexity may be prohibitive and the computation may take a very long time.

Minimum Time Control This method yields the minimal time controller with respect to a target set around the origin, i.e. the controller will drive the state into this set in minimal time. In general, the complexity of minimum time controllers is significantly lower than that of their $1/2/\infty$-norm cost optimal counterparts. The controller is guaranteed to cover all controllable sets and asymptotic stability and constraint satisfaction are also assured.

Low Complexity Control This method yields a controller for a prediction horizon $N = 1$ with additional constraints that guarantees asymptotic stability and constraint satisfaction for the closed loop system. The controller covers all controllable states. The complexity of this 1-step controller is generally significantly lower than all other control schemes in $\mathbb{MPT}$ which cover the maximal controllable set, but also in this case the computation requires a long time.

Applying the MPC to the suspension system, we were only able to obtain partitions for the finite time optimal control and the infinite time optimal control. For the suspension system it was not possible to compute the partitions using the time optimal control and the low complexity control, because the computation did not finish in an adequate time. The partitions for the FTOC and the ITOC will be shown below.
In order to limit the computational complexity we chose the following bounded polyhedron

$$K = \{x \in \mathbb{R}^4 \mid \mid x_i \mid \leq 1,\ i = 1, \ldots, 4\}$$

as the set of states, that are of interest for the fourth-order suspension model.
Furthermore, the weight matrices Q and R we have chosen are

$$Q = \begin{bmatrix} 11 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad R = 0.8 \cdot 10^{-9}$$

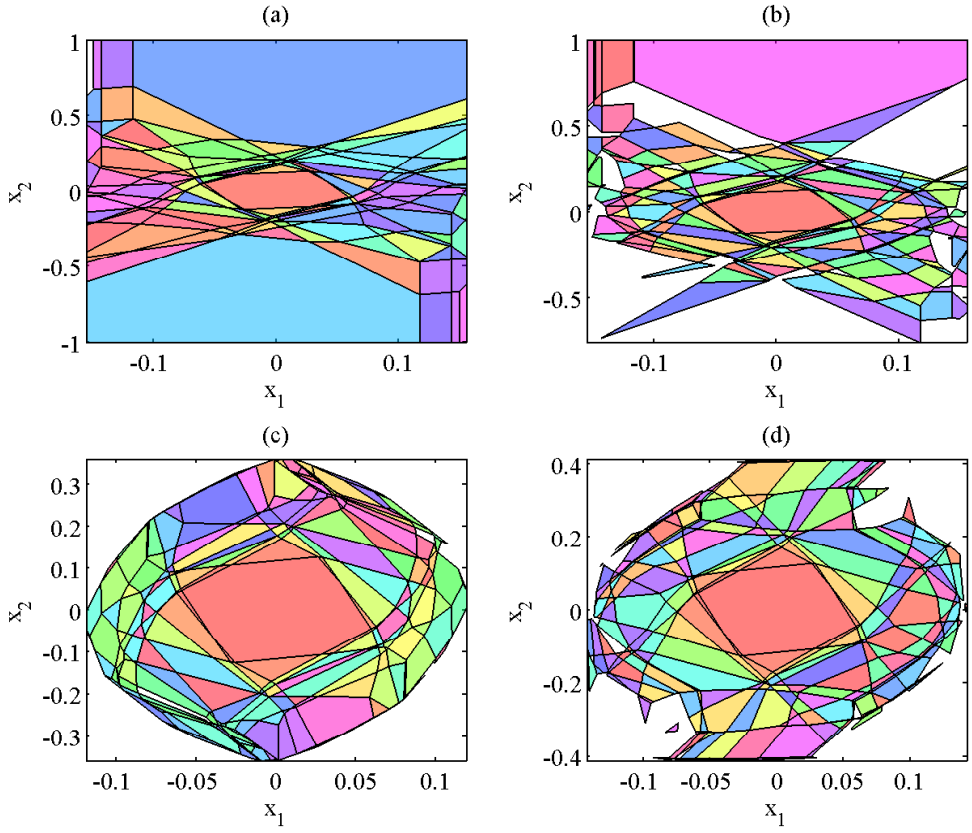

Figure 6.2: Partition for the fourth-order suspension model, cut through $x_3 = x_4 = 0$: (a) FTOC, no terminal set, $N = 10$, 557 regions; (b) FTOC, no terminal set, $N = 15$, 1038 regions; (c) FTOC , terminal set automatically computed, $N = 10$, 2195 regions; (c) FTOC , terminal set automatically computed, $N = 15$, 3852 regions.

and we assumed the sampling interval $T = 0.01$ s.
The resulting partitions, more precisely a cut through the partitions at $x_3 = x_4 = 0$, are depicted in Figure 6.2. The FTOC setting `probStruct. Tconstraint=0` does not guarantee stability and closed loop constraint satisfaction and as it can be seen in Figure 6.2 (a) and (b) increasing the prediction horizon N from 10 to 15 shrinks the controlled set, so that it converges towards the maximum controllable set from the outside inwards. The projections of the partitions for the FTOC employing `probStruct.Tconstraint=1` are illustrated in Figure 6.2 (c) and (d) for the prediction horizon $N = 10$ and $N = 15$, respectively. As mentioned above, by increasing the prediction horizon N the controllable set should converge to the maximum controllable set from the inside outwards. Obviously, this does not hold true for the partition of the suspension system, because parts of the state space, that has been covered by the partition with $N = 10$ are not covered anymore

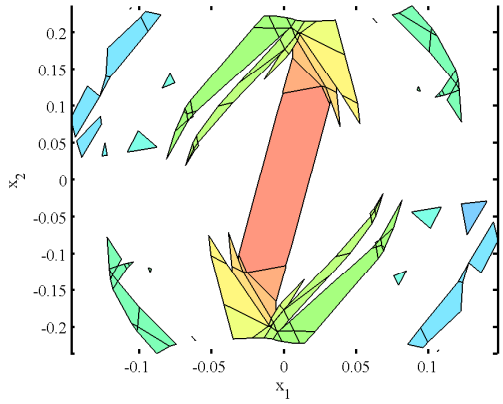

Figure 6.3: Projection of the partition into the x_1-x_2-plane for the fourth-order suspension system for the ITOC

considering $N = 15$, as visualized in Figure 6.2(d). In theory, employing this mode of partitioning the state space, it is not possible, that a state is controllable under a given prediction horizon, but does not maintain this property after increasing the latter.
These results are probably due to numerically problems of the algorithm.

We also applied the ITOC to the suspension system, where the problems, that occured above became even more obvious. Figure 6.3 illustrates the resulting partition for the infinite horizon problem for the fourth-order suspension model. Only very few parts of the state space that were identified to be controllable, (refer to Figure 6.2 (c)), are covered by the partition computed under the infinite prediction horizon, although this method should yield a controller that covers all controllable states.

Chapter 7

Applying OGS, discontinuous VSC and explicit MPC to the suspension system: A comparison

7.1 Introduction

In this chapter we compare the three different control design methods we presented in chapters 4, 5 and 6 applying them to the suspension system illustrated in Chapter 3. All of the three methods (OGS, dVSC and eMPC) are based on off-line partitioning of the state space, assign subcontrollers to these regions – linear subcontrollers for the OGS and the dVSC, affine subcontrollers for the MPC – and switch between these subcontrollers during the on-line regulation cycles according to the current state.
For the OGS approach we will consider two different strategies:

OGS Case A: We will recall some of the simulation results presented in [11] and therefore chose the parameters as described in Section 4.3.

OGS Case B: We considered the same set of closed loop matrices for the OGS and the dVSC approach in order to obtain a more immediate and more meaningful comparison. Therefore we determined the feedback gain matrices such that the closed loop eigenvalues are equal to the eigenvalue distribution shown in Figure 7.1.

This chapter is structured as follows: First we illustrate the differences of the results of partitioning the state space for the suspension system. In a second step we will compare simulation results with respect to the control system's performance both for the active and the semiactive application to the suspension system.

7.2 Partitioning the State Space

In this section we will compare the partitions of the state space, that are the results of the off-line phase of the three design methods OGS, dVSC and eMPC. First, we will compare

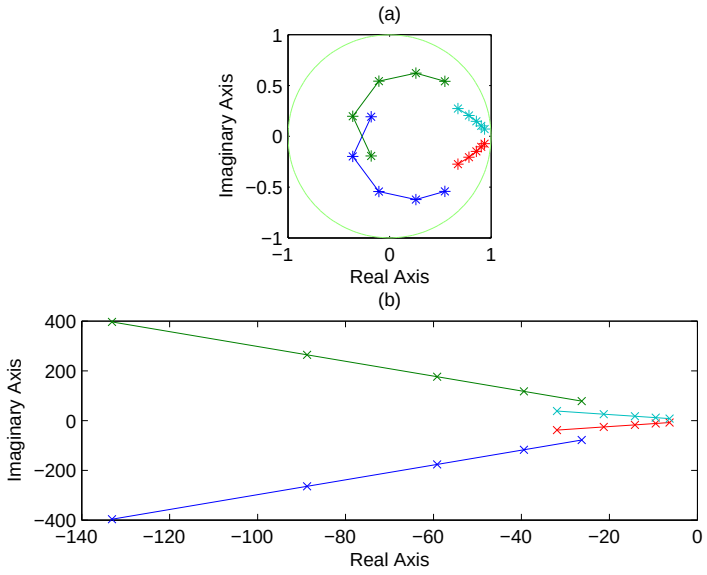

Figure 7.1: Sets of designated eigenvalues for the fourth-order suspension system (a) in the z-plane and (b) in the s-plane. $h = 1.5$

the partitions considering the second-order one-degree of freedom model of the suspension system. Since the state space here is only two-dimensional it is possible to depict the partitions completely. Further, we will proceed illustrating a cut at $x_3 = x_4 = 0$ through the partitions of the fourth-order two-degrees of freedom suspension model, since simulation results for the fourth-order model are our main interest.

In order to obtain a comparison as meaningful as possible, we made the following assumptions:

- For the OGS and for the MPC design method we considered the discretized model of the suspension system as it has been presented in Chapter 3 assuming the sampling interval $T = 0.01$ s.
- For the dVSC we considered the continuous time model (see Chapter 3), as the theory presented in Chapter 5 requires the latter.
- We have chosen the same sequence of closed loop eigenvalues (refer to (5.13)) for the OGS Case B and the dVSC in the z-plane and in the s-plane respectively. According to these sets of eigenvalues we determined the closed loop matrices via pole placement. Note that the closed loop matrices obtained in this manner do not guarantee, that the Yoshida regions are nested for the OGS Case B. Figure 7.1 shows the sequence of the designated sets of eigenvalues for the fourth-order model of the suspension system in the s-plane and in the z-plane respectively.
- In order to limit the run times of the computation of the convex polyhedral regions employing explicit MPC we considered the following bounded polyhedron for the

second-order suspension model

$$X = \{x \in \mathbb{R}^2 \mid |x_i| \leq 1,\ i = 1, 2\}. \tag{7.1}$$

Analogously, we determined the partition of the state space for the fourth-order suspension model considering the bounded polyhedron

$$X = \{x \in \mathbb{R}^4 \mid |x_i| \leq 1,\ i = 1, \ldots, 4\} \tag{7.2}$$

as the state space of interest. Furthermore, we considered the FTOC with a prediction horizon $N = 10$ and we set `probStruct.Tconstraint=1` to obtain a controller that guarantees closed loop stability and constraint satisfaction for all times.

- For the OGS Case A and the explicit MPC we considered the same weight matrix on the states Q and we chose the weight on the input for the eMPC equal to the weight on the input for the OGS Case A divided by $\rho_{\max}$

  $$Q_{\text{MPC}} = Q_{\text{OGS}}, \quad R_{\text{MPC}} = \frac{R_{\text{OGS}}}{\rho_{\max}}$$

  This guarantees for both approaches the same level of optimality.

Below we will present and comment the partitions, which we computed with MATLAB. The results for the second-order model and for the fourth-order model of the suspension system are depicted in Figure 7.2 and Figure 7.3 respectively.

Figure 7.2 shows the partitions for the second-order suspension model of the four considered cases with the same scale. The Yoshida regions for the OGS Case A and B are depicted in Figure 7.2 (a) and (b), respectively, whereas (c) and (d) illustrate the elliptic Lyapunov regions resulting from discontinuous VSC and the convex polyhedral partitions computed with explicit MPC.

Comparing Figure 7.2 (b) and (c) it can be seen, that the Yoshida regions of the OGS Case B are much larger, i.e. cover much more of the state space, than the Lyapunov regions of the discontinuous VSC. Here, for the OGS Case B the linear regions are nested and therefore we can compare the simulations of the resulting controllers. Figure 7.2 (d) depicts the partition of the convex polyhedral regions of the eMPC, which are in the x_2-direction constrained by the assumptions we made (compare (7.1)), but in the x_1-direction cover less of the state space compared to the OGS Case A and Case B. The size of the resulting partitions is important with regard to robustness towards varying initial conditions and disturbances, e.g. caused by the road profile.

Figure 7.3 depicts a cut through $x_3 = x_4 = 0$ of the partitions for the fourth-order suspension model resulting from the OGS Case A and the explicit MPC. Regarding Figure 7.3 the difference of the size of the partitions becomes even more obvious than in the second-order case, the Yoshida regions cover a much larger subset of the state space of interest than the partition of the explicit MPC.

For the fourth-order model we take into account only these two approaches, because the Lyapunov regions obtained by the discontinuous VSC were too small to cover the state

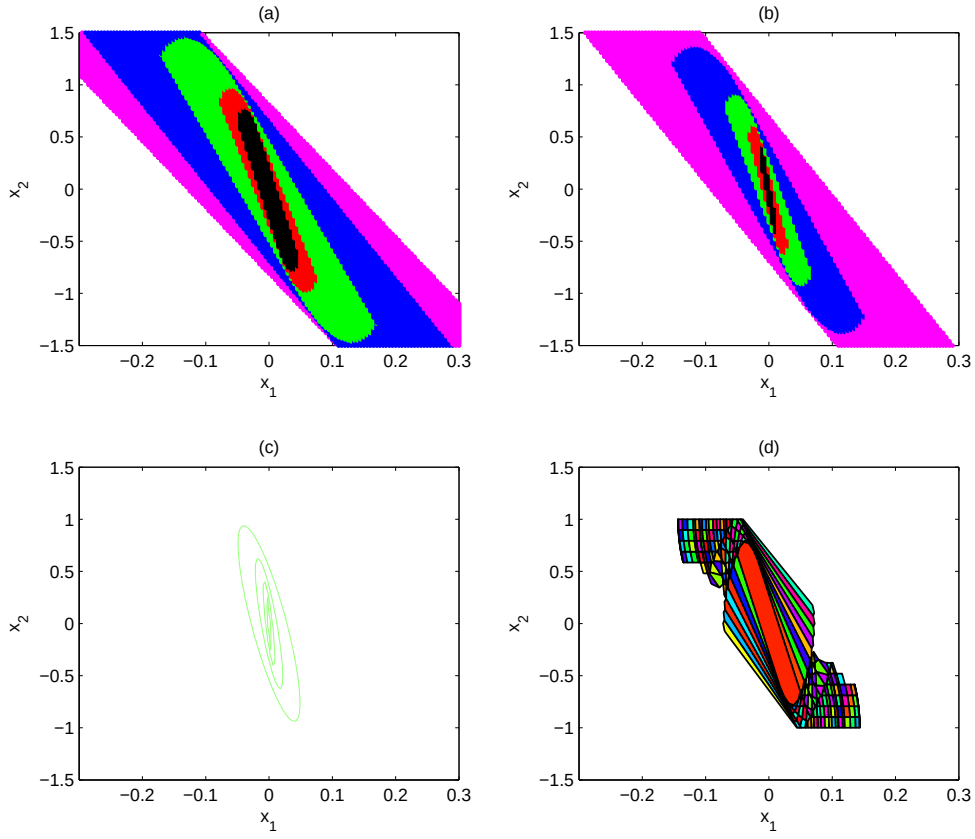

Figure 7.2: Partition of the state space for the second-order model. (a) linear regions of OGS Case A (b) linear regions of OGS Case B (c) elliptical Lyapunov regions of discontinuous VSC (d) convex polyhedral regions of explicit MPC

space of interest. Since we considered the OGS Case B only with respect to a meaningful comparison with the discontinuous VSC we do not look at this case for the fourth-order model. Furthermore, the Yoshida regions for the OGS Case B were not nested for the fourth-order model and therefore this way of computing the partition of the state space does not lead to an applicable controller.
The differences of the size of the partitions are due to the following reasons:

- The size of the Yoshida regions depends on the sequences of closed loop matrices, that determine the linear regions.

- For the dVSC the size of the linear regions depends on the R_p and therefore on the choice of the Q_p in the Lyapunov equation (5.8). Because the c_p, that determines the Lyapunov regions, is a function of R_p^{-1}, the regions seem to be nearly independent of

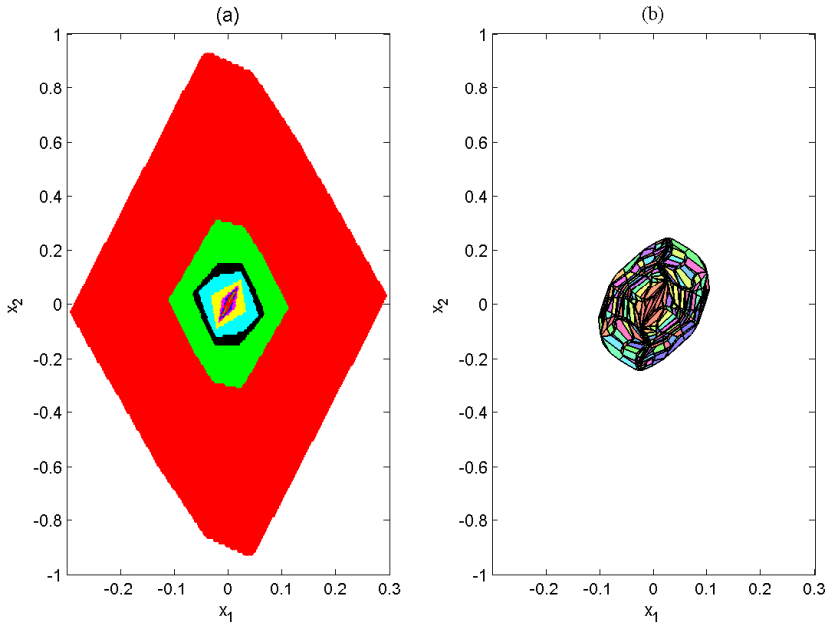

Figure 7.3: Partition of the state space for the fourth-order suspension model: cut through $x_3 = x_4 = 0$. (a) linear regions of OGS Case A (b) convex polyhedral regions of explicit MPC

the choice of Q_p. We have not been able to enlarge the Lyapunov regions significantly by varying Q_p (see Figure 7.3 (c)).

- For the explicit MPC the size of the feasible set depends on the constraints on the states, which are defined by the polyhedron K in (7.1) and (7.2), on the mode chosen to compute the controller and in the case of the FTOC mode also on the prediction horizon N.

Comparing the different approaches we can summarize the following results:

- The Lyapunov regions obtained by the discontinuous VSC cover only a small subset of the state space of interest, but the computational effort is very low.

- The Yoshida regions obtained by the OGS Case B are significantly larger than the linear regions obtained by the discontinuous VSC for the same sequence of closed loop eigenvalues, but the nesting condition is not necessarily fulfilled.

- The partition obtained by the explicit MPC covers a larger set of states than the discontinuous VSC, but the resulting partitions are still significantly smaller than those of the OGS Case A and meanwhile the computational effort is significantly higher.

- The OGS Case A seems to lead to the best results both with respect to the set of states covered by the linear regions and the computational effort.

In the next section we compare the simulation results applying the different design approaches to the suspension model both the active and the semiactive performance.

7.3 The control performance

In this section we compare the performance of the active controller obtained with the OGS Case A and B, the discontinuous VSC and the explicit MPC for the second-order suspension model. Later on we will present simulation results for the OGS Case A and the explicit MPC applying both the active and the semiactive controller to the fourth-order suspension model.

7.3.1 The second-order model

For the second-order suspension model we computed the system's evolution for the initial state $x_0 = [0.01\ 0.1]^T$ with MATLAB.

Active suspension

The simulation results for the active suspension system are illustrated in Figure 7.4, where the blue line and the red line represent the OGS Case A and Case B, respectively, the discontinuous VSC is depicted with the green line and the magenta line shows the evolution of the explicit MPC.
It is visualized in Figure 7.4 that under the assumptions we made the OGS Case A and the explicit MPC obtain almost exactly the same evolution of the states and the input.
The results for the discontinuous VSC are not satisfying compared to the other approaches; this is due to the fact that it does not yield a good exploitation of the allowed maximal control input.
The controller obtained by the OGS Case B yields even a better exploitation of the range of the allowed control input than the OGS Case A and the explicit MPC and therefore regulates the state faster to the origin, but it cannot be compared directly to the other approaches as the controller in this case is not determined minimizing a performance index.
Figure 7.4 indicates the regions, which the current state belongs to for the OGS Case A and B and the discontinuous VSC. Here 1 denotes the largest and 5 the smallest region, compare Figure 7.2.

7.3.2 The fourth-order model

For the fourth-order model we simulated the system's evolution considering initial conditions different from zero and also the performance for disturbances which may be caused by the road profile. Here we concentrate on the OGS Case A and the explicit MPC and, therefore, we leave out the "Case A".

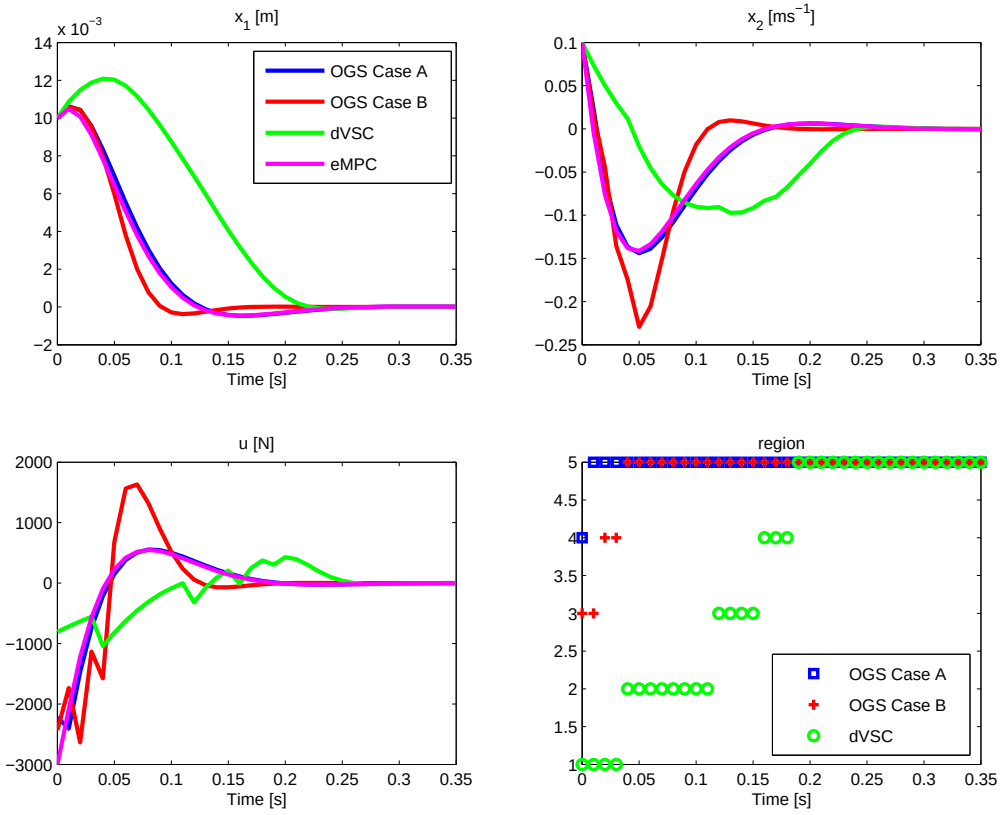

Figure 7.4: Evolution of the second-order suspension system comparing OGS Case A and B, discontinuous VSC and eMPC. Initial state $x_0 = [0.01\ 0.1]^T$.

Active suspension

The simulation results for the fourth-order suspension model with regard to the initial state $x_0 = [0.015\ 0.1\ 0\ 0]^T$ are depicted in Figure 7.5. The evolution for the OGS is visualized with blue, whereas magenta shows the evolution for the explicit MPC. Since our main attention is directed towards the passenger's comfort, we depicted only the evolution for the sprung mass here. The maximum amplitude of the sprung mass velocity is relatively high. This is due to the fact that we did not impose weights on the velocities in the optimal control problem, what should be done in a further step as a high velocity up- or downwards is in conflict with a pleasant ride.

Like for the second-order model we obtain almost exactly the same performance applying the OGS and the explicit MPC to the suspension system. As it can be seen in Figure 7.5 the resulting active control force u is very similar for the OGS and the explicit MPC.

In addition to these shock test simulations, we ran simulations assuming zero initial conditions but adding a disturbance that may represent the road profile. In order to simulate

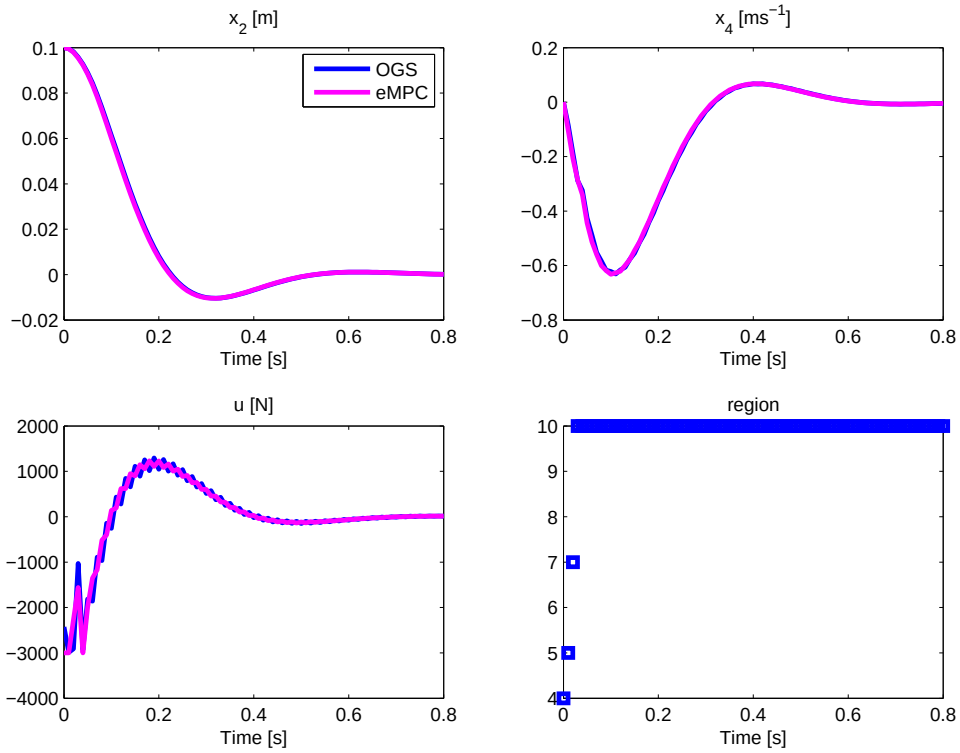

Figure 7.5: Evolution of the fourth-order suspension system comparing OGS and explicit MPC. Initial state $x_0 = [0.015\ 0.1\ 0\ 0]^T$.

the disturbance, we chose the same parameters as presented in [11].
The evolution is shown in Figure 7.6, where the green line represents the added disturbance. Like in the previous cases the performance of the OGS and the explicit MPC are similar. The simulation results show that the suspension filters the high frequencies smoothing the movement of x_2, i.e. the evolution of the sprung mass.

Semiactive suspension

In this section we compare the simulation results for the semiactive fourth-order suspension model, that was introduced in Chapter 3. Recall that the feedback control law in this case

$$u_{\text{sem}}(k) = -\underbrace{[-\lambda_s \quad \lambda_s \quad -f(k) \quad f(k)]}_{K} x(k) \tag{7.3}$$

is determined by approximating the active control law according to

$$f(k) = \min_{f \in [f_{\min}, f_{\max}]} \arg F[f, x(k)] = \begin{cases} f_{\max} & \text{if} \quad f^*(k) > f_{\max} \\ f^*(k) & \text{if} \quad f^*(k) \in [f_{\min}, f_{\max}] \\ f_{\min} & \text{if} \quad f^*(k) < f_{\min} \end{cases} \tag{7.4}$$

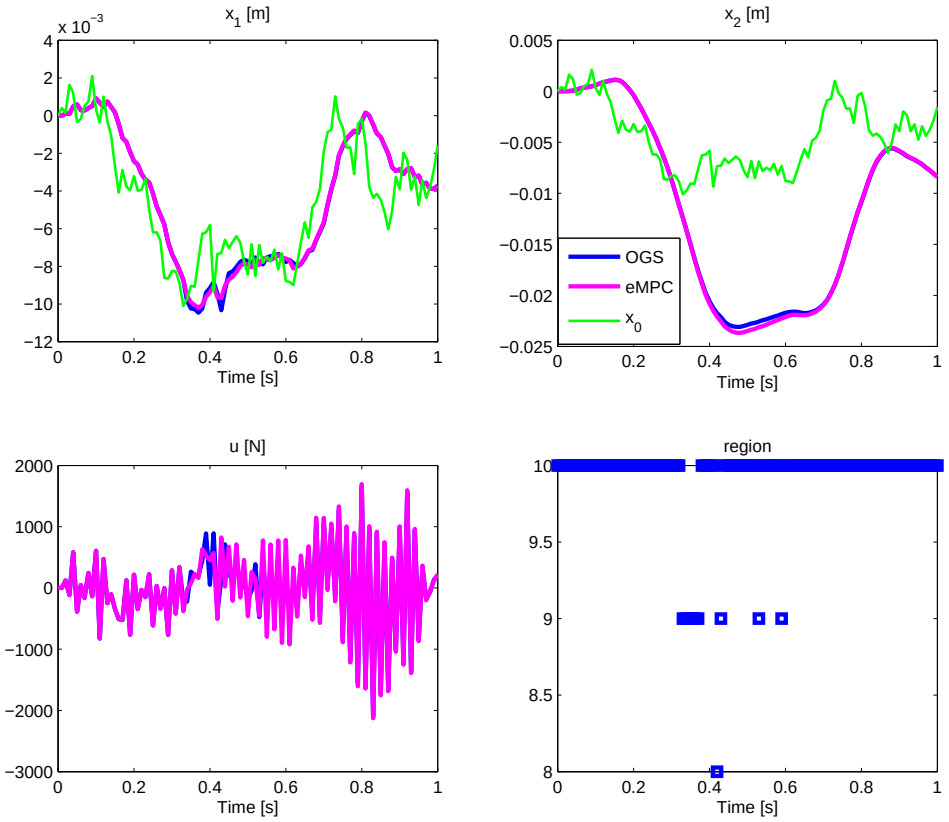

Figure 7.6: Evolution for the fourth-order suspension system with an additive disturbance x_0.

where the $f^*(k)$ is chosen that the squared difference of the active and semiactive control law is minimized.

In a first step we compare the present simulation results for the semiactive suspension system considering the shock test with the initial state $x_0 = [0.015\ \ 0.1\ \ 0\ \ 0]$. Like for the active suspension model, we only present the evolution of the sprung mass as we are mostly interested in the passenger's comfort property of the suspension.

Figure 7.7 shows the evolution of the semiactive suspension system compared to those of the active suspensions. Again the OGS and the explicit MPC performances are very similar as the states at each time instant only differ in the order of magnitude of $|x_{i,\mathrm{OGS}} - x_{i,\mathrm{eMPC}}| \approx 10^{-10}$.

In Figure 7.7 down to the left the evolution of the target control laws computed with the OGS and the explicit MPC are compared to those control laws that are "really" applied to the system by the semiactive suspension adjusting the damping coefficient f according to (7.3) and (7.4), whose evolution is visualized in Figure 7.7 down to the right.

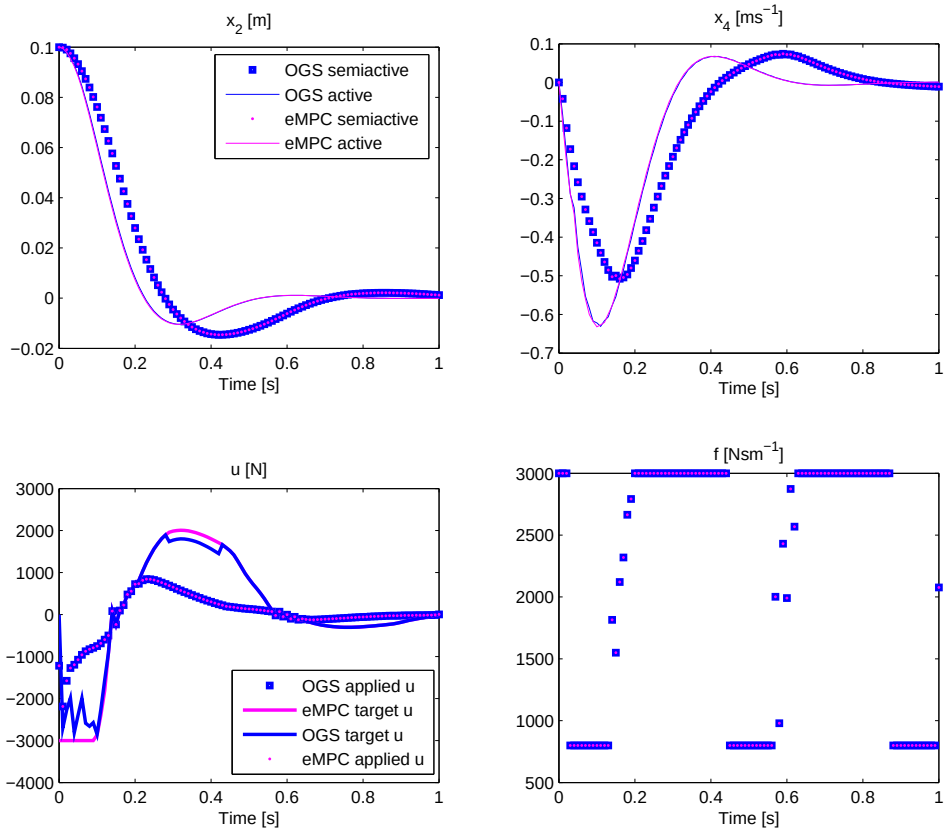

Figure 7.7: Evolution for the fourth-order semiactive suspension system with initial state $x_0 = [0.015\ 0.1\ 0\ 0]$.

The last simulations we present here were run considering the fourth-order semiactive suspension model with regard to zero initial conditions, but considering an additive disturbance x_0.
Figure 7.8 depicts the evolution of the wheel displacement, the sprung mass displacement and the control input both for the active and the semiactive suspension model. As it can be seen from these simulations substituting the actuator by an semiactive suspension leads to a reasonable performance, especially when considering disturbance models.

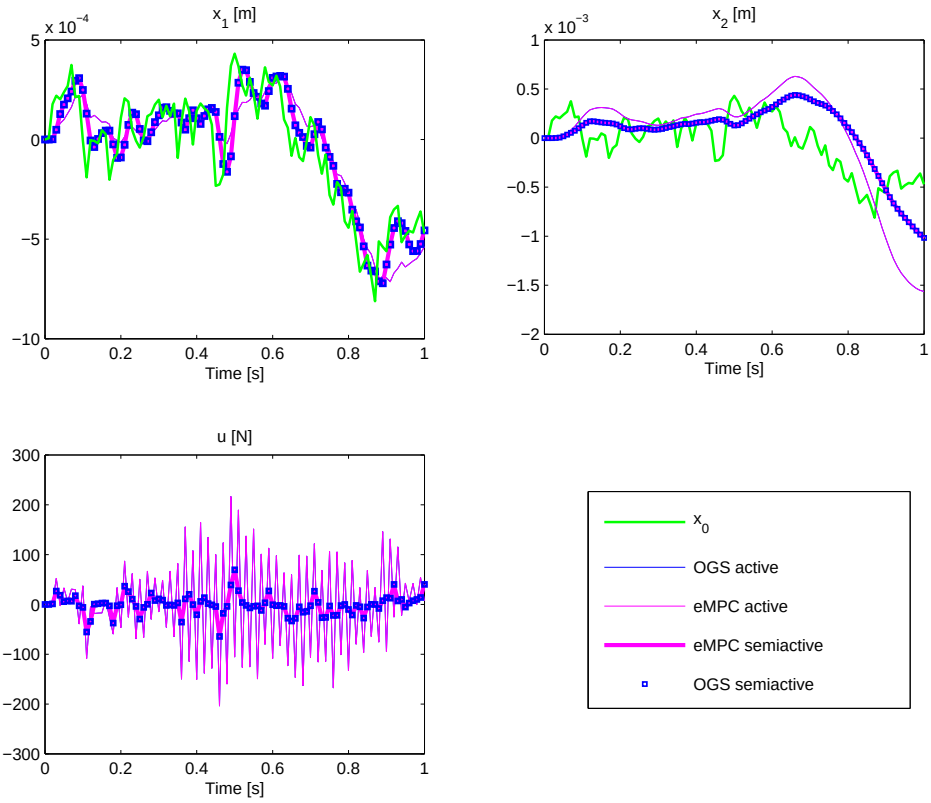

Figure 7.8: Evolution for the fourth-order semiactive suspension system with an additive disturbance x_0.

Chapter 8

Conclusions

In this thesis we aimed at comparing three different control design methods applying them to a semiactive suspension system. After summarizing some principles of control theory in Chapter 2, we introduced the quarter-car suspension model in Chapter 3.
The following chapters were dedicated to the theory of the three different methods, optimal gain switching, discontinuous variable structure control and explicit model predictive control. Common to all of these methods is the division into an off-line and an on-line phase of the design process. The off-line phases partition the state space into convex regions and assign linear or affine subcontrollers to each region. The investigated approaches all intend to approximate time optimal control laws, and thus we call them suboptimal. During the on-line phase the controller switches between these subcontrollers according to the current state.
The simulation results of the application of these concepts to the semiactive suspension model were compared in Chapter 7.

Here we briefly summarize the comparison of the OGS, discontinuous VSC and the explicit MPC.

off-line phase The best results from the partitioning phase of the design process were obtained by the optimal gain switching method as it lead to the largest subset of controllable states for the state space of interest and the computational time was relatively low.
For the discontinuous variable structure control the computational effort is also very low, but we were not able to compute Lyapunov regions, which covered the set of states of interest.
The explicit model predictive control provided a partition that was sufficiently large but still did not reach the size of the Yoshida regions of the OGS. The main drawback of eMPC was that its computational times were exhaustive under distinct assumptions, like guaranteed closed loop stability and large prediction horizons.

on-line phase Comparing the performance of the three methods we obtained almost exactly the same evolution for the OGS and the explicit MPC. To underline this fact we computed a performance index for the OGS and the eMPC, whose results

	OGS	**eMPC**
cost	6.2138	6.3057
computational time	6 s	1 h 45 min

Table 8.1: Comparison of the OGS and the explicit MPC for the fourth-order suspension system.

are given in Table 8.1.
The simulation results for the discontinuous VSC were not satisfying, because the controller did not lead to settling times that were as short as those of the other approaches.

Finally, we conclude that the optimal gain switching and the explicit model predictive approaches provide controllers are equivalent in terms of their performance, but the effort to compute the controller for the explicit MPC is way higher than for the OGS. Regarding the robustness of the controllers, the OGS method seems to be advantageous.

A number of challenges remain and appear from these investigations. For the suspension system it may lead to interesting results to run further simulations imposing also weights on the velocities of the sprung and the nonsprung mass. Moreover, it is an interesting challenge to prove the equivalence of the OGS and the explicit MPC more formally.

Acknowledgements

There are many people I would like to thank for their help and assistance they gave me during the work for this thesis.
Firstly, I would like to thank Dr. Carla Seatzu for being so patient during the semester I spend at the University of Cagliari. It has been a pleasure for me to have the opportunity to work with her as a tutor and friend.
Secondly, I would like to thank Prof. Alessandro Giua for offering me the possibility to work with him and for his ideas and support he gave me.
Without one person, I would never have thought about spending one semester on the wonderful island Sardinia. Therefore, I want to thank Prof. Raisch for arranging the contact with the University of Cagliari and especially with Prof. Giua.
Furthermore, I would like to thank Michal Kvasnica for his immediate and very helpful support concerning the model predictive toolbox for MATLAB.
Finally, I want to thank all my friends and my family for supporting me during the writing process of my first thesis. Especially, I would like to thank Thomas, Angela, Barbara, Daniele, Eleonora, Federica, Ines, Jasmina, Kamila, Laura, and all the other friends, who were close to me during my period in Cagliari.

Appendix A

MATLAB Programs

A.1 The system's data

```
% data file for the fourth-order suspension system

% parameters of the suspension system

l=155900;
lambdas=14345;
M1=28.58;
M2=288.9;
umax=3000;
fmin=800;
fmax=3000;

% matrices of the state space representation

A=[0 0 1 0; 0 0 0 1; -l/M1 0 0 0; 0 0 0 0];
B=[0 0; 0 0; -1/M1 l/M1; 1/M2 0];  %B'=[B|L]
C=[1 0 0 0; 0 1 0 0];

% weight matrices

Q=[11 -1 0 0; -1 1 0 0; zeros(2,4)];
R=.8*10^(-9);
RO=[.01 .1 .5 1
4 20 50 100 1000 10^5];

% discretization of the system

Ts=0.01;
[G,H1,Cd,Dd]=c2dm(A,B,C,zeros(2,2),Ts,'zoh');
H=H1(1:4,1);
Ltilde=H1(1:4,2);
```

A.2 Partitioning the State Space

A.2.1 Computation of the Yoshida Regions

```
% loading the data file
Dati;

% initialisation
[dim,col]=size(G); Jzi=[1]; fine=0;

% construction of the linear regions

for r=1:nr
    % determine the feedback gain matrices
    K(r,:)=place(G,H,eigvals_vsc_discrete(r,:));
    Gcloop=G-H*K(r,:);
    % define GG containing the closed loop matrices
    GG(:,(r-1)*dim+1:r*dim)=Gcloop;
    Z(:,1)=K(r,:)';
    j=0;
    how='ok';
    D = [];
    % determine the value j_0 by solving a LP
    while how>0
         j=j+1;
         Z(:,j+1)=((Gcloop')^j)*Z(:,1);
         for i=1:j
                  D(:,i)=Z(:,i)-Z(:,j+1);
                  D(:,i+j)=-Z(:,i)-Z(:,j+1);
         end
         [m,n]=size(D);
         b=zeros(m,1);
         c=-ones(n,1);
         [x,fval,how]=linprog(c,[],[],D,b,zeros(n,1),[]);
    end
    % determine the matrix Z_rho
    J(r)=j;
    Z=Z(:,1:j);
    if r~=1,
          Jzi(r)=Jzi(r-1)+J(r-1);
    end
    Jzf(r)=j+fine;
    fine=Jzf(r);
    % define ZG containing all Z_rho
    ZG(:,Jzi(r):Jzf(r))=Z;
end
```

A.2.2 Computation of the Lyapunov Regions

```
% loading the data file
dati;

% initialisation
[dim,col]=size(G);

%define the Q_p for the Lyapunov equation
Q_p=[10 0 0 0; 0 10 0 0; 0 0 .1 0;0 0 0 .1];

for r=1:nr
    % determine the feedback gain matrices
    K(r,:)=place(A,B(:,1),eigvals_vsc(r,:));
    Gcloop=A-B(:,1)*K(r,:);
    GG(:,(r-1)*dim+1:r*dim)=Gcloop;
    Qp=Q_p;
    % compute Rp from the Lyapunovequation
    Rp=lyap(Gcloop',Qp);
    R_p(:,(r-1)*dim+1:r*dim)=Rp;
    % compute the cp to determine the regions
    cp=umax^2/(K(r,:)*inv(Rp)*K(r,:)');
    c_p(r)=cp;
    % define a varialbe to test the nesting condition
    Rp_cp(:,(r-1)*dim+1:r*dim)=Rp/cp;
    % define auxiliary variables to plot the regions
    a(r)=Rp(1,1);
    b(r)=Rp(2,2);
    c(r)=Rp(1,2)+Rp(2,1);
    d(r)=-cp;
end

% verifying the nesting condition
for ind=1:nr-1
    nest_test=Rp_cp(:,(ind)*dim+1:(ind+1)*dim)-
    Rp_cp(:,(ind-1)*dim+1:ind*dim);
    if eig(nest_test)>=0
        test(ind)=1;
    else
        test(ind)=-1;
    end
end

if test>=0
    disp('Lyapunov␣regions␣are␣one␣inside␣the␣other')
else
    disp('Lyapunov␣regions␣are␣not␣nested!')
end
```

A.2.3 Computation of the Convex Polyhedral Regions

```
% loading the data file
dati;

% define a structure containing the systems information
sysStruct.A=G;
sysStruct.B=H;
sysStruct.C=Cd;
sysStruct.D=Dd(:,1);
sysStruct.xmax=[1;1;1;1];
sysStruct.xmin=[-1;-1;-1;-1];
sysStruct.ymax=[1;1];
sysStruct.ymin=[-1;-1];
sysStruct.umin=-umax;
sysStruct.umax=umax;

% introduce a polytope limiting the
% feasible state-space of interest
sysStruct.Pbnd = unitbox(4,1);

% define the prediction horizon
probStruct.N=10;

% consider a quadratic performance index
probStruct.norm= 2;

% define weights on the states
probStruct.Q=Q;
% define weights on the inputs
probStruct.R=R;

% choose level of optimality
probStruct.subopt_lev =0;
% terminal set constraint
probStruct.Tconstraint=1;

% compute the partition and the assigned control matrices
[ctrlStruct] = mpt_control(sysStruct, probStruct);
```

A.3 Simulation of the Semiactive Suspension System

```
% load the data file
dati;

% define parameters to model the disturbances
alfa=.2;
vel=20;
sigmaquadro=.1^2;
w=rumbian(alfa,sigmaquadro,vel);
rumore;
xo=xo*0;

% define the initital vector
Xcgo(:,1)=[.015; .1; 0; 0];

% simulate the semiactive suspension closed loop system
t=1; T(1)=0; f(1)=fmax; while (t<200)
   v=0;
   r=1;
   cont=1;
   while (r<=nr & cont==1)
            S=abs(ZG(:,Jzi(r):Jzf(r))'*Xcgo(:,t));
            max(S);
            if max(S)<=umax
               v=r;
            else
                cont=0;
            end
            r=r+1;
   end % search the smallest region, which contains x(t)
   if v==0
      disp('x(t) does not belong to the largest region');
   end
   Va(t)=v;
   % determine the value of f that has to be adjusted
   if (t>=2)
       if (Xcgo(4,t)~=Xcgo(3,t))
           f_star(t)=-1*((K(v,:)*Xcgo(:,t)+
           lambdas*(Xcgo(2,t)-Xcgo(1,t)))/
           (Xcgo(4,t)-Xcgo(3,t)));
           if f_star(t)>=fmax
               f(t)=fmax;
           elseif f_star(t)<=fmin
               f(t)=fmin;
           else
               f(t)=f_star(t);
           end
       else
```

```
48              f(t)=fmax;
49          end
50      end
51      % compute the evolution of the system
52      Kcgo(:,t)=-[lambdas -lambdas f(t) -f(t)]';
53      Ua(t)=-Kcgo(:,t)'*Xcgo(:,t);
54      Xcgo(:,t+1)=(G-H*Kcgo(:,t)')*Xcgo(:,t)+Ltilde*xo(t);
55      t=t+1;
56  end;
```

Bibliography

[1] Adamy, J.: Strukturvariable Regelungen mittels impliziter Ljapunov-Funktionen. Ph.D dissertation (1991), University of Dortmund

[2] Adamy, J. and Flemming, A.: Soft variable-structure controls: a survey. Automatica 40 (2004), pp 1821–1844

[3] Bemporad, A., Fukuda, K. and Torrisi, F.D.: Convexity recognition of the union polyhedra. Computational Geometry 18(2001), pp 141-154

[4] Bemporad, A., Morari, M., Dua, V. and Pistikopoulos, E.N.: The explicit linear quadratic regulator for constrained systems. Automatica 38 (2002), pp 3-20

[5] Borrelli, F.: Discrete time constrained optimal Control. Ph.D dissertation (2002), ETH Zurich

[6] Cha, P.D., Rosenberg, J.J. and Dym, C.L.: Fundamentals of Modeling and Analyzing Engineering Systems. University Press, Cambridge (2000)

[7] Corona, D., Giua, A. and Seatzu, C.: Optimal control of hybrid automata: design of a semiactive suspension. Control Engineering Practice 12 (2004), pp 1305–1318

[8] Corriga, G., Sanna, S. and Usai, G.: An optimal tandem active-passive suspension for road vehicles with minimum power consumption. IEEE Transactions on Industrial Electronics, 38(3) (1991)

[9] Fukuda, K.: FAQ in polyhedral computation. On line document (2004) available from `http://www.ifor.math.ethz.ch/staff/fukuda`

[10] Giorgetti, N., Bemporad, A., Tseng, H. E. and Hrovat, D.: Hybrid model predictive control application towards optimal semi-active suspension. Proc. IEEE Int. Symp. on Industrial Electronics, Dubrovnik, Croatia (2005)

[11] Giua, A., Seatzu, C. and Usai, G.: Semiactive Suspension Design with an Optimal Gain Switching Target. Vehicle Systems Dynamics, 31(1999), pp 213–232

[12] Glad, T. and Ljung, L.: Control Theory: multivariable and nonlinear methods. Taylor & Francis (2000)

[13] Hung, J. Y., Gao, W. and Hung, J.C.: Variable structure control: A survey. IEEE Transactions on Industrial Electronics, 40(1) (1993), 2-22

[14] Kiendl, H., and Schneider, G.: Synthese nichtlinearer Regler fuer die Regekstrecke const/s^2 aufgrund ineinandergeschalchtelter abgeschlossener Gebiete beschraenkter Stellgroesse. Regelungstechnik und Prozess-Datenverarbeitung, 20(7) (1972)

[15] Kiendl, H., Adamy, J. and Stelzner, P.: Vector norms as Lyapunov functions for linear systems. IEEE Transactions on Automatic Control, 37(6) (1992), pp 839–842

[16] Kvasnica, M., Grieder, P. and Baotic, M.: Multi-Parametric Toolbox (MPT), `http://control.ee.ethz.ch/ mpt/`, (2004)

[17] Kvasnica, M., Grieder, P., Baotic, M. and Christophersen, F.J.: Multi-Parametric Toolbox Manual (2004)

[18] Ljung, L. and Glad, T.: Modeling of Dynamic Systems. Prentice Hall (1994)

[19] Loskot, K., Polánski, A. and Rudnicki, R.: Further comments on "Vector norms as Lyapunov functions for linear systems". IEEE Transactions on Automatic Control, 43(2) (1998)

[20] Ogata, K.: Discrete-Time Control Systems, Second Edition. Prentice Hall (1995)

[21] Savastano, A.: Synthesis of a optimal gain switching controller for linear discrete time systems with constrained input: application to an active suspension system for cars. Laurea Thesis, Department of Electrical and Electronical Engineering, University of Cagliari (Italy) (1997) (in Italian)

[22] Yoshida, K., Nishimura, Y. and Yonezawa, Y.: Variable gain feedback control for linear sampled-data systems with bounded control. Control Theory and Advanced Technology, 2(2) (1986)

[23] Ziegler, G. M.: Lectures on Polytopes. Springer (1994)