# Analysis and Design of Hybrid Control Systems

Jörgen Malmborg

Department of Automatic Control, Lund Institute of Technology

# Analysis and Design of Hybrid Control Systems

# Analysis and Design of Hybrid Control Systems

Jörgen Malmborg

Lund 1998

*To Annika*

Published by
Department of Automatic Control
Lund Institute of Technology
Box 118
S-221 00 LUND
Sweden

ISSN 0280–5316
ISRN LUTFD2/TFRT--1050--SE

Printed in Sweden by Printing Malmö AB
Malmö 1998

# Contents

## Acknowledgments

It would not have been possible to produce this thesis without a lot of help from many people and it is a great pleasure to be given this opportunity to express my gratitude.

First of all I would like to thank the three persons that have given me invaluable help with my research. Bo Bernharsson, who has been a great support both during the research and during the writing of the thesis. I truly admire his great analytical skills and his excellency in finding errors, small as large, in my manuscripts. Karl Johan Åström, who with his deep knowledge of automatic control and his great enthusiasm has been a source of inspiration throughout my years at the department. Johan Eker, a dear friend and research colleague, who retains his sense of humor even after spending endless hours over the computer code for our experiments.

It is truly a great privilege to work in such an intelligent and stimulating environment as the Automatic Control Department at Lund Institute of Technology. I cannot mention all of you but thanks to Henrik Olsson, Johan Nilsson and the rest of you for good times at, and after work.

The work has been partly supported by the Swedish National Board for Industrial and Technical Development (NUTEK), Project Heterogeneous Control of HVAC Systems, Contract 4806. Diana Control AB provided us with controller hardware and the opportunity to test the hybrid controller on a real process.

Thanks also to friends and family who have rooted for me even without actually knowing what I was doing. The last and most important thanks go to my beloved Annika for her love and support during times of hard work.

J.M.

# 1

# Introduction

In control practice it is quite common to use several different controllers and to switch between them with some type of logical device. One example is systems with selectors, see Åström and Hägglund (1995), which have been used for constraint control for a long time. Systems with gain scheduling, see Åström and Wittenmark (1995), is another example. Both selectors and gain scheduling are commonly used for control of chemical processes, power stations, and in flight control. Other examples of systems with mode switching are found in robotics. Some examples are the systems described in Brockett (1990), Brooks (1990) and Brockett (1993). In this case many different controllers are used and the coordination of the different controllers are dealt with by constructing a special language. The expert controller discussed in Åström and Årzén (1993) represents another type of system with an hierarchical structure where a collection of controllers are juggled by an expert system. The autonomous controllers systems described in Antsaklis *et al.* (1991) and Åström (1992) have a similar structure.

Lately there has been a considerable effort in trying to unify some of the approaches and get a more general theory for hybrid systems. The fundamental problem with hybrid systems is their complex mixture of discrete and continuous variables. Such systems are very general and they have appeared in many different domains. They have, for example, attracted much interest in control as well as in computer science. In automatic control the focus has been on the continuous behavior, while computer science has emphasized the discrete aspects, see Alur *et al.* (1993), Alur and Dill (1994) and Henzinger *et al.* (1995a).

It is generally difficult to get analytical solutions to mixed difference/differential equations. For some problems it is possible to do qualitative analysis for aggregated models. Because of the lack of good analysis methods, many investigations of hybrid systems have relied heavily on simulation. Unfortunately the general purpose simulation tools available today are not so well suited for hybrid systems.

This chapter gives a brief overview of some aspects of hybrid control systems: modeling, analysis, design and verification. The main part of the rests of thesis, chapters two to five, treat specific problems in hybrid control systems. The topics are fast mode changes, simulation, a design method that guarantees stability and finally some experiments with hybrid control systems.

## 1.1 What is a hybrid system?

Hybrid systems research can be characterized in many ways. In broad terms, approaches differ with respect to the emphasis on continuous or discrete dynamics, and on whether they emphasize simulation, verification, analysis or synthesis.

### A young multi-disciplinary field

Hybrid systems appear both in automatic control and in computer science. This creates some difficulties because researchers from different areas use different terminology. This is also an advantage because it generates a wide spectrum of problems. Depending on the researcher's background and the actual problem they can use timed automata, dynamical systems theory, automata theory, discrete event systems, programming verification methods, logic programming etc etc. Whatever basic method used, the questions asked are often the same. What mix of continuous and discrete properties is rich enough to capture the properties of the systems that is modeled? How can it be verified that the hybrid model satisfies the demands on performance and stability? How can a controller be derived that meets discrete and continuous specifications?

A typical trend is that the computer scientists focus on the logic and the discrete aspects and the continuous aspects are treated quite cavalierly while the control engineers do the opposite. A good approach should perhaps take a more balanced view.

### Hybrid Control Systems

A hybrid control system is a control system where the plant or the controller contains discrete modes that together with continuous equations govern the behavior of the system. This general definition covers basically every existing control system.

When the discrete parts are within the controller it is often in the form of a scheduler or a supervisor. A limiter or a selector can also be viewed as a discrete part of the controller. The process itself can also have discrete modes. Many mechanical systems have physical limitations, e.g. stops and barriers. Today's more complex control systems normally contain discrete parts both in the controller and in the plant. More often than not, the supervisory functions on higher hierarchical layers contain discrete modes.

Since systems with mode switches appear in so many contexts there is no unified approach. Different ways to deal with them are scattered in many application areas. They also appear under many different names: heterogeneous systems, multi-modal systems, multi-controller systems, logic-based switching control systems, discrete event systems are some of the names that appear in the literature. The name hybrid systems has been used as a label for a large variety of engineering problems. In this thesis it is used for control systems consisting of both continuous and discrete parts.

EXAMPLE 1.1—TEMPERATURE CONTROL
The first example of a hybrid control system is temperature control with two actuators, one for heating and one for cooling. The system thus has two discrete modes. It is assumed that the continuous behavior of the system can be described by one state, the temperature $T$. The behavior in the heating and the cooling mode is different and a hybrid model of the system is

$$\frac{dT}{dt} = \begin{cases} f_c(T,t,u) \\ f_h(T,t,u) \end{cases}, \tag{1.1}$$

where $f_c(t,T,u)$ are the dynamics using the cooler and $f_h(t,T,u)$ are the dynamics using the heater. Imagine that only one of the controllers, heater or cooler, can be active at the time. To keep a certain temperature it is therefore necessary to switch between them. The relative time a controller is used decides the temperature of the object. The control problem is to determine a switch strategy. □

The hybrid model described by Equation 1.1 is in many ways too simplistic to capture all the features of a general hybrid system. Sometimes there is a need for a more advanced description with additional properties like state jumps, state space structure changes, infinitely fast mode changes, more than two controllers, etc.

## Hybrid control systems is really nothing new

Even if hybrid systems have been used for a long time the name hybrid systems is relatively new. Many older control paradigms fit nicely into the hybrid systems framework. Examples of these are: Selector-based Control, Gain Scheduling, Fuzzy Control and Logic Control.

EXAMPLE 1.2—FLIGHT CONTROL
Control of airplanes has already from the beginning been done with a hybrid controller with several modes. The airplane has several operating modes depending on: speed, load, air pressure, take-off and landing. □

## Present status of HCS

In spite of their common practical use there is very little general purpose theory available for systems with mode switching. For this reason both analysis and design are often done intuitively. Variable structure systems, Emelyanov (1967), Itkis (1976) and Utkin (1977) is one approach that is well established. Discrete event dynamical systems, Ramadge and Wonham (1989) and Cassandras (1993) is another approach, with a theoretical foundation.

Different efforts have different focus. There are today methods that solve very specific problems of low complexity. These methods are already in the implementation phase now. There are also very abstract methods that deal with large complex systems on a high theoretical level. It seems to be a longer way to actual implementation for these methods.

Today, most investigations of hybrid systems are done by simulation. Even if much research effort is put into the the field of hybrid systems this situation is not likely to change in the near future. It is therefore crucial to have good simulation tools for hybrid systems. There are several simulation packages that allow for the mixture of continuous variables and discrete variables, but the simulation performance is often poor. One important problem is simulation of models where so called sliding mode behavior arises, i.e. where there are infinitely many mode switches in a finite time interval. The simulation results for such systems are not to be trusted. There will be more about this later on.

The control systems currently in use are more or less of hybrid nature. Many controllers are linear controller with 'fixes'. One example is turning on and off integration in a PI-controller. Maybe the most common hybrid controller is the one that the operator is implementing together with the normal controller for the process. For start-up the controller is switched into *manual*. When close enough to the operating point the controller is switched to *automatic*. One of the design methods in this thesis is solving, i.e. automating, this problem as a special case.

## What's next

Over the last few years there has been a considerable research effort in the area of hybrid systems. Numerous control systems with different models are presented at the control conferences over the world. The merging of the practical implementation of hybrid controllers with theoretical results are beginning to show up. This will lead to much better performing control systems together with a more thorough understanding of why some of the good old ones work.

It is the hope that the study of hybrid systems will evolve into a tool capable of analyzing complex systems with discrete and continuous dynamics. Lately there has been a lot of work in the area of computer aided verification of hybrid systems, see Henzinger *et al.* (1995b). The computer aided verification and analysis is getting better. It will be absolutely necessary to be able to do automatic analysis of systems as they grow more complex. A probable scenario is that simulators and automatic analyzers will grow into one tool, maybe also with some theorem proving capabilities.

## 1.2 Why use hybrid control systems

Many physical systems are hybrid in the sense that they have barriers or limitations. Inside the limitations they are modeled with differential equations. A natural way to model these systems is to use a mixture of differential equations and inequalities. Other systems have switches and relays that can be naturally modeled as hybrid systems. These hybrid models show up in many areas. Typical examples are flight control, air traffic control, missile guidance, process control, robotics etc. Even if the modes are strictly speaking not discrete it can be advantageous to model systems that way. An example of this is when a nonlinear system is modeled with a set of linear models each one covering a part of the state space.

EXAMPLE 1.3—HELICOPTERS
Helicopters are dynamical systems with several distinct modes. To model the behavior, at least three sub-models are needed. One for hovering, one for slow motions and one for fast motions where the helicopter dynamics approach the dynamics of an airplane. □

EXAMPLE 1.4—AUTOMOTIVE SYSTEMS
A good example of a complex hybrid system is a car. Discrete signals are gear, load and road characteristics, driver inputs, ABS control signal and warnings. The continuous parts are often nonlinear e.g. motor characteristics, sensor signals. To make a good model suitable for simulation and controller design it is essential to use hybrid system modeling. □

The major reason for using hybrid control systems is that they can outperform single controller systems and that they can solve problems that can not be dealt with by conventional control. Another reason is that it is easier to profit from the richer structure of a hybrid plant model if using a hybrid control system adapted to this model.

Hybrid controllers are a special class of nonlinear controllers. They are not restricted by some of the limitations always present for linear systems. Sometimes hybrid control methods can be used to achieve better performance than the limitations in the next section say is possible for linear systems.

## Fundamental limitations for linear systems

For some problems it can be necessary to use a nonlinear controller to guarantee stability. A popular example is the *nonholonomic integrator*.

EXAMPLE 1.5—NONHOLONOMIC INTEGRATOR
The system described by

$$\begin{aligned}\dot{x} &= u\\ \dot{y} &= v\\ \dot{z} &= xv - yu, \end{aligned}\tag{1.2}$$

where $u$ and $v$ are control signals, cannot be locally asymptotically stabilized with any time-invariant, smooth controller, see Brockett (1983). On the other hand there are several hybrid design methods that can be used to derive a stabilizing controller. □

Research going back to Bode (1945) shows that there are fundamental limitations to what can be achieved by linear feedback. Feuer *et al.* (1997) shows some of these limitations and how hybrid control systems can be used to improve performance. The famous result that the sensitivity function $S(s)$ satisfies

$$\frac{1}{\pi}\int_0^\infty \log|S(j\omega)|d\omega \geq \sum_{i=1}^{n_c} p_i \tag{1.3}$$

where $p_i$ are the open loop right half plane poles is one such limitation. In Middleton (1991) a similar time domain result is shown. If $q$ is a right half plane zero of the plant and $y(t)$ is the closed loop system step response, then using error feedback

$$\int_0^\infty e^{-qt}y(t)dt = 0. \tag{1.4}$$

From this follows the result that the step response for a non-minimum phase system starts in the wrong direction. Another limitation of this kind is if the open linear system contains two integrators then the error $e(t)$ of any stable closed loop system after a step input satisfies

$$\int_0^\infty e(t)dt = 0. \tag{1.5}$$

if error feedback is used. The overshoot is thus unavoidable. A hybrid control solution of the kind introduced in the next section and further analyzed in Chapter 4 can eliminate this problem. The solution there is to use two controllers, for instance one time-optimal controller and one fixed linear controller.

## Starting point for this thesis

The next example illustrates the design problem that was the starting point for this thesis. The question to answer was this one: what can be said about mixing control signals from several controller and when is it worth while doing?

EXAMPLE 1.6—FAST STEP RESPONSE
In the design of a PID control system there is often a choice between a fast controller giving a large overshoot or a slow controller without a overshoot. See Figure 1.1 (left). (For some processes the overshoot can be reduced by set-point weighting.) For the process in Figure 1.1, a first order system with a time delay, it is quite easy to manually control the system during the set-point change without getting any overshoot. The manually applied control signal is imitating the control signal from a time-optimal controller. See Figure 1.1(right). A good hybrid controller for this problem

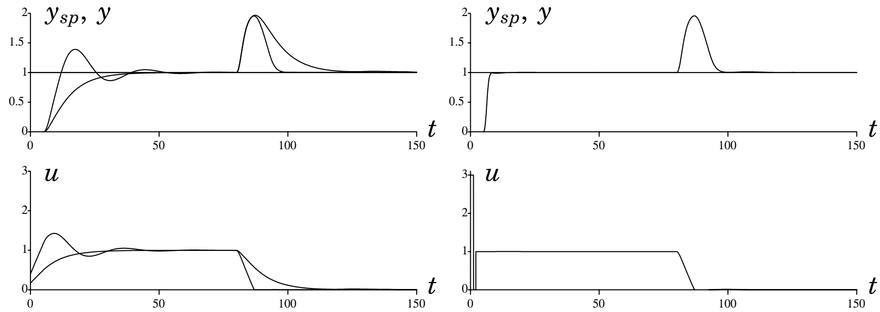

**Figure 1.1** Two examples of PID control (left). Combined manual and PID control (right). The set-point $y_{sp}$ and the measured variable $y$ (top). Control signal $u$ (bottom).

would consist of a PID controller, a time-optimal controller, and a selector.

The results of this example are promising. Very fast step responses could be combined with good steady state regulation. □

The example above illustrates the advantages of combining a strategy for fast set-point responses with a good regulation property. Expanding on this idea it is clear that there are many control problems that could profit from a hybrid design method. In fact, as soon as there are multiple design goals that cannot be met by a single controller it can be advantageous to use several controllers.

There are today many good controller design methods. Only a few of these methods allow the designer to take several different design objectives into account. A sample of possible design goals are

- stability of the closed loop system,
- fast load disturbance rejection,
- fast set-point response,
- low sensitivity to measurement noise,
- low sensitivity to process variations.

Chapter 4 introduces a method that combines control signals from different controllers while guaranteeing stability. The hybrid controller design method is based on Lyapunov stability theory.

### Why avoid hybrid control systems?

Some advantages of hybrid systems have been mentioned. What about their disadvantages? The major drawbacks with hybrid control systems are

- higher complexity,
- incomplete system understanding,
- lack of analysis methods,
- lack of appropriate design methods.

While it is fairly easy to come up with a hybrid controller that seems to perform well for a given system it is often very difficult to prove that it works under all conditions.

## 1.3 A block model of a general hybrid control system

This section presents a conceptual model of a hybrid system. Some proposed mathematical models of hybrid systems are reviewed in a following section. The conceptual model is a graphical description of the different parts of a hybrid control system. Figure 1.2 shows a general hybrid control system with seven main blocks or parts. Many existing hybrid control

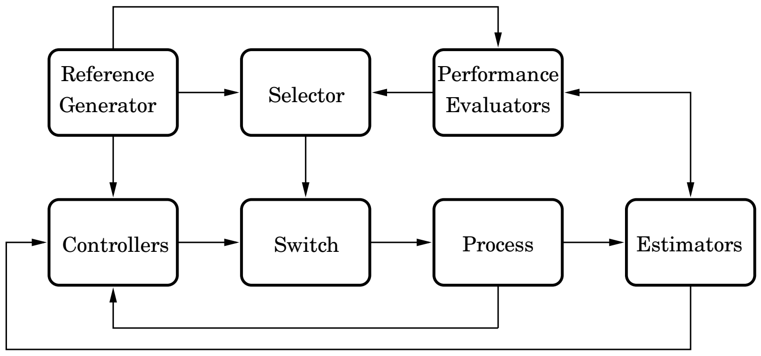

**Figure 1.2** A block model of a general hybrid control system.

systems fit into this picture. The complexity of the blocks can vary from rudimentary to very complex. In the actual implementation several blocks can of course be coded together. The choice can for example depend on the requirements on message passing between the blocks. The signals and messages between the blocks are a mixture of discrete and continuous signals.

The major tasks and features of the building blocks are presented in the following sections.

## Reference generator block

The basic task for this block is to generate a reference signal for the controllers. In many cases other parts of the hybrid control system will benefit from a mode information signal. The mode signal could carry information of the current task of the control system. Examples of such mode signal information are

$$\{\textit{keep level}, \textit{change operating point}, \textit{perform automatic tuning}\}. \qquad (1.6)$$

Information about the control objective could be used both by the selector block and the performance evaluation block. The control signal is thus a vector comprised of the continuous reference values and discrete mode information signals, $[y_{ref}, q]$.

## Controller block

In a hybrid control system there is a set of controllers to choose from. The set could contain anything from two to an infinite number of controllers.

It is not necessary that the controllers belong to the same class or that they share the same state space information. However, to be able to handle wind-up and bump-less transfer problems the control signal value to the active controller has to be available for all the other controllers.

In a gain scheduled system the controllers have the same structure but different parameters. Here it is preferable to let the controllers share state information.

Other designs methods do not use controllers of the same type. For the fast set-point response problem it was for instance useful to combine a time-optimal controller with a PID controller.

### Process block

The process can be time-varying. The variation could be due to load disturbance, different operating points etc. Exceeding certain levels, e.g. fluid overflows, can also cause the process dynamics to change drastically. This type of discrete mode changes are useful information to the process evaluation and selector block.

### Estimator block

The estimator block has basically two tasks. The first one is to continuously update the parameters of a changing process. Secondly, if not all states are measurable then the non-measurable states need to be estimated with some kind of filters. Discrete mode changes could cause resets in the estimates or a change in the estimated model structure. An additional task could be to estimate mode information from the output signals.

### Performance evaluation block

The performance evaluation block will for many hybrid systems be quite elaborate. The task is to assess the quality of the control and the estimations. The evaluation signal sent to the selector block is discrete and qualitative, examples are

$$\begin{aligned}&\{\textit{sensor one is out of order}, \textit{critical value for output } y_2 \textit{ is reached},\\ &\quad \textit{high noise level}, \textit{parameter estimates are good}, \textit{mode} = 2\}.\end{aligned} \tag{1.7}$$

***Robust controller*** It is possible to use a tighter, more aggressive, control if the correspondence between the model and system is better. One possibility is to let the performance evaluation estimate the model accuracy. Based on this, a controller with matching robustness properties is selected.

One criterium for controller selection could be the noise level in the system. If the noise level is high some controllers could be unusable. If on the other hand the noise level is low those controllers could perform better.

### Selector block

Based on the information from the reference generator and performance evaluation block this block selects which controller to use.

EXAMPLE 1.7
If the information from the performance evaluation block is that the knowledge of the model is excellent and the reference generator block demands a set-point change a time-optimal controller could be selected.
□

The selector is in practical implementations often a static logic function. A more advanced version is with memory in the selector device. A simple version of this is a hysteresis function. More complex structures could be implemented with finite state machines.

If not all the controllers in the controller set are implemented, for example when using gain-scheduling, then a recalculation of controller parameters could be done by this block.

### Switch block

This block takes care of the switching from the controller in use, $u_1$, to the new controller, $u_2$, chosen by the selector. There are several methods in use for switching

- Smooth transition. The control signal is calculated as a weighted mean $u^* = (1-\lambda)u_1 + \lambda u_2$. Where $\lambda$ gradually goes from 0 to 1.
- Abrupt transition from $u_1$ to $u_2$ and then low-pass filtering of the control signal sent to the process.

If bump-less transfer can be accomplished it is an advantage for many applications. It will be shown later that the smooth transition, even for stabilizing controllers $u_1$ and $u_2$, may lead to unstable systems.

The switching mechanism has a large influence of the properties of the closed loop. Despite this, it is sometimes neglected in the analysis of hybrid control systems.

## 1.4 Hybrid control system design

In this section some hybrid control schemes are reviewed. All schemes fit into the general picture in Figure 1.2, but not all blocks are explicitly present in every design method.

## Multi-control

Morse (1995b) uses a somewhat simpler architecture, see Figure 1.3. Based on that architecture he introduces the notation *multi-controller* for several different controller designs. The basic idea is that the process outputs $y$ are fed back to a set of controllers. Then each controller generates a candidate control signal. Which control signal that is finally used is decided by the supervisor. At a certain point in time only one control signal is used, $u = u_\eta$, where $\eta$ takes values in the index set of the controller set.

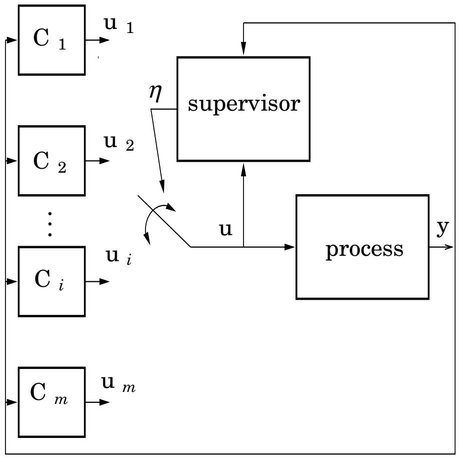

**Figure 1.3** Multi-controller architecture, see Morse (1995b).

## State sharing – gain scheduling

The actual implementation of the controllers is not necessarily done as the structure in Figure 1.3 indicates. If the controllers have the same internal structure then the full controller set can be implemented with only one control algorithm and some parameters. An example of this is a gain scheduled controller where all the controllers, in the abstract multi-controller, share the state information. Figure 1.3 could then be further reduced to only one controller and a parameter passing method from the supervisor.

It is not necessary to limit the number of controllers. In this set-up it is possible to use an infinite range of controllers.

## Manual control

Simple controllers of the PID type have undergone an interesting development over the past 15 years with the introduction of automatic tuning

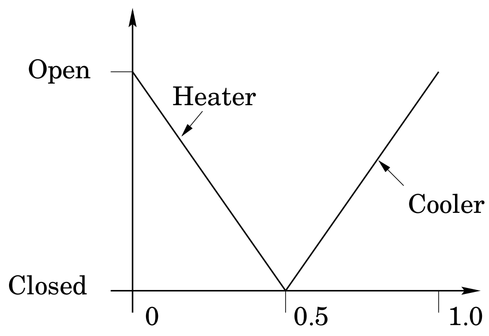

**Figure 1.4** Split range control. The measured variable determines which controller to use.

and adaptation. This work has also led to a significantly improved understanding of PID controllers and their tuning. A common engineering practice is to tune the controllers for good load disturbance rejection and small sensitivity to process variations. The set-point response is then handled by set-point weighting. Such controllers will perform quite well for moderate set-point changes. Their response to large set-point changes can, however, be improved.

Knowing that the controller is not optimized for large set-point changes a process operator that has good knowledge of the process can put the controller in manual mode and give the control signal himself. The control signal thus implemented by the operator is often an approximation of a time-optimal control signal. A typical control sequence is

$$\langle \textit{full speed forward}, \textit{maximum break} \rangle. \tag{1.8}$$

## Split range control

Split range control is sometimes used for systems with several controllers and only one measured signal. The measured variable determines which controller to use, see Figure 1.4. The method is commonly used in systems for heating and ventilation and it could have been applied to the temperature control in Example 1.1. If the measured temperature is high, only the cooler is active, and if the measured temperature is low, the heater is on.

The heterogeneous controller in Kuipers and Åström (1991) has a similar idea but with overlapping working regions.

## Max and Min selectors

Selector control is the counter-part of split range control. In selector control there are many measured signals and only one actuator. The selector is a static device with many inputs and one output. Common selector types are: maximum and minimum.

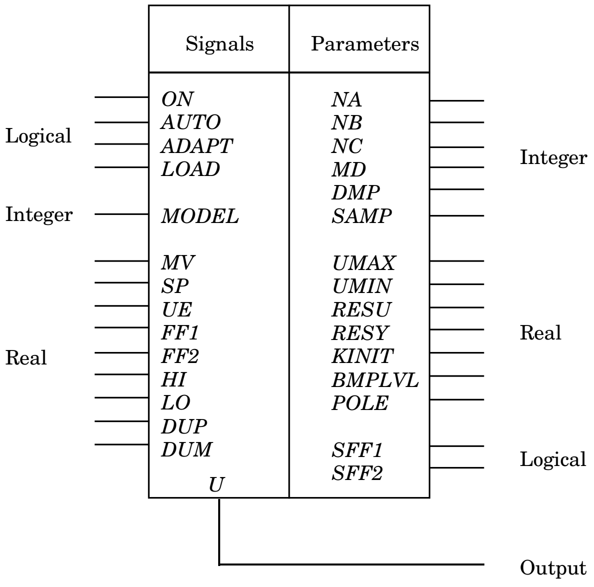

**Figure 1.5** The expert module STREGX in Firstloop. Picture taken from Åström and Wittenmark (1995) p. 513.

### Adaptive control

Adaptive controllers are hybrid controllers with several modes. One example is the adaptive controller from the company First Control. The expert module STREGX in Firstloop is shown in Figure 1.5. The mode switches, *ON, AUTO* and *ADAPT* control if the adaptive controller is in modes on/off, automatic/manual and adapting or not.

### Variable structure systems

Another configuration where several controllers are used is in variable structure systems. The basic idea in variable structure systems is to define a switching function $S = S(x)$. The applied control signal then depends on the sign of the switching function. In the multi-variable situation there can be several switching functions $s_i(x)$ and one control signal for each

$$\dot{x} = f(x,u,t)$$
$$u_i = \begin{cases} u_i^+(x,t), & if s_i(x) > 0 \\ u_i^-(x,t), & if s_i(x) < 0. \end{cases} \tag{1.9}$$

The applied control is such that the trajectories will approach the subspace $S = 0$. This example illustrates the importance of being able to detect fast switches and being able to analyze the dynamics for a sliding mode behavior. This problem is further addressed in chapters 2 and 3.

## Knowledge-based control

The structure of a knowledge-based controller, Figure 1.6, see Åström and Årzén (1993), fits into Figure 1.2 The expert controller represents a type

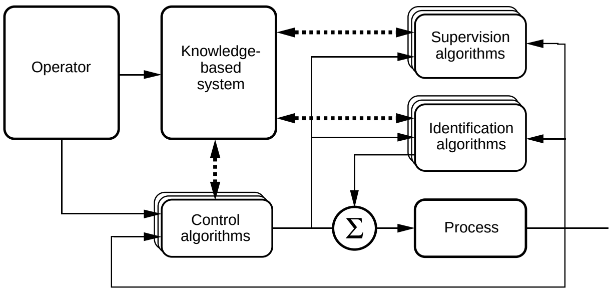

**Figure 1.6** The expert system decides witch controller to use.

of system with an hierarchical structure, where a collection of controllers is juggled by an expert system.

## Hierarchical control

Reducing complexity is an important reason for dealing with hybrid systems. Using hierarchical models is one method of modeling dynamic processes with different levels of abstraction. In fact, all hybrid controllers fitting the general conceptual model are more or less hierarchical.

One example of successful hierarchical control is the hybrid control of air traffic, see Antsaklis (1997) pp 378-404. The control structure for an airplane is layered as: Strategic Planner, Tactical Planner, Trajectory planner and Regulation

- Strategic Planner: design a coarse trajectory for the aircraft.
- Tactical Planner: refines the strategic plane and predicts conflicts with other airplanes.
- Trajectory Planner: designs full state and input trajectories for the aircraft together with a sequence of flight modes necessary.
- Regulation Layer: tracks the feasible dynamic trajectory

Each layer uses a finer and more detailed model of the airplane.

### Reconfigurable controllers

In airplane control there are often more control surfaces than actually needed to stabilize the plane. It is in some cases possible to fly with one engine out of order, with a partially broken wing etc. It is possible to build a hybrid controller based on several configurations of the airplane. Imagine that controller $c_1$ is used when all systems are ok, controller $c_2$ is used when left engine is out, controller $c_3$ is used when right rear wing is blown away, etc. Here the performance evaluation block should detect the failure of parts of the system and schedule the use of a new controller.

### Real-time demands

In real-time systems with several control loops it is possible to let the loop with highest demands on sample time get the most cpu time. Doing this leaves other control loops with less time. It might be necessary to use less cpu time demanding controllers for those loops. In this case the selector block needs information of available cpu time to be able to chose controller. The area is quite new but it is imaginable to distribute cpu time as a function of the tasks that the controllers are performing at a certain instant. Then given a certain amount of cpu time, suitable controllers are selected.

### Fuzzy control

The idea of mixing or switching between control signals for hybrid systems that are represented as local models with local controllers are very much in line with what is done in fuzzy control. Some hybrid control schemes could be viewed as a fuzzy controller, Sugeno and Takagi (1983).

## 1.5 Mathematical models of hybrid systems

There is a number of mathematical models describing hybrid systems. A common feature is that the state space $\mathcal{S}$ has both discrete and continuous variables, e.g. $\mathcal{S} \subset R^n \times Z^m$. The equations can be linear or nonlinear and in general the discrete parts cannot be separated from the continuous parts. The models proposed by various researchers differ in definition of and restrictions on dynamic behavior. The difference between the models are on aspects as generality, allowance of state jumps, dynamic restrictions etc. Many models do not allow fast switching or sliding.

### The modeling problem

The modeling problem consists of creating models with a sufficient complexity to capture the rich behavior of hybrid control systems. Yet they should be easy enough to analyze, and formulated in a way that allows simulation.

In the following sections there are short presentations of some of the proposed models. Different branches of control science have their favorite traditional model structures. There is no unified approach and not yet any agreement on what constitutes the most fruitful compromise between model generality and expressibility. For a review of different approaches see Branicky (1995) or Morse (1995a).

### Tavernini

Most of the work in this thesis will be be on systems that are on the Differential Automata form described in in Tavernini (1987):

$$\begin{aligned} \dot{x} &= f(x(t), q(t)), & x &\in R^n \\ q(t) &= \nu(x(t), q(t^-)), & q &\in Z^{m+} \end{aligned} \tag{1.10}$$

where $x$ denotes the continuous and $q$ the discrete variables. This model does not allow for autonomous or controlled state jumps. Hybrid models of this type are often represented with a graph, see Figure 1.7. Here each

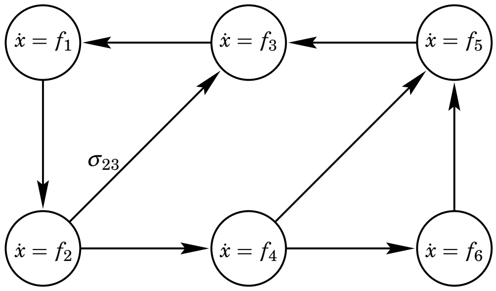

**Figure 1.7** Graph of a Tavernini-type hybrid system.

of the nodes represents a mode of the system. Associated with each mode is a dynamic equation $\dot{x} = f_q$ and mode jump conditions $\sigma_{qr}$.

In their article in Antsaklis (1997) pp 31-56, Branicky and Mattsson discuss modeling and simulation of hybrid systems. The mathematical models below of Branicky, Brockett and Artstein are taken from that article.

### Branicky

Branicky (1995) makes a formal definition of a controlled hybrid dynamical system (CHDS), $H_c = [Q, \Sigma, A, G, V, C, F]$, where $V$ is the discrete controls, $C$ the collection of controlled jump sets and $F$ the collection of controlled jump destination maps. Under some conditions this is a hybrid dynamical system (HDS) $H = [Q, \Sigma, A, G]$ where $Q$ is the discrete states, $\Sigma$ the continuous dynamics, $A$ the autonomous jump sets and $G$ the autonomous jump transition map. In an equation form comparable with other models in this section this is

$$\begin{aligned} \dot{x}(t) &= f(x(t), q(t)), \quad x(t) \notin A_{q(t)} \\ \left.\begin{aligned} q(t^+) &= G_q(x(t), q(t)) \\ x(t^+) &= G_x(x(t), q(t)) \end{aligned}\right\}&, \quad x(t) \in A_{q(t)} \end{aligned} \tag{1.11}$$

### Brockett's model

Brockett (1993) uses the description

$$\begin{aligned} \dot{x}(t) &= f(x, p, z), \\ \dot{q} &= r(x, p, z), \\ z\lceil p \rceil &= \nu(x, p, z\lfloor p \rfloor), \end{aligned} \tag{1.12}$$

where $x \in X \subset \mathbb{R}^n, p(t) \in \mathbb{R}$, and the rate equation r is nonnegative for all arguments. Brockett has a mix of continuous and discrete variables. The variable $z$ is changed whenever $p$ passes through integer values. The notation $\lfloor p \rfloor$ denotes the largest integer less than or equal to $p$. The notation $\lceil p \rceil$ denotes the smallest integer greater than $p$.

### Artstein's model

Artstein uses a number $N$ of sub-models. Each sub-model has its own dynamics, $\dot{x} = f_q(x)$. A specific model is run for a certain time $T_q$. After time $T_q$ there is possibly a switch to another sub-model depending on the value of a test function $\psi_q(x)$. In equations this is

$$\begin{aligned} [\dot{x}(t), \dot{\tau}(t)]^T &= [f_{q(t)}(x(t)), 1]^T, & 0 \le \tau \le T_{q(t)}, \\ q(t^+) &= \left\{ \begin{aligned} G_p(q(t)), \psi_q(t)(x(t)) \ge 0, \\ G_n(q(t)), \psi(t)(x(t)) < 0. \end{aligned} \right\}, & \tau = T_{q(t)}, \\ \tau(t^+) &= 0, & \tau = T_{q(t)}. \end{aligned} \tag{1.13}$$

### Stiver–Antsaklis

Stiver and Antsaklis (1992) uses a hybrid description with the plant modeled with continuous equations

$$\begin{aligned} \dot{x} &= f(x(t), r(t)) \\ z(t) &= g(x, t) \end{aligned}$$

and a discrete regulator

$$\begin{aligned} t_i &= \delta(t_{i-1}, zh_i) \\ rh_i &= \phi(t_i), \end{aligned} \tag{1.14}$$

where $zh_i$ are discrete events generated by the plant and $rh_i$ are discrete events generated by the controller. The translation from discrete to continuous and vice versa are done with the interface equations

$$\begin{aligned} r(t) &= \gamma(rh_i) \\ zh_i &= \alpha(x(t)). \end{aligned} \tag{1.15}$$

The function $\alpha(x(t))$ is partitioning the state space into various regions. A plant event is generated each time a region is entered. These event can then be used by the controller. The partitioning must be detailed enough to give the controller sufficient information whether or not the current state is in an acceptable region. It must also be detailed enough for the controller to determine the next control signal value. The over-all systems can be viewed as two interacting discrete event systems and analyzed as such.

### Nerode-Kohn

Nerode and Kohn (1992) have developed a formal model for hybrid systems. The model provides a framework to represents the interaction between the continuous and discrete portions of the system and proposes stability definitions. They study the topological issues of hybrid systems. Small topologies are introduced for the design of the analog-to-digital map.

### Michael Tittus

Tittus (1995) models batch processes. The batch processes consist of continuous flows of material and energy with discrete actuators and sensors. The modeling is done with integrator processes, $\dot{x} = k_q$. The class of hybrid systems is very limited but these models are important for control of batch processes. Using such simple models he is able to derive result as stability and controllability of the systems.

### Petri nets and timed automata

Petri nets have been used extensively in modeling and analysis of discrete event systems. One example is Peleties and DeCarlo (1989) where the continuous plant is approximated with a Petri net and a supervisor consisting of two interacting Petri nets controls the plant.

## 1.6 Analysis

The way to analyze hybrid systems has been to convert the systems into either discrete or continuous systems. The reason for this is of course that the analytical framework for a control engineer deals only with pure discrete or pure continuous systems. There is a lack of theory dealing directly with mixed systems.

The analysis methods used for hybrid systems are often very customized to a specific problem or, in best case, to a small class of problems. The goal is to generalize all notions from pure control theory to hybrid control systems but there is a long way to go. Some analysis methods are reviewed in this section.

Perhaps the most fundamental feature to verify for a control system is the stability of the closed loop.

### Stability for special classes of hybrid systems

For some classes of hybrid systems there are methods to prove stability, for example, piecewise linear systems and fuzzy systems.

***Piecewise linear systems*** Stability of Piecewise linear systems described by

$$\begin{aligned} \dot{x} &= A_i x + a_i \\ x &\in X_i, \end{aligned} \tag{1.16}$$

where the state space $X$ is divided into regions $X_i$ such that $X = \cup_i X_i$ and $i$ belongs to an index set $I$ have been studied by several authors, see e.g. Ferron (1996), and for discrete systems, Sontag (1981). The basic idea is to try to find a quadratic matrix $P = P^T > 0$ such that

$$A_i^T P + P A_i < 0, \quad i \in I. \tag{1.17}$$

If this is possible then $V = x^T P x$ is a Lyapunov function for the piecewise linear system in Equation 1.16. Many systems do not admit such quadratic Lyapunov functions. There are three ways to extend the applicability of this method

1 Modify the definition of $X_i$

2 Allow piecewise quadratic functions $P_i$

3 Use other functions than quadratic as Lyapunov functions

The first method is rather limited and will always be very problem specific, thus it is difficult to define a general algorithm to do it.

The second method has recently been enhanced in Johansson and Rantzer (1997) and Johansson and Rantzer (1998). The search of piecewise quadratic Lyapunov functions is formulated as a convex optimization problem in terms of linear matrix inequalities. The idea, using piecewise Lyapunov functions, is to reduce conservatism while retaining efficient computation using convex optimization.

Chapter 4 introduces a design method for hybrid systems based on local controllers and local Lyapunov functions. The local Lyapunov functions can be of any kind and play an active part in designing the switching strategy. Some controller design methods such as LQG come with a Lyapunov function. In such a case it can be advantageous to be able to use that Lyapunov function.

***Fuzzy systems*** A method similar to piecewise linear systems is used in Kiendl and Rüger (1995). Takagi-Sugeno fuzzy systems are described by fuzzy IF-THEN rules that locally represent linear input-output relations of a system. The rules are of the following form:

$$\begin{aligned} Rule_i: \quad & IF\ x_1(k)\ is\ M_{i1} \ldots\ and\ x_n(k)\ is\ M_{in} \\ & THEN\ x(k+1) = A_i x(k) + B_i u(k). \end{aligned} \tag{1.18}$$

The control laws of fuzzy controllers are approximated by means of affine facet functions and stability is analyzed.

Caines and Ortega (1994) shows that a certain class of fuzzy controllers can be viewed as piecewise constant controls. This is then used to prove stability for nonlinear systems by using multiple Lyapunov functions.

## Controllability and observability of hybrid systems

Understanding the controllability property of a dynamics system is another important objective of control design and analysis. Many practical systems are nonlinear and operate in multiple state-space regions where dynamics vary significantly from one region to another.

***Hierarchical hybrid control systems*** Caines and Wei (1995) study the controllability problem for hybrid systems by dividing the state space into cells. The controllability problem is then divided into two layers: between block and in-block controllability. The controllability definitions are then combined so that global controllability is achieved

***Periodic systems*** Ezzine and Haddad (1989) construct observability and controllability matrices for periodic systems of the type

$$\begin{aligned}\dot{x} &= A(r(t))x(t) + B(r(t))u(t)\\ y(t) &= C(r(t))x(t),\end{aligned} \tag{1.19}$$

where r(t) is a deterministic scalar sequence taking values in the index set $N = \{1, 2, ..., n\}$. Even if the subsystems $[A(i), B(i), C(i)]$ are controllable and observable this is not necessarily true for the hybrid system in Equation 1.19.

### Aggregation

The analysis problem of a large hybrid system is immense. One method of breaking it down is to impose a hierarchical model structure where models residing on high levels only have a crude knowledge of the behavior. In Antsaklis (1997), pp 329-341, Pappas and Sastry make abstractions of hybrid systems. These abstractions should then generate the abstracted behaviors of the original systems. The method's advantage lies in that the abstracted systems are smaller and easier to analyze.

## 1.7 Verification of behavior

This research line has its roots in computer science. Computer programs are modeled as automata see Figure 1.8. There is a finite number of states and the transitions between the states are governed by logical expressions. These automata can be analyzed for properties as dead-lock, liveness, safety etc. In this basic automaton model there is no dynamics just static logic switching.

The method has been developed gradually to encompass more complex dynamical systems. Dynamics have been added to the nodes and the transitions are allowed to depend on the values of the states in the nodes. The development has gone from automata to Timed Automata, Alur and Dill (1994), and Hybrid Automata, Alur *et al.* (1993).

Currently it is allowed to have dynamics of the kind $\dot{x} = [k_1, k_2]$ i.e. integrator processes where the integration rate lies in an interval $[k_1, k_2]$.

Both the controller and the plant are modeled and it can the be verified that all nodes can be reached or that a certain reference trajectory can be followed with a certain resolution.

### Hybrid Automata

The paper Henzinger *et al.* (1995b) introduces a framework of hybrid automata as a model and specification language for hybrid systems. They

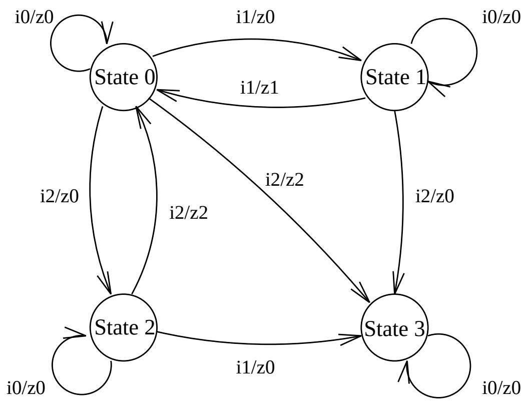

**Figure 1.8** An automaton without dynamics.

present two semi-decision procedures for verifying safety properties of piecewise-linear hybrid automata, in which all variables change at constant rates. The two procedures are based, respectively, on minimizing and computing fix-points on generally infinite state spaces. They show that if the procedures terminate, then they give correct answers. The procedures provide an automatic way for verifying the properties of the hybrid systems.

The discrete states of the controller are modeled by the vertices of a graph (control modes), and the discrete dynamics of the controller is modeled by the edges of the graph (control switches). The continuous states of the plant are modeled by points in $\mathbb{R}^n$, and the continuous dynamics of the plants are modeled by flow conditions such as differential equations.

## The verification problem

The verification problem can be formulated as: Given a collection of automata defining the system and a set of formulas of temporal logic defining the requirements, derive conditions under which the system meets the requirements. There are several software handling systems of different complexity levels. For timed automata there are COSPAN, KRONOS, Daws *et al.* (1996) and UPPAAL. One of the most advanced dealing with hybrid automata is HYTECH.

The next example shows how formal verification methods can be used to prove safety properties.

EXAMPLE 1.8—RAILROAD CROSSING
The railroad crossing problem is a typical example, see Halbwachs (1993). The problem is to investigate if a specific logic for opening and closing a

gate is safe under all possible conditions. The condition is simply that the gate is down when the train passes the crossing. The problem is that the speed of the train can vary and that the gate opening and lowering rates are limited. The specification consists of three models

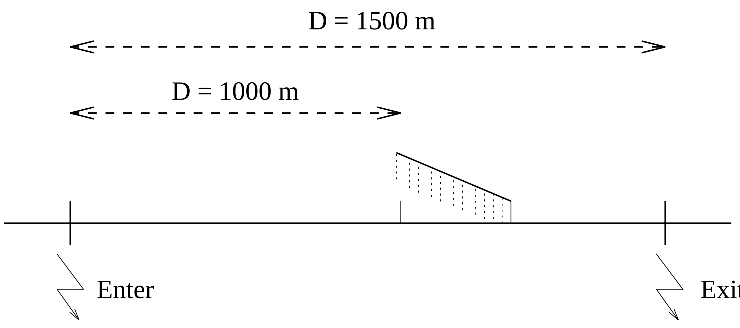

**Figure 1.9** The train enters the dangerous zone at *Enter* and leaves at *Exit*. While the train is in the dangerous zone the gate should be down.

- The train model
  - Continuous variable $x$ measuring distance from entry point.
  - Maximum speed: $\dot{x} \leq 60$ m/s.
  - *enter* is sent when entering the region.
  - *exit* is sent when leaving the region.
- The gate model
  - Continuous variable $\alpha$ measuring the gate angle.
  - Speed: $\dot{\alpha} = \pm 10$ degrees/s
- The controller model
  - Reads *enter* & *exit* signals from the train.
  - Emits *raise* & *lower* to the gate.

The three subsystems are modeled with one automaton each, see Figure 1.10.

The three subsystems are then combined into one, see Figure 1.11. Formal verification can then be applied to the combined model to see which nodes that are visited. An analysis shows that failure is possible if a train is approaching while the gate is going up. There is no support in the gate model for changing mode from going up to going down. □

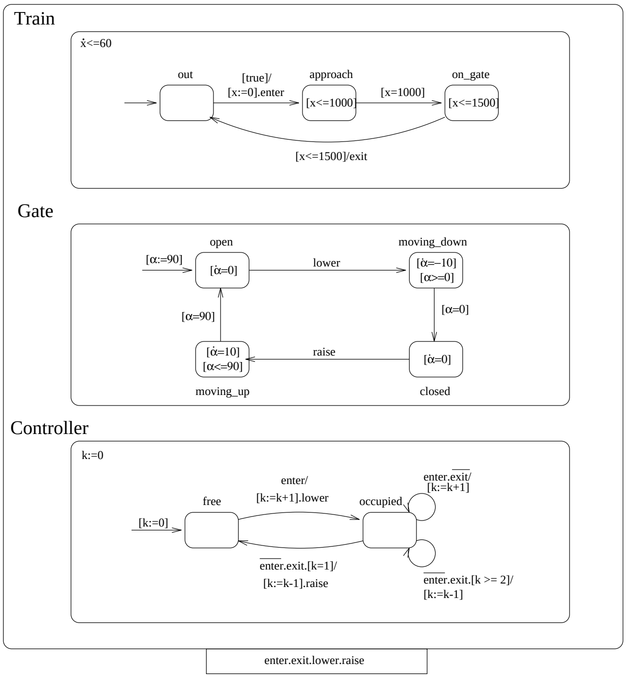

**Figure 1.10** Three automata, one for each subsystem.

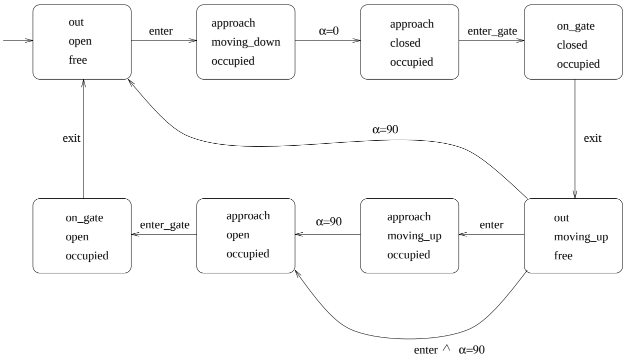

**Figure 1.11** The combined automaton for Example 1.8.

## 1.8 This thesis

This introduction chapter has defined hybrid systems and described some advantages and problems with using them for modeling and control. There is not yet a unified approach to analysis and design of hybrid systems. It was seen that there are many different models and that several 'old' controller design methods could be viewed as hybrid systems.

The rest of this thesis consists of four chapters. They treat different aspects of hybrid control systems: Analysis of fast mode changes, simulation of hybrid system, a design method and experiments with hybrid control systems.

### Fast mode changes

Chapter 2 is a rather theoretical chapter dealing with the problem of defining solutions to differential equations with discontinuous control. A special interest will be the study of what happens in the case of infinitely fast mode changes. The analysis has its roots in Filippov's theory for differential equations with discontinuous right hand sides, see Filippov (1988). The chapter is based on the paper Malmborg and Bernhardsson (1997).

### Simulation of hybrid systems

The analysis tools available for hybrid systems are not nearly enough. Therefore researchers must rely on simulation to assess the performance of hybrid system. Today's simulation tools are not good enough to handle hybrid systems. Chapter 3 addresses some problems concerning simulation of hybrid systems. Further, there will be suggestions of some features to be added to simulation tools to improve their reliability. The ideas of this chapter are found in Malmborg and Bernhardsson (1997).

### A design method

In Chapter 4 it is shown how hybrid controllers can be used to improve the performance of a control system. A hybrid controller design method that guarantees stability will be presented. In this chapter there will also be a warning and explanation why some of the stability results in the literature on hybrid control systems are false. The chapter is an extended version of Malmborg *et al.* (1996).

### Experiments with two hybrid control systems

The last chapter is a presentation of experimental results. The design method from Chapter 4 is applied to two processes: Level control of a double-tank process and air temperature control in a school. The result in both cases is much faster responses to set-point changes. The double-tank system control was presented in Malmborg and Eker (1997).

# 2

# Fast mode changes

## 2.1 Introduction

As discussed in the previous chapter there are many ways to model hybrid control systems. The common feature that the state space $\mathcal{S}$ has both discrete and continuous variables leads to new possible behaviors of the overall control system. This chapter analyzes higher order sliding and sliding on multiple surfaces. The two-relay case in $R^2 \times R^{n-2}$ is treated in more detail. The term sliding will also be used for the case where the system experiences a rapid cyclic switching.

For many of the proposed hybrid models, restrictions are introduced to prevent infinitely fast switching between the discrete modes. For instance in Tavernini (1987) the distance between any two sets with different discrete transitions is bounded away from zero and the next set from which another discrete transition takes place is at least a fixed distance away. After a discrete transition the new starting point is in an open set on which the dynamics are well defined.

If no such restrictions are imposed on the control system design then it is quite common to get sliding modes, i.e. cycles of infinitely fast mode changes. The resulting dynamics are not always well defined and that is an indication that the modeling must be refined. Infinitely fast switches and under-modeling are always difficult to deal with for simulation tools.

### An example

To give a flavor of the difficulties that can exist, a simple example with two states and one relay is examined.

EXAMPLE 2.1—A SIMPLIFIED IVHS
The Equations 2.1 describe a simple model of a vehicle with a control system. The control objective is to drive along the $x_2 = 0$ axis. The equations

for $x_2 \neq 0$ are

$$\begin{aligned} \dot{x}_1 &= \cos(u) \\ \dot{x}_2 &= -\sin(u) \\ y &= x_2 \\ u &= \operatorname{sgn}(y), \end{aligned} \tag{2.1}$$

where sgn$(y)$ is defined as

$$\operatorname{sgn}(y) = \begin{cases} 1 & y > 0 \\ \textit{undefined} & y = 0 \\ -1 & y < 0. \end{cases} \tag{2.2}$$

The applied control is such that the car always points towards the $x_2 = 0$ subspace, see Figure 2.1. The direction $u$ of the car is controlled with

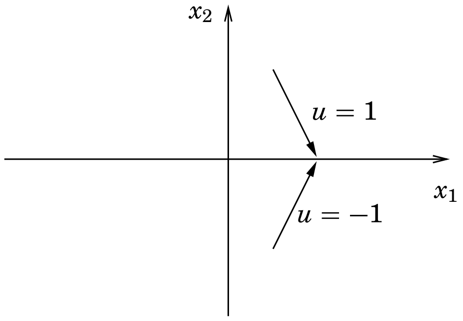

**Figure 2.1** Transversal sliding in Example 2.1

the relay $u = \operatorname{sgn}(y)$. The system description can be aggregated into the hybrid automaton in Figure 2.2. The equation $y \geq 0$ indicates that the transition from $mode_2$ to $mode_1$ is enabled if $y \geq 0$.

The dynamics in the two modes are

$$\begin{aligned} f_1 : \dot{x}_1 &= \cos(1), \quad \dot{x}_2 = -\sin(1) \\ f_2 : \dot{x}_1 &= \cos(1), \quad \dot{x}_2 = \sin(1). \end{aligned} \tag{2.3}$$

Note that both the transition from $mode_1$ to $mode_2$ and the transition from $mode_2$ to $mode_1$ are enabled for $y = 0$ and that fast switching is possible on the subspace $y = x_2 = 0$. On that subspace the dynamics are in fact not well defined by Equation 2.1. □

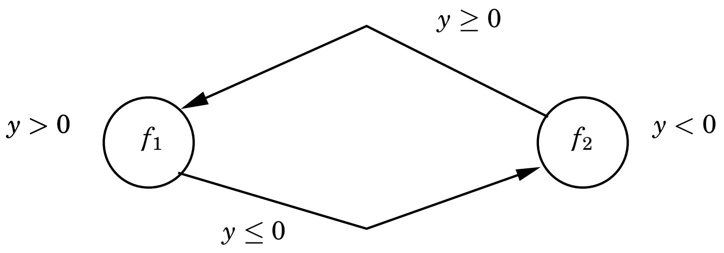

**Figure 2.2** The hybrid automaton for Example 2.1.

It would be advantageous if a simulation or analyzing tool automatically could detect the structural possibility of fast switching in large systems with several relays, extend the system with so called induced modes so that the new system has only well-defined transitions without fast switching and describe how these modes should inherit dynamics from the basic system. The new automaton should have one induced mode and only well-defined transitions, see Figure 2.3. For Example 2.1 the new mode represents the part of the state space where $x_2 = 0$.

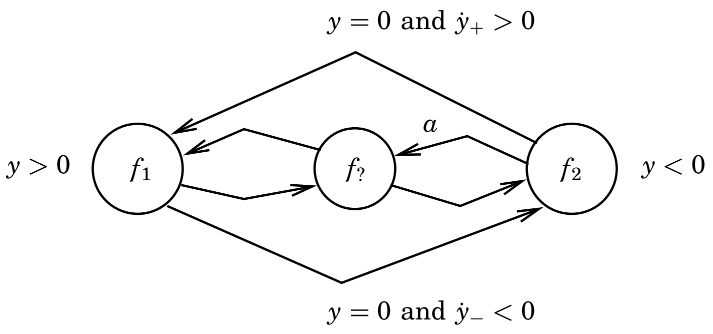

**Figure 2.3** System with one new mode. Only one induced transition is indicated, $a : (y = 0 \text{ and } \dot{y}_+ < 0)$

The introduction of the new modes is nontrivial. To determine what the dynamics should be often requires more information about the salient features of the relay. There are several possible definitions of the inherited dynamics. To determine which is physically motivated more modeling is typically needed.

DEFINITION 2.1—SLIDING SET
Denote the transition functions $\sigma_{ij}(x, \dot{x}, \ddot{x}, \dots)$. A transition from state $i$

to state $j$ is enabled if $\sigma_{ij}(x, \dot{x}, \ddot{x}, \dots) \geq 0$. Let the set $\mathcal{M}_I$ be defined as

$$\begin{aligned} \mathcal{M}_I &= \{x : \sigma_{(ij)_1} \geq 0 \ \ \cap \dots \cap \sigma_{(ij)_k} \geq 0\} \\ I &= \{(ij)_1, \dots, (ij)_k\}. \end{aligned} \tag{2.4}$$

If for an index set $I$ such that the $\sigma_{ij}$ form a loop $(i_1 = j_k)$ the set $\mathcal{M}_I$ is nonempty then $\mathcal{M}_I$ is called a sliding set. □

In the rest of this chapter $\sigma_{ij}$ will be input functions to relays.

EXAMPLE 2.2—SLIDING SET FOR ONE RELAY
A relay $u = \text{sgn}(x)$ is governing the transition between two modes. The mode changes from $mode_1$ to $mode_2$ when the relay input goes from a positive value to a negative value and vice versa. If $\sigma_{12} \geq 0$ when $x \geq 0$ and $\sigma_{21} \geq 0$ when $x \leq 0$, then the sliding set is $x = 0$. □

Whether there is any sliding or not depends on the vectorfields in the neighborhood of the sliding set.

EXAMPLE 2.3—EXAMPLE 2.1 CONTINUED
There are at least two natural candidate definitions for the dynamics in the new mode, $x_2 = 0$. Either choose $u = 0$ in order to stay on $x_2 = 0$, this is called equivalent control and gives the equation

$$\begin{aligned} \dot{x}_1 &= \cos 0 = 1 \\ \dot{x}_2 &= 0, \end{aligned} \tag{2.5}$$

or else alternate between control signals $u = 1$ and $u = -1$, which gives the convex solution

$$\begin{aligned} \dot{x}_1 &= \frac{1}{2}(\cos(-1) + \cos(1)) = \cos(1) \\ \dot{x}_2 &= 0. \end{aligned} \tag{2.6}$$

Which solution to choose depends on the physical system that the relay models. This is further discussed in the next section. □

## 2.2 Dynamics inheritance

The new modes, such as the mode for $x_2 = 0$ in Example 2.1 will have a set of dynamics that sometimes can be derived from the dynamics of the adjacent modes. This procedure is denoted dynamics inheritance. Already the small Example 2.1 revealed that several definitions resulting in different dynamics were possible. For many reasons, e.g. simulation and stability analysis, the dynamics on the sliding surface have to be calculated and known in detail. The rest of this chapter treats the question of how do define the dynamics for some different cases of sliding and related motions. An important issue is to decide if the inherited dynamics can be defined in a unique way. Several different sliding-type motions will be investigated: First order sliding, second order sliding, $n$th order sliding and sliding on multiple sliding surfaces.

## Transversal vectorfields – first order sliding

This section summarizes the results on sliding motion where the controllers give rise to vectorfields that have a nonzero component perpendicular to the sliding surface. This is called transversal sliding or first order sliding. The system class under consideration here is

$$\begin{aligned} \dot{x} &= f(x,u), \quad x \in \mathbb{R}^n \\ u &= [u_1 \quad u_2 \quad \dots \quad u_m]^T \\ u_i &= \operatorname{sgn}(y_i), \quad i = 1, \dots, m. \end{aligned} \tag{2.7}$$

How to define the sliding motion along a surface, such as $x_2 = 0$ in Example 2.1, has been studied by several researchers, see Filippov (1988), Utkin (1977) and Clarke (1990). For a smooth surface described by the intersection of $m$ smooth surfaces $y_i(x) = 0, \ i = 1, \dots m$ the dynamics along the intersection can be defined in at least two possible ways. Which one that is more appropriate depends on how the switches are modeled in detail. Relays can be used for implementing the switches and in practice the relays are not perfect. Two approximations are shown in Figure 2.4. For the continuous approximation the relay output $u$ is changed gradually

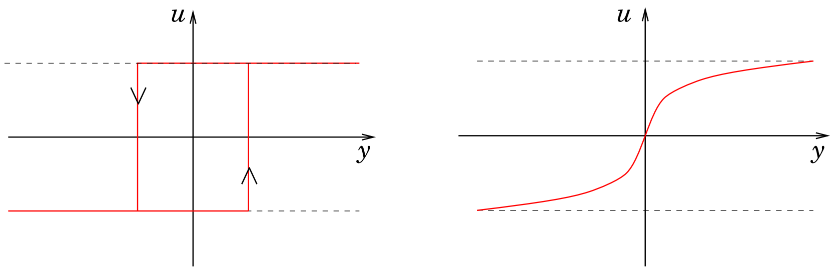

**Figure 2.4** Approximations of the relay switch $u = \operatorname{sgn}(y)$. Hysteresis (left) or continuous function (right)

as the relay input $y$ changes. For the approximation with a hysteresis the output is changed infinitely fast but only as the input exceeds a threshold level. The two definitions used in this thesis are the definitions introduced by Filippov and Utkin.

DEFINITION 2.2—FILIPPOV CONVEX DEFINITION
The function $x(t)$ is called a solution of Equation 2.7 if $x(t)$ is differentiable almost everywhere and

$$\dot{x}(t) \in \mathrm{co}\{f(x,u)\}, \ \textit{where} \ u_i = \begin{cases} \mathrm{sgn}(y_i), & y_i \neq 0 \\ \{-1,1\}, & y_i = 0 \end{cases} \tag{2.8}$$

In 2.8, *co* denotes the convex hull and the differential inclusion can be written as

$$\dot{x} \quad = \sum_{u \in \{-1,1\}^m} \alpha_u(t) f(x,u),$$

where $\sum_i \alpha_u(t) = 1$ and $\alpha_u(t) \geq 0$. The sum is taken over all combinations of the $u$-states. If sliding on a surface $y_i = 0$ occurs, then $\alpha_u$ is derived from the equations $\frac{\partial y_i}{\partial x}\dot{x} = 0$. □

This definition has its roots in optimal control under the name of extended control and from generalized derivatives in non-smooth analysis, see Clarke (1990). The definition can be motivated by a limiting process where the relay equations are replaced by $u_i(t) = \mathrm{sgn}(y_i(t - \varepsilon_i))$, where $\varepsilon_i \to 0+$. This can for example be a good approximation if the relays are implemented on a digital computer. This definition can also be motivated by hysteresis in space, see Figure 2.4 (left).

An alternative definition of the sliding mode dynamics is the following.

DEFINITION 2.3—UTKIN EQUIVALENT CONTROL
The function $x(t)$ is a solution to Equation 2.7 if $x(t)$ is differentiable almost everywhere and the if $m$ discrete variables in $u$ can be relaxed to continuous variables in $[-1,1]$ and an equivalent control $u_{eq}(t) \in [-1,1]^m$ can be found such that

$$\dot{x} = f(x, u_{eq}). \tag{2.9}$$

If sliding occurs the equivalent control is derived from the equations $y_i = 0$ and $\dot{y}_i = 0$. □

This definition can be motivated by a limiting process where each relay is approximated by a continuous function $\mathrm{sgn}_\varepsilon(x)$ that tends to $\mathrm{sgn}(x)$ pointwise as $\varepsilon \to 0^+$. This can be a good approximation if the relays are implemented with analog components, see Figure 2.4 (right). The Filippov convex definition and the Utkin equivalent control definition coincide if $f(x,u)$ is affine in $u$.

For Example 2.1 the convex definition gives the dynamics $\dot{x}_1 = \cos(1)$ along the switching line $x_2 = 0$. The equivalent control definition gives $\dot{x}_1 = \cos(0) = 1$. Both definitions can be natural candidates for the physical behavior of the car. If the turning of the wheel can be done very fast, but it takes a while to observe that the $x_2 = 0$ line has been crossed, then the convex definition is appropriate. If the turning is slow, then it is perhaps better modeled with the continuous approximation of the relay i.e by equivalent control.

## Multiple switching surfaces

If there are more than one relay, i.e. if $m > 1$ then equations 2.8 or 2.9 are not sufficient to define the sliding motion uniquely. The sliding takes place on an $(n-m)$-dimensional manifold. The number of unknowns, $\alpha_u$, is $2^m - 1$ and there are only $m$ equations. The case with several relays is in fact not very well understood. The "physical" sliding motion will depend on the salient features of the different relays, e.g. which relay is the fastest. The problem with two relays is investigated in Section 2.4.

## Non-transversal sliding – higher order sliding

In the literature it is common to assume transversal intersection of vectorfields and switching surfaces, i.e. $f_i^T \nabla \sigma_j \neq 0$, where $\sigma_j = 0$ define the sliding sets and $f_i$ the vectorfields. Non-transversal intersections can however arise quite naturally and should not be considered degenerate or non-generic. To see this, Example 2.1 is extended with a third equation describing sensor dynamics.

EXAMPLE 2.4—IVHS EXAMPLE WITH SENSOR DYNAMICS
The new equation set is

$$\begin{aligned} \dot{x}_1 &= \cos u \\ \dot{x}_2 &= -\sin u \\ \dot{x}_3 &= -x_3 + x_2 \\ y &= x_3 \\ u &= \operatorname{sgn}(y). \end{aligned} \tag{2.10}$$

The switching surface is now given by $x_3 = 0$ and there is non-transversal sliding on the subspace $x_2 = x_3 = 0$. Figure 2.5 shows a typical trajectory close to the $x_1$-axis. Note that $\frac{\partial y}{\partial x} \cdot f(x,u) = 0$ on this line for all $u$. Thus this equation cannot be used for finding the equivalent control $u_{eq}$ or the $\alpha_u$ in the convex definition.

To define the sliding dynamics it can be motivated to extend the sliding definitions for transversal vectorfields in the following way: differentiate $y(x)$ with respect to $t$ until it is possible to solve for $u$.

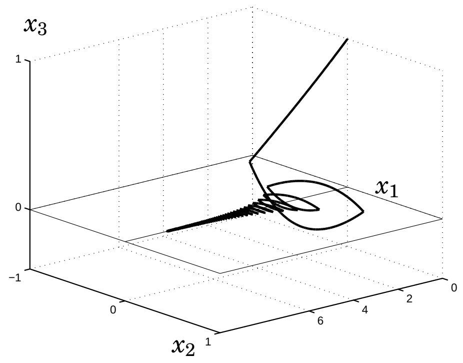

**Figure 2.5** IVHS example with stable sensor dynamics. There is sliding on the subspace $x_2 = x_3 = 0$.

For this example the convex definition is derived from equations $\dot{y} = 0$ and $\ddot{y} = 0$. The equivalent control is given by solving $y = 0, \dot{y} = 0$ and $\ddot{y} = 0$ for $u$. (For both cases this gives the same sliding behavior in the $x_1$ direction as without sensor dynamics.) There is still a difference depending on if the convex or equivalent control definition is used since $(\dot{x}_1)_{eq} = 1$ and $(\dot{x}_1)_{convex} = \cos(1)$.

Note that initial conditions close to the subspace $x_2 = x_3 = 0$ give approximately the convex solution. Hence this might be the best definition of the dynamics on the subspace $x_2 = x_3 = 0$. □

Necessary and sufficient conditions for existence of higher order sliding for systems with one relay are unknown. A necessary condition is that $\frac{\partial}{\partial u}y^{(k)} < 0$ where $k$ is the smallest integer such that $\frac{\partial}{\partial u}y^{(k)} \neq 0$. The number $k$ is called the nonlinear relative degree, see Nijmeijer and van der Schaft (1990). If $f(x,u) = a(x) + b(x)u$ and $y = h(x)$ the condition can be written $L_b L_a^{k-1} h(x) < 0$. Here $L_a$ denotes the Lie-derivative in the direction of $a(x)$, i.e $L_a h = \sum \frac{\partial h_i}{\partial x} a_i$. If the relative degree $k$ is greater than one the intersection is non-transversal. Transversal intersections are hence generic only in the same weak sense as "linear systems are generically of relative degree one".

Sufficient conditions for non-transversal sliding are hard, since stability of the switching set must also be studied. If for instance the sensor dynamics equation above is changed to $\dot{x}_3 = x_3 + x_2$ the switching line

$x_2 = x_3 = 0$ is better described as being unstable and there will not be any sliding motion along/around the line lasting a long time. The example motivates the following investigation.

## 2.3 Non-transversal sliding of degree two

A system with two discrete modes is modeled with the equations

$$\begin{aligned} \dot{x} &= f_u(x) \\ u &= \operatorname{sgn}(\sigma(x)) \\ \dot{\sigma}(x) &= \frac{\partial \sigma}{\partial x} f_u(x). \end{aligned} \tag{2.11}$$

Switching between the modes takes place when $\sigma = 0$. To begin with, the switching surface is transformed to $\sigma(x) = y = 0$. Any smooth discontinuity surface can be transformed to this with a local change of variables that can be found from the implicit function theorem. The following discussion assumes that there has been a coordinate transformation to arrive at $y = \sigma(x) = 0$. Non-transversal sliding takes place on a subspace where $y = \dot{y} = 0$. Assume further that another smooth transformation can be found as to have the sliding subspace at $\{x : x_1 = 0, x_2 = 0\}$. In order to investigate the dynamics on such a subspace the following definitions are done. The subspace $\mathcal{S}_{12}$ is defined as

$$\mathcal{S}_{12} := \{x_1 = 0, x_2 = 0\}, \tag{2.12}$$

and the $\varepsilon$-environment of the subspace $\mathcal{S}_{12}$ is

$$\mathcal{S}_{12}(\varepsilon) := \{|x_1| < \varepsilon, |x_2| < \varepsilon\}. \tag{2.13}$$

Definition 2.4—Locally stable subspace
A subspace $\mathcal{S}_{12}$ is said to be locally stable if for all $\varepsilon > 0$ there exists $\delta > 0$ so that $x(0) \in \mathcal{S}_{12}(\delta) \Rightarrow x(t) \in \mathcal{S}_{12}(\varepsilon), t \geq 0$. □

### Stable second order sliding

The stability issue is now solved for the case $k = 2$. The analysis was inspired by Filippov (1988) pp. 234-238 where he does a similar analysis for a two dimensional case. Let $x(t), y(t), z(t) \in \mathbb{R} \times \mathbb{R} \times \mathbb{R}^{n-2}$ be given by the following equations if $y > 0$

$$\begin{aligned} \dot{x} &= P^+(x, y; z) \\ \dot{y} &= Q^+(x, y; z) \\ \dot{z} &= R^+(x, y; z), \end{aligned}$$

and if $y < 0$ by

$$\begin{aligned}\dot{x} &= P^-(x, y; z)\\ \dot{y} &= Q^-(x, y; z)\\ \dot{z} &= R^-(x, y; z),\end{aligned}$$

where $P^\pm$, $Q^\pm$, $R^\pm \in C^2$. The plane $y = 0$ is hence a discontinuity plane for the dynamics, see Figure 2.6.

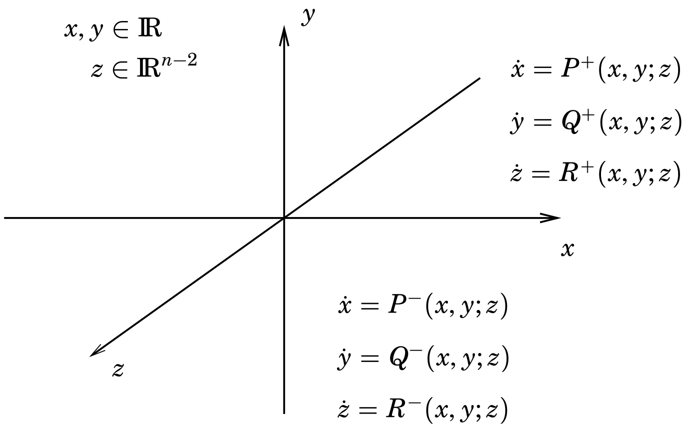

**Figure 2.6** The discontinuity plane is $y = 0$.

Assume that $Q^\pm(0, 0; z) = 0, \forall z$, which is the case when there can be non-transversal sliding along the $x = y = 0$ subspace. With the sign conditions

$$xQ^+(x, 0; z) < 0, \qquad xQ^-(x, 0; z) < 0, \quad x \neq 0, \tag{2.14}$$

there is no sliding in the plane $y = 0$ unless possibly along the subspace $x = y = 0$. Further sign conditions assumed are

$$P^+(0, 0; z) > 0, \qquad P^-(0, 0; z) < 0. \tag{2.15}$$

From Equation 2.14 follows that $Q_x^\pm(0, 0; z) \leq 0$. If this condition is sharpened to $Q_x^\pm(0, 0; z) < 0$ the following theorem can be proved.

THEOREM 2.1—SECOND ORDER STABLE SLIDING
For the system described above, the subspace $\mathcal{S}_{yx}$ is locally stable around the point $(0, 0; z)$ if $a_2(z) = A^+ - A^- < 0$, where

$$A^\pm = \left(\frac{P_x + Q_y}{P} - \frac{Q_{xx}}{2Q_x} + \frac{(Q_x P_z - P Q_{xz})R}{Q_x P^2}\right)^\pm, \tag{2.16}$$

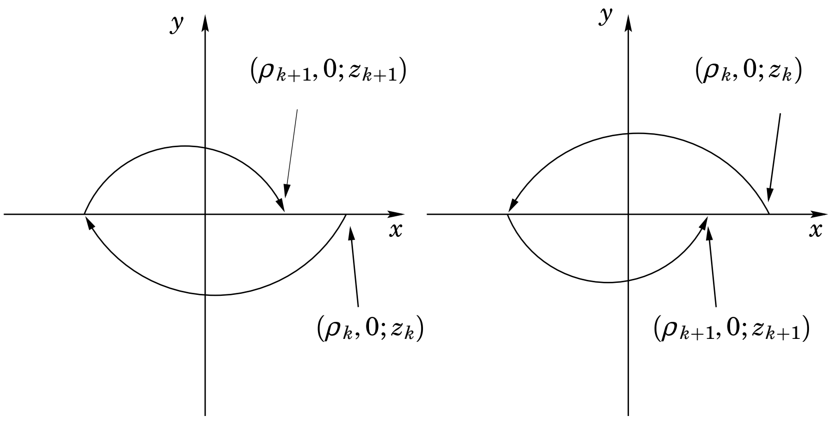

**Figure 2.7** The $y-x$-plane. The $z$-directions, $z \in \mathbb{R}^{n-2}$, are omitted for simplicity. The intersections with the $y = 0$ plane for $x > 0$ are denoted $\rho_k$.

where all functions are evaluated at $(0, \pm 0; z)$. The sliding dynamics on the subspace $\mathcal{S}_{yx}$ are defined by:

$$\dot{z} = \alpha^+ R^+(0,0;z) + \alpha^- R^-(0,0;z), \tag{2.17}$$

where $\alpha^+$ and $\alpha^-$ are uniquely defined by

$$\begin{aligned} \alpha^+ + \alpha^- &= 1 \\ \alpha^+ P^+ &= \alpha^- P^-. \end{aligned} \tag{2.18}$$

Thus the convex definition is the natural solution if the dynamics on the subspace $\mathcal{S}_{yx}$ should be the limit of dynamics just off it The series of intersections with the $y = 0$ plane for $x > 0$ is denoted $[0, \rho_k, z_k]$. The sequence $\rho_k$ is monotonously decreasing with

$$\rho_{k+1} = \rho_k + a_2(z)\rho_k^2 + O(\rho_k^3). \tag{2.19}$$

□

***Proof*** See Appendix A.

***Remark*** Note that the rotational direction and the corresponding sign conditions, equations 2.14 and 2.15 can be changed. The rotation in Figure 2.7 (left) is used in the proof of Theorem 2.1. The rotation in Figure 2.7 (right) is the natural choice for affine systems on normal form, see Section 2.5, with $y = x_1$. If the rotational direction is changed then the stability condition is also reversed, $a_2(z) = A^+ - A^- < 0$ becomes $a_2(z) = A^+ - A^- > 0$.

EXAMPLE 2.5—IVHS, WITH FILTER DYNAMICS

Introduce the filter parameter $a$ in $\dot{x}_3 = -ax_3 + x_2$ and check the stability conditions for

$$\begin{aligned} \dot{x}_1 &= \cos(u) &&= R \\ \dot{x}_2 &= -\sin(u) &&= P \\ \dot{x}_3 &= -ax_3 + x_2 &&= Q \\ y &= x_3 \\ u &= \operatorname{sgn}(y). \end{aligned} \tag{2.20}$$

Apply Equation 2.16 to derive $A^+$ and $A^-$

$$A^+ = \frac{-a}{-\sin(1)}, \quad A^- = \frac{-a}{-\sin(-1)}. \tag{2.21}$$

The rotation direction is as in Figure 2.7 (right) and the stability condition is thus $a_2(x_1) > 0$,

$$a_2 = \frac{2}{3}(A^+ - A^-) = \frac{4a}{3\sin(1)} > 0. \tag{2.22}$$

It is now easy to see that using a stable filter, i.e $a > 0$, leads to a stable sliding motion. An unstable filter gives an unstable sliding manifold.

***Simulation of Example 2.5*** Figure 2.8 (left) shows a simulation of Example 2.5. It is a phase portrait in the $x_2$ and $x_3$ coordinates and the $x_1$ coordinate is omitted. The velocity in the $x_1$-direction is cos(1). Figure 2.8 (right) shows an estimate of $a_2(x_1)$ based on the formula derived in Appendix A.

$$\rho_{k+1} = \rho_k - \rho_k^2 \cdot \hat{a}_2(x_1), \tag{2.23}$$

where $\rho_k$ is defined as the $x_2$-coordinate for the intersections with the $(x_3 = 0)$-plane for $x_2 > 0$. The limiting value for $a_2(x_1)$, as $k$ tend to infinity, is $\frac{4a}{3\sin(1)} = 1.58$, for $a = 1$.

***What about convergence rates?*** The time between two consecutive crossings with $(x_3 = 0, x_2 > 0)$ is approximately proportional to the $x_2$ coordinate of the starting point $[\,x_1(0), x_2(0), 0\,]^T = [\,x_1(0), \rho_0, 0\,]^T$. Denote the time for lap $k$ as $t_k$. The recursive equation for lap times hence becomes

$$t_{k+1} \;=\; t_k - t_k^2 \cdot \frac{2}{3}(A^+ - A^-) + O(t_k^3). \tag{2.24}$$

How long time does it take to reach the subspace $x_2 = x_3 = 0$?

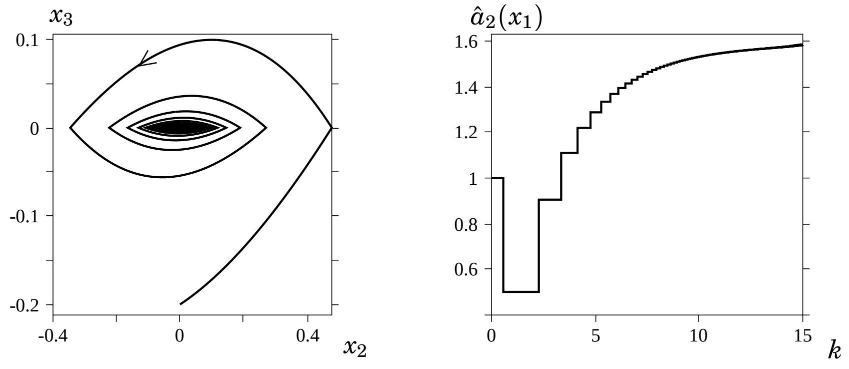

**Figure 2.8** The $x_2$-$x_3$ phase plane (left) and the estimate, $\hat{a}_2(x_1)$, of the stability function (right). The theoretical limiting value is 1.58.

LEMMA 2.1—TIME TO REACH THE SUBSPACE $x_2 = x_3 = 0$
Given Equation 2.24 for the lap times, $t_k$ then the sum

$$T = \sum_{k=1}^{\infty} t_k \tag{2.25}$$

diverges.

***Proof*** Re-scale the time variable to $\tau_k = -\frac{2}{3}(A^+ - A^-)\cdot t_k$. The difference equations is now $\tau_{k+1} = \tau_k(1-\tau_k)$, with $\tau_1$ small. For any large value of $k_0$, a constant $C$ can be chosen such that $0 < \frac{1}{Ck_0} < \tau_{k_0} < \frac{C}{k_0}$. Then using Equation 2.24 it is easily seen that $0 < \frac{1}{Ck} < \tau_k < \frac{C}{k}$ is true for all $k > k_0$. The sum is bounded by two positive diverging sums and thus diverging itself. □

This will cause severe problems to simulation tools. The trajectory will get closer and closer to the subspace $x_2 = x_3 = 0$ and the time between two consecutive switches will become shorter and shorter. An advanced simulation tool could notice this and introduce a new mode, modeling the dynamics on the subspace $x_2 = x_3 = 0$. When the trajectory comes close to this subspace the simulation could turn to this new mode.

## Affine systems

If the system is affine in $u$ then the expression for the $A^{\pm}$-function can be simplified. Start with an affine system

$$\begin{aligned} \dot{x} &= f(x) + g(x)\cdot u, \qquad && x \in \mathbb{R}^n,\ u \in \mathbb{R} \\ y &= h(x), && y \in \mathbb{R}, \end{aligned} \tag{2.26}$$

of nonlinear relative degree two, i.e. with $L_g h = 0$ in Equation 2.26 and apply a nonsingular coordinate transformation

$$z = S(x) = [\,h(x) \quad L_f h(x) \quad s_3(x)\,]^T.$$

The vectorfield pattern in Figure 2.7 (right) applies with $y = z_1, x = z_2$. After transformation the new state equations are

$$\begin{aligned}
\dot z_1 &= \dot h(x) = (L_f + u \cdot L_g)h(x) = L_f h = z_2 &&= \overline{Q} \\
\dot z_2 &= (L_f + uL_g)L_f h(x) = (L_f^2 + uL_g L_f)h(x) &&= \overline{P} \\
\dot z_3 &= f_3(x) + g_3(x)u = \frac{\partial s_3}{\partial x}(f(x) + g(x)u) &&= \overline{R}.
\end{aligned} \tag{2.27}$$

The sign and relative degree conditions in equations 2.14 and 2.15 become

$$\begin{aligned}
\overline{Q}^{\pm}_{z_2} &= 1 > 0 \\
\overline{P}^{+} &= (L_f^2 + L_g L_f)h(x) < 0 \\
\overline{P}^{-} &= (L_f^2 - L_g L_f)h(x) > 0.
\end{aligned} \tag{2.28}$$

The equation for $A^{\pm}$ is reduced to

$$A^{\pm} = \frac{1}{\overline{P}}(\overline{P}_{z_2} + \frac{\overline{P}_{z_3}\overline{R}}{\overline{P}}), \tag{2.29}$$

where again all functions are evaluated at $(0^{\pm}, 0; z_3)$ and the stability conditions is $A^+ - A^- > 0$.

***Linear dynamics*** Non-transversal sliding for linear dynamics and one relay was studied in Johansson and Rantzer (1996). They worked with the system

$$\begin{aligned}
\dot x &= Ax + Bu \\
y &= Cx \\
u &= -\operatorname{sgn}(y) \\
x &\in \mathbb{R}^n, u, y \in \mathbb{R},
\end{aligned} \tag{2.30}$$

and proved that there can only be non-transversal sliding if $CB = 0$ and $CAB > 0$. Their results are consistent with the results for nonlinear systems derived in this section.

**What about higher order stable sliding?**

For simplicity only affine systems are treated here and the result is that there can be no $n$th order sliding where $n \geq 3$. Without loss of generality the systems can be written as

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_3 \\ \dot{x}_3 &= a(x) + b(x)u \\ \dot{x}_4 &= f_r(x) \\ y &= -x_1 \\ u &= \operatorname{sgn}(y) \end{aligned} \tag{2.31}$$

DEFINITION 2.5—THIRD ORDER SLIDING
The third order sliding manifold of Equation 2.31 is defined as the set

$$\mathcal{S}_{123} := \{x_1 = 0, x_2 = 0, x_3 = 0\}.$$

□

The following definition is needed to check if infinitely many switches take place in finite time.

DEFINITION 2.6—FAST SWITCHING POINTS
A point $\bar{x} \in \mathcal{S}_{123}$ is denoted a fast switching point if $\forall \tau > 0$ and $\forall M$ there $\exists \delta > 0$ such that if $x(0) \in B_\delta(\bar{x}) \setminus \mathcal{S}_{123}$ then there $\exists x(t)$ that satisfies Equation 2.31 at almost all $t \in [0, \tau]$ such that

- there will be at least $2M$ switches in the time interval $[0, \tau]$,
- $\|x(t_{2M}) - \mathcal{S}_{123}\| \leq \|x(0) - \mathcal{S}_{123}\|$,

where $t_{2M}$ is the time of the $2M$th switch. □

LEMMA 2.2—A NECESSARY CONDITION FOR FAST SWITCHING IS $|b(\bar{x})| \geq |a(\bar{x})|$
Assume that $|b(\bar{x})| < |a(\bar{x})|$. If $u$ switches from $+1$ to $-1$ or vice versa, then given an arbitrary short time interval $\tau$, $x$ will be close to $\bar{x}$ and the sign of $a(x) + b(x)u$ will be constant, hence $x_3$ changes sign at most once, $x_2$ changes sign at most two times and $x_1$ changes sign at most three times in the same time interval. □

THEOREM 2.2—THERE ARE NO FAST SWITCHING POINTS FOR AFFINE SYSTEMS OF ORDER 3.
Assume that $b(\bar{x}) > |a(\bar{x})| \geq 0$ and that $a, b, f_r$ are Lipschitz continuous. Then $\bar{x}$ is not a fast switching point.

***Proof*** Partial integration gives for any $\phi \in C^2$

$$\int_{t_1}^{t_2} \phi(t)\,dt = \frac{t_2-t_1}{2}(\phi(t_2)+\phi(t_1)) - \int_{t_1}^{t_2} \frac{(t-t_1)(t_2-t)}{2}\phi''(t)\,dt.$$

Now apply this to $\phi(t) = x_2(t)$ in Equation 2.31 with $t_k$ as the times where $x_1(t) = 0$. The expression for the state $x_2$ at the next switching time is

$$x_2(t_2) = -x_2(t_1) + \int_{t_1}^{t_2} \frac{(t_2-t)(t-t_1)}{t_2-t_1}(a+bu)\,dt.$$

Continuing to next switch,

$$\begin{aligned} x_2(t_2) &= x_2(t_0) \\ &\quad + \int_{t_1}^{t_2} \frac{(t_2-t)(t-t_1)}{t_2-t_1}(a+b)\,dt - \int_{t_0}^{t_1} \frac{(t_1-t)(t-t_0)}{t_1-t_0}(a-b)\,dt. \end{aligned} \tag{2.32}$$

Since $\tau$ is small and the interval between switches is small continuity gives that $a(x) \pm b(x)u$ are almost constant and hence do not change sign in the intervals $t_1 - t_0$ and $t_2 - t_1$ respectively. Hence $x_2(t_2)$ can be approximated with

$$x_2(t_2) = x_2(t_0) + \frac{(t_2-t_1)^2}{6}(b+a) + \frac{(t_1-t_0)^2}{6}(b-a).$$

Hence, the sequence $x_2(t_{2k})$ is growing and therefore the second condition in Definition 2.6 is not fulfilled. □

## 2.4 Two-relay systems

If there are several relays and thus several switching surfaces the situation gets more complicated as there can be sliding on several switching surfaces simultaneously. In this situation it is not enough to choose definition, Filippov or Utkin, for the sliding dynamics because there is not enough equations to define unique dynamics. The sliding dynamics will in general depend on the salient features of the relays. In this section it is shown that if the relays work on different time scales, i.e. the difference in switching times are large then it is possible to derive unique sliding dynamics for simultaneous sliding on two switching surfaces. The dynamics are again

$$\dot{x} = f(x,u), \quad x \in \mathbb{R}^n, u \in [0,1]^2, \tag{2.33}$$

and in the analysis that follows it is assumed that the relays are defined as $u_i = \mathrm{sgn}(x_i)$ and that the vectorfields $f(x, u_1, u_2)$ are constant. That the inputs of the relays are the coordinates $x_i$ is not a severe restriction since this situation can often be achieved by a smooth transformation of the original inputs. In the following discussion it is assumed that such a transformation has been done. The relays are not perfect and the hysteresis approximation in Figure 2.4 is used.

For two-relay systems there are four different relay states and four vectorfields. Define the possible switching sets as

$$\begin{aligned}
&\mathcal{S}_1 := \{x : x_1 = 0\} \quad \mathcal{S}_2 := \{x : x_2 = 0\} \quad \mathcal{S}_{12} := \{x : x_1 = 0, x_2 = 0\}\\
&\mathcal{S}_1^- := \{x : x_1 = 0, x_2 < 0\} \quad \mathcal{S}_2^- := \{x : x_2 = 0, x_1 < 0\}\\
&\mathcal{S}_1^+ := \{x : x_1 = 0, x_2 > 0\} \quad \mathcal{S}_2^+ := \{x : x_2 = 0, x_1 > 0\}. \qquad (2.34)
\end{aligned}$$

## Stable configurations

For stable vectorfield patterns in the neighborhood of $\mathcal{S}_{12}$, see Figure 2.9 to Figure 2.11. The notation 'r' and 's' indicates that the sizes and directions of the vectorfields are such that the resulting motion is either rotational or sliding. The automata indicate the different modes. There are four basic

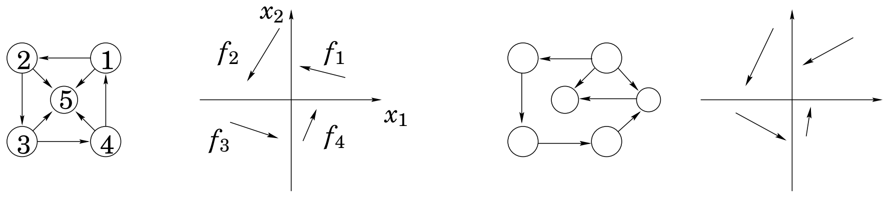

**Figure 2.9** Type rrrr and type rrrs

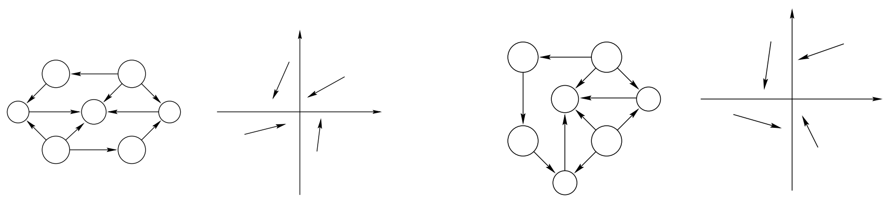

**Figure 2.10** Type rsrs and type rrss

modes, one for each quadrant. If more than one vectorfield is active, such, as in the sliding mode case, then new modes are added to the automaton. There are six vectorfield patterns for which $\mathcal{S}_{12}$ can be a locally stable subspace. All six cases thus contain a mode in the middle corresponding to the subspace $\mathcal{S}_{12}$. On $\mathcal{S}_{12}$ all four vectorfields are active and a linear combination of them defines the dynamics there.

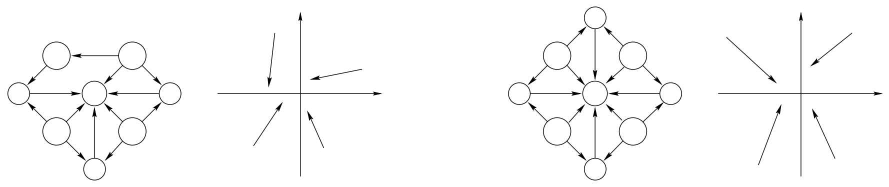

**Figure 2.11** Type rsss and type ssss

## Classification matrices

To facilitate the analysis some matrices describing the problem will be used, see Seidman (1992). Introduce the matrix $V$ containing the following four vectorfields

$$V = [\, f_1 \quad f_2 \quad f_3 \quad f_4 \,] = \begin{bmatrix} f_{11} & f_{21} & f_{31} & f_{41} \\ f_{12} & f_{22} & f_{32} & f_{42} \\ f_{13} & f_{23} & f_{33} & f_{43} \end{bmatrix}, \quad V \in \mathbb{R}^{n\times 4}. \qquad (2.35)$$

The dynamics can be written $\dot{x} = V\alpha$, where $\alpha = [\alpha_1, \alpha_2, \alpha_3, \alpha_4]^T$. Outside the sliding sets one of the $\alpha_i = 1$ and the other $\alpha_i = 0$. Note that $f_{i1}, f_{i2} \in \mathbb{R}$ and that $f_{i3} \in \mathbb{R}^{n-2}$ contains the rest of the dynamics. The coefficient matrix $C$ is used for checking the sliding conditions.

$$C = \begin{bmatrix} 1 & 1 & 1 & 1 \\ f_{11} & f_{21} & f_{31} & f_{41} \\ f_{12} & f_{22} & f_{32} & f_{42} \end{bmatrix}, \quad C \in \mathbb{R}^{3\times 4}. \qquad (2.36)$$

Note that $C\alpha = [1, \dot{x}_1, \dot{x}_2]^T$ and that the equation $\mathrm{C}\alpha = [1, 0, 0]^T$ holds on the subspace $\mathcal{S}_{12}$.

## Non-unique sliding dynamics

Sliding on the subspace $\mathcal{S}_{12}$ requires

$$C\alpha = [\,1 \quad 0 \quad 0\,]^T. \qquad (2.37)$$

Typically $C$ has rank 3 and the null space $\alpha^N$ is defined by any nonzero solution $\alpha^N$ to

$$C\alpha^N = [\,0 \quad 0 \quad 0\,]^T. \qquad (2.38)$$

The solutions, $\alpha$, to Equation 2.37 defining, $\dot{x} = V\alpha$, lie on a line segment in $\mathbb{R}^4$ and can be written

$$\alpha = \alpha^0 + \alpha^N p, \quad \alpha_i \geq 0, \qquad (2.39)$$

where $\alpha^0$ is any solution to Equation 2.37 and $p$ is a scalar. The dynamics on the subspace $\mathcal{S}_{12}$ can depend on the choice of $p$ and are generally not unique. Linear programming can be used to find specific properties such as

$$\begin{aligned} \max_p \quad & \{V(\alpha^0 + \alpha^N p)\}_i, \\ & \alpha^0 + \alpha^N p \geq 0, \end{aligned} \tag{2.40}$$

i.e. the maximum velocity in the $x_i$ direction.

## Unique sliding dynamics

There are three situations where the dynamics can be uniquely defined on a sliding set: if only two vectorfields are involved, if the dynamics are affine in the relay output signals or if the relays work on different time scales.

***Only two vectorfields involved*** In the case when the sliding motion involves only two modes the new dynamics can be uniquely determined. For example, sliding on $\mathcal{S}_2^+$ involves only vectorfields $f_1$ and $f_4$ and the sliding conditions are

$$\begin{aligned} C\alpha &= [1 \quad \dot{x}_1 \quad 0]^T \\ \alpha &= [\alpha_1 \quad 0 \quad 0 \quad \alpha_4]^T. \end{aligned} \tag{2.41}$$

These equations give unique sliding dynamics in $\dot{x} = V\alpha$. This is the Filippov convex definition as relays with hysteresis are used.

***Dynamics affine in relay outputs*** For special cases of $V$ the minimum and the maximum coincide. This happens e.g. when $f(x,u)$ is affine in $u$.

LEMMA 2.3—UNIQUE SLIDING VELOCITY
A unique sliding velocity is obtained if the vectorfields $f_1$, $f_2$, $f_3$ and $f_4$ can be written as $f_i = f^0 + f^1 u_1 + f^2 u_2$, where $u_1$ and $u_2$ take the value $-1$ or 1.

***Proof*** Any solution will have dynamics $\dot{x}$ that can be written as

$$\begin{aligned} \dot{x} &= \sum_{i=1}^{i=4} \alpha_i f_i = \alpha_1 f_1 + \alpha_2 f_2 + \alpha_3 f_3 + \alpha_4 f_4 \\ &= f^0 + f^1((\alpha_1 - \alpha_2 - \alpha_3 + \alpha_4) + f^2(\alpha_1 + \alpha_2 - \alpha_3 - \alpha_4) \\ &= f^0 + [f^1 \quad f^2] \begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \end{bmatrix} \alpha = [0 \quad 0 \quad \dot{x}_3]^T. \end{aligned} \tag{2.42}$$

For stable systems the sign conditions on the vectorfields give that the rank of the left hand side is 2 and the dynamics will thus be unique. The different solutions $\alpha$ can written as

$$\alpha = \alpha^0 + \begin{bmatrix} 1 & -1 \\ 1 & 1 \\ 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix}. \tag{2.43}$$

□

***Relays with hysteresis*** The third possibility to define unique sliding dynamics is to use relays with hysteresis. The relays can have hysteresis in either space or time. Space hysteresis mean that the input has to exceed a certain threshold level $\varepsilon$ before the relay switches. Time hysteresis means that the relay switches a certain time after passing $x_i = 0$. If the relays work on different time scales i.e. one relay is much faster than the other, then it is possible to derive unique solutions to all the six cases in figures 2.9 to 2.11. That relay one is much faster than relay two means that $\varepsilon_1, \varepsilon_2 \to 0$ in such a way that

$$\lim_{\varepsilon_1,\varepsilon_2 \to 0} \frac{\varepsilon_1}{\varepsilon_2} = 0, \tag{2.44}$$

and where relay one is the relay controlling the switch over $x_1 = 0$.

For all six cases it holds that the subspace $\mathcal{S}_{12}$ is reached in finite time. This is proven for the case *rrrr* where it is not so obvious. That the constant vectorfields requirement can be relaxed to nonlinear vectorfields is proved in the *ssss* case. In the following sections the limiting sliding dynamics are calculated for all six cases. The first case, *rrrr*, is analyzed both with and without hysteresis.

## The case *rrrr*

Let the four constant vectorfields in $V$,

$$V = \begin{bmatrix} f_{11} & f_{21} & f_{31} & f_{41} \\ f_{12} & f_{23} & f_{32} & f_{42} \\ f_{13} & f_{23} & f_{33} & f_{43} \end{bmatrix}, \tag{2.45}$$

be oriented as in Figure 2.12. If $0 < \phi_i < \pi/2$, $i = 1 \dots 4$, then there is a cyclic motion in the $x_1$-$x_2$-plane. Outside the set

$$\mathcal{S}_{12} := \{x : x_1 = 0, x_2 = 0\}, \tag{2.46}$$

the vectorfields are used in the fixed sequence, $f_1$–$f_2$–$f_3$–$f_4$ and the solution to $\dot{x} = V\alpha$ is well defined and depends only on the initial conditions.

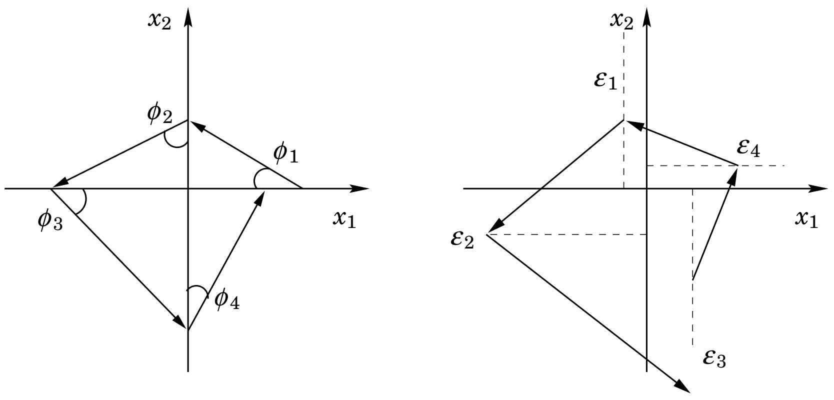

**Figure 2.12** Trajectory of a stable cyclic system. The stability condition is $\Pi_i \tan(\phi_i) < 1$. Without hysteresis (left) and with hysteresis (right).

***Relays without hysteresis*** To begin with, a system where the switching is implemented with perfect, infinitely fast, relays is studied. The starting point is $x(0) = [x_1(0), 0; x_3(0)]^T$. After one loop the new coordinates are $x(T_1) = [x_1(T_1), 0; x_3(T_1)]^T$, where

$$x_1(T_1) = \tan\phi_4 \tan\phi_3 \tan\phi_2 \tan\phi_1 \cdot x_1(0), \tag{2.47}$$

hence the following lemma.

LEMMA 2.4—LOCAL STABILITY OF $\mathcal{S}_{12}$
The subspace $\mathcal{S}_{12}$ is locally stable for perfect relay system of the form in Figure 2.12 if

$$\tan\phi_1 \tan\phi_2 \tan\phi_3 \tan\phi_4 < 1. \tag{2.48}$$

***Proof*** Sample at each time that the positive $x_1$-axis is crossed and denote the samples $x_1(k)$. Then

$$\begin{aligned} x_1(k+1) &= \lambda x_1(k) \\ \lambda &= \tan\phi_1 \tan\phi_2 \tan\phi_3 \tan\phi_4 \end{aligned} \tag{2.49}$$

with the solution

$$\begin{aligned} x_1(k) &= \lambda^k x_1(0) \\ \lim_{k\to\infty} x_1(k) &= 0, \quad |\lambda| < 1. \end{aligned} \tag{2.50}$$

□

Approaching the set $\mathcal{S}_{12}$, switching will be faster and faster. Finally, after a finite time $t^\dagger$ the trajectory will reach $\mathcal{S}_{12}$. The time to reach the set $\mathcal{S}_{12}$ is given by

$$t^\dagger = \lim_{k\to\infty} t_k = \sum_{j=1}^{\infty} \lambda^{j-1} T_1 = \frac{1}{1-\lambda} T_1, \tag{2.51}$$

where $T_1$ is the first lap time. After $t = t^\dagger$ the solution is no well-defined and today's analysis and simulation tools fail.

The limiting average dynamics depends on the initial conditions and are not unique. The limiting average dynamics could be calculated from the relative time spent in each quadrant.

LEMMA 2.5—AVERAGE DYNAMICS
The relative time spent in each quadrant is constant for every lap. From this follows that the average velocity is constant.

***Proof*** With constant vectorfields the following equations relate the points where new quadrant is entered each loop and the time spent in each quadrant, $x \notin \mathcal{S}_{12}$,

$$\begin{aligned} x_i(k+1) &= \lambda \cdot x_i(k), \quad i = 1\ldots 4 \\ t_i(k+1) &= \lambda \cdot t_i(k), \quad i = 1\ldots 4. \end{aligned} \tag{2.52}$$

The $\alpha_i$ are constant

$$\alpha_i(k+1) = \frac{t_i(k+1)}{\sum_{i=1}^4 t_i(k+1)} = \frac{\lambda \cdot t_i(k)}{\sum_{i=1}^4 \lambda \cdot t_i(k)} = \alpha_i(k), \tag{2.53}$$

and the average dynamics are

$$\dot{x}(t) = \sum_{i=1}^{4} \alpha_i f_i = \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^4 (\alpha_i f_i)_3 \end{bmatrix}. \tag{2.54}$$

□

However, the values of the $\alpha_i$ depend on the initial conditions. This means that this approach does not lead to a meaningful definition of the dynamics on the subspace $\mathcal{S}_{12}$.

***Relays with hysteresis*** If hysteresis is introduced, see Fig. 2.12 (right), the solution will tend to a unique limit cycle. One natural attempt to define the dynamics on $\mathcal{S}_{12}$ would be to introduce hysteresis, compute the average dynamics in the limit cycle and finally let the hysteresis tend to zero. Assume now that the relays do not switch until the next quadrant is penetrated a certain distance, see Figure 2.12 (right).

LEMMA 2.6—HYSTERESIS DOES NOT AFFECT STABILITY
The set $\mathcal{S}_{12}$ is locally stable for the system with hysteresis if $\mathcal{S}_{12}$ is locally stable for the system without hysteresis. Furthermore the trajectory converges to a limit cycle.

***Proof*** Sample when the $\varepsilon_4$ line is crossed and define $x_1(k)$ as the $x_1$ coordinate for the crossing. The coordinates for the crossings can be written as iterative functions, e.g. $x_1(k+1) = f(x_1(k))$, where $f(x_1) =$

$$\begin{aligned}&((((x_1+\varepsilon_1)\tan\phi_1+\varepsilon_4+\varepsilon_2)\tan\phi_2+\varepsilon_1)\tan\phi_3+\varepsilon_2+\varepsilon_4)\tan\phi_4\\ &=\lambda x_1+\zeta,\end{aligned} \tag{2.55}$$

hence the iteration is stable and

$$\lim_{k\to\infty} x_1(k) = \bar{x}_1 = \frac{\zeta}{1-\lambda} = \bar{\zeta}. \tag{2.56}$$

□

One of the switching points is $[\bar{x}_1, \varepsilon_4; x_3(k)]$. From this the other switching points can be calculated and also how much time that is spent in each domain. This gives the $\alpha_i$ and the average dynamics for $\dot{x}_3$,

$$\dot{x} = \sum_{i=1}^{4} \alpha_i(\varepsilon, f_1, f_2, f_3, f_4) f_i. \tag{2.57}$$

However, the limit

$$\lim_{\varepsilon\to 0} V\alpha(\varepsilon), \tag{2.58}$$

is not unique and will in general depend on the relative size of the $\varepsilon_i$.

***Unique solutions for $\alpha$ and $\dot{x}_3$*** In the previous section it was shown that in general $\alpha$ and $\dot{x}_3$ depend on the relative sizes of the $\varepsilon_i$. Only the cases where $\varepsilon_1 = \varepsilon_3$ and $\varepsilon_2 = \varepsilon_4$ are considered in the rest of this section. If relay one is much faster than relay two then the limit cycle has one of its vertex at $(0, \gamma\varepsilon_2)$, see Figure 2.13 (left). Taking one lap in the limit cycle gives $\gamma$ as the solution to

$$\gamma\varepsilon_2 = ((1+\gamma)\varepsilon_2 \tan\phi_2 \tan\phi_3 + 2\varepsilon_2)\tan\phi_4 \tan\phi_1$$
$$\gamma = \frac{\tan\phi_1 \tan\phi_2 \tan\phi_3 \tan\phi_4 + 2\tan\phi_1 \tan\phi_4}{1 - \tan\phi_1 \tan\phi_2 \tan\phi_3 \tan\phi_4}. \tag{2.59}$$

Knowing one vertex it is easy to calculate the others and the time spent in each quadrant. Define $\delta t_i$ as the time interval vectorfield $i$ is used. The total lap time is $\sum_{i=1}^{4} \delta t_i$ and the $\alpha_i$ are $\frac{\delta t_i}{\sum_{i=1}^{4} \delta t_i}$. The dynamics in $\dot{x} = V\alpha$ are now unique. Making relay two faster just results in a permutation of the indices, see Figure 2.13 (right).

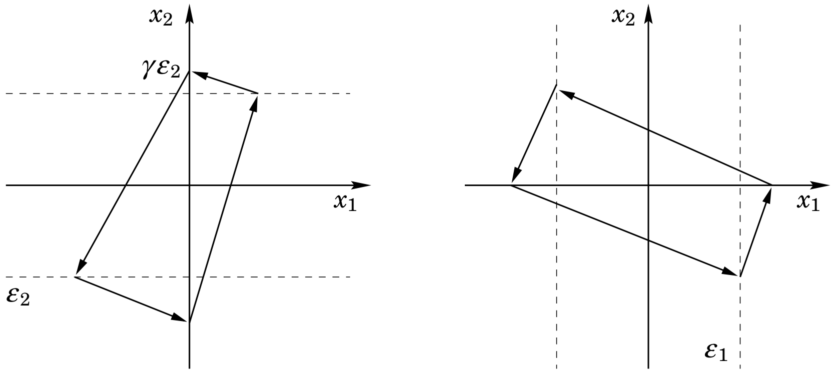

**Figure 2.13** Motion close to the subspace $\mathcal{S}_{12}$ for the case *rrrr*. Faster relay one (left) and faster relay two (right).

## The case *ssss*

In this case there can be sliding on all four half-lines $\mathcal{S}_1^{\pm}$ and $\mathcal{S}_2^{\pm}$. Three simulation sets illustrate the importance of knowing the relative speeds of the relays.

***Simulation with different relay speeds*** To get a feeling for the impact of relative speeds of the relays a numerical example is studied. Three simulation set are done. One with a faster relay over the $x_1 = 0$ axis, one with a faster relay over the $x_2 = 0$ axis and one with two relays with equal speed. The vectorfield matrix

$$V = \begin{bmatrix} -2 & 1 & 0.3 & -0.1 \\ -1 & -1 & 0.7 & 10 \\ 1 & 2 & 3 & 4 \end{bmatrix}, \tag{2.60}$$

is used in the first two simulation sets. Space hysteresis is used and the hysteresis constants are $\varepsilon_1$ and $\varepsilon_2$ for relay one and relay two respectively.

***Simulation set 1, small*** $\varepsilon_2$ In this first simulation set, $\varepsilon_1$ is kept constant and $\varepsilon_2$ is decreased. For each value of $\varepsilon_2$ the average velocity $\dot{x}_3$ is measured, see Table 2.1 for observed velocities.

| $\varepsilon_1$ | 1 | 1 | 1 | 1 |
|---|---|---|---|---|
| $\varepsilon_2$ | 1 | 0.5 | 0.1 | 0.05 |
| $\dot{x}_3$ | 2.34444 | 2.29643 | 2.26792 | 2.26784 |

**Table 2.1** Results of simulation set 1, decreasing $\varepsilon_2$.

***Simulation set 2, small*** $\varepsilon_1$ Same principle as in simulation set 1 but this time $\varepsilon_1$ is decreased. Table 2.2 shows the observed velocities. Note

| $\varepsilon_1$ | 1 | 0.5 | 0.1 | 0.05 |
|---|---|---|---|---|
| $\varepsilon_2$ | 1 | 1 | 1 | 1 |
| $\dot{x}_3$ | 2.34444 | 2.26537 | 1.98814 | 1.92444 |

**Table 2.2** Results of simulation set 2, decreasing $\varepsilon_1$.

that the limiting velocities are not the same in the two simulation sets.

***Simulation set 3, small*** $\varepsilon_1$ ***and*** $\varepsilon_2$ If both $\varepsilon_1$ and $\varepsilon_2$ are decreased at the same rate then the velocity in the $x_3$ direction depends only on the initial conditions. The matrix $V$ is in this simulation set was

$$V = \begin{bmatrix} -1 & 1 & 1 & -1 \\ -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}. \tag{2.61}$$

Starting at the point $x = [\delta, \delta, x_3(0)]$ results in the sliding dynamics $\dot{x}(3) = 1$. Starting at the point $x = [\delta, -\delta, x_3(0)]$ gives the sliding dynamics $\dot{x}(3) = -1$. Other initial conditions give something in between.

***Interpretation of the simulation results*** The simulation results can be motivated by using Filippov convex solutions for the dynamics in the $x_3$ direction.

***Simulation set 1*** In the first simulation set $\varepsilon_1$ was fixed and the size of $\varepsilon_2$ was decreased. The velocity seems to converge to something between 2.26 and 2.27. This velocity is the one obtained if the Filippov convex definition is used to calculate the sliding velocity on the manifold $x_2 = 0$. Then the Filippov convex definition is again used to calculate the velocity on the manifold $x_1 = 0$. In equations this looks like

$$\begin{aligned}
[0 \quad 1 \quad 0]\,((1-\alpha_{14})f_1 + \alpha_{14}f_4) &= 0 \\
[0 \quad 1 \quad 0]\,((1-\alpha_{23})f_2 + \alpha_{23}f_3) &= 0 \\
(1-\alpha_{14})f_1 + \alpha_{14}f_4 &= f_{14} \\
(1-\alpha_{23})f_2 + \alpha_{23}f_3 &= f_{23} \\
[1 \quad 0 \quad 0]\,((1-\alpha_{1423})f_{14} + \alpha_{1423}f_{23}) &= 0 \\
\alpha_{1423}f_{14} + (1-\alpha_{1423})f_{23} &= F^C_{1423}.
\end{aligned} \tag{2.62}$$

Calculations result in the sliding velocity $F^C_{1423,3} = \dot{x}_3 = 2.2679$, which is consistent with the simulations.

***Simulation set 2*** In this simulation set $\varepsilon_2$ is fixed and the size of $\varepsilon_1$ is decreased. The velocity seems to converge to somewhere between 1.90 and 1.95. This is the velocity obtained if the Filippov convex definitions is used but in the other order, first over $x_1 = 0$ and then over $x_2 = 0$. The corresponding equations are

$$\begin{aligned}
[0 \quad 1 \quad 0]\,((1-\alpha_{12})f_1 + \alpha_{12}f_2) &= 0 \\
[0 \quad 1 \quad 0]\,((1-\alpha_{34})f_3 + \alpha_{34}f_4) &= 0 \\
(1-\alpha_{12})f_1 + \alpha_{12}f_2 &= f_{12} \\
(1-\alpha_{34})f_3 + \alpha_{34}f_4 &= f_{34} \\
[1 \quad 0 \quad 0]\,((1-\alpha_{1234})f_{12} + \alpha_{1234}f_{34}) &= 0 \\
\alpha_{1234}f_{12} + (1-\alpha_{1234})f_{34} &= F^C_{1234}.
\end{aligned} \tag{2.63}$$

Solving for $F^C_{1234}$, gives $F^C_{1234,3} = \dot{x}_3 = 1.9068$. Again this is consistent with the simulations.

***Conclusions from simulations sets 1–3*** It was seen that the sliding velocity becomes well defined if the relative size $\varepsilon_i/\varepsilon_j$ is small. As for simulation set 3, the resulting sliding velocity depends on the initial conditions.

The results of this example suggest the following theorem where the requirement that the vectorfields should be constant is relaxed.

THEOREM 2.3—THE CHATTERBOX THEOREM
Assume that the vectorfield pattern is as in the case *ssss*, see Figure 2.11. If relay two is faster than relay one i.e.

$$\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_2}{\varepsilon_1} = 0, \tag{2.64}$$

then the limiting dynamics as $\varepsilon_1, \varepsilon_2 \to 0$ are $\dot{x} = F^C_{1423}$. This is the dynamics given by taking Filippov convex solutions first over $x_2 = 0$ and then over $x_1 = 0$. □

***Proof*** See Appendix B.

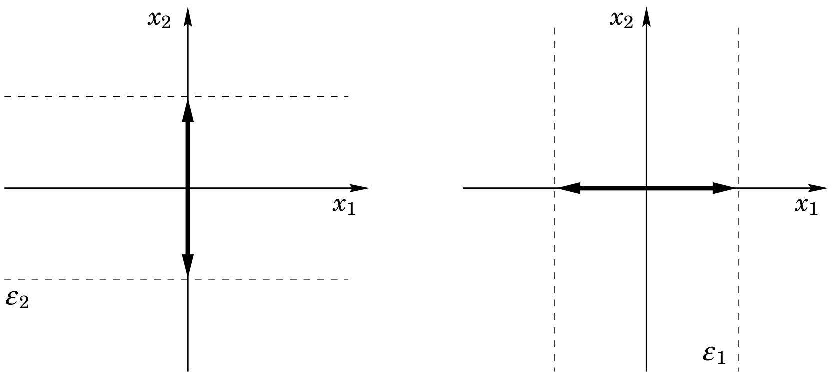

**Figure 2.14** Motion close to the subspace $\mathcal{S}_{12}$ for the case *ssss*. Faster relay one (left) and faster relay two (right).

To be consistent with rest of the six cases equations 2.62 and 2.63 are rewritten here using another notation.

***Relay one faster:*** $\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_1}{\varepsilon_2} = 0$ Two balance equations for the $x_2$ direction are used, one on $\mathcal{S}_2^+$ and one on $\mathcal{S}_2^-$,

$$\begin{aligned} f_{12}\alpha_1 + f_{22}\alpha_2 &= 0 \\ f_{32}\alpha_3 + f_{42}\alpha_4 &= 0. \end{aligned} \tag{2.65}$$

Those two equations replace one single conditions on $\dot{x}_2$ in the $C\alpha = 0$ constraint.

***Relay two faster:*** $\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_2}{\varepsilon_1} = 0$ Now there are two equations for the $x_1$ direction instead, one on $\mathcal{S}_1^+$ and one on $\mathcal{S}_1^-$,

$$\begin{aligned} f_{11}\alpha_1 + f_{41}\alpha_4 &= 0 \\ f_{21}\alpha_2 + f_{31}\alpha_3 &= 0. \end{aligned} \tag{2.66}$$

## The case *rrrs*

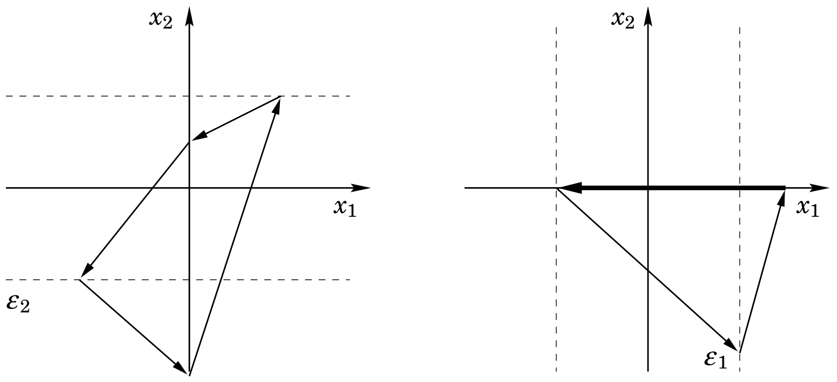

**Figure 2.15** Motion close to the subspace $\mathcal{S}_{12}$ for the case *rrrs*. Faster relay one (left) and faster relay two (right).

For motion close to the subspace $\mathcal{S}_{12}$, see Figure 2.15. A sliding motion can possibly occur on $\mathcal{S}_2^+$. Depending on which relay that is the fastest two different situations occur.

***Relay one faster:*** $\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_1}{\varepsilon_2} = 0$ If relay one is the fastest then the same solution as for the *rrrr* case is obtained. Find one vertex at $(0, \gamma\varepsilon_2)$. This gives $\alpha_i$ and $\dot{x} = V\alpha$ on $\mathcal{S}_{12}$. See Figure 2.15 (left).

***Relay two faster:*** $\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_2}{\varepsilon_1} = 0$ Now the vectorfield $f_2$ is used less and less as $\varepsilon_2$ becomes smaller and smaller. The $\alpha$ is found by adding the equation $\alpha_2 = 0$ to the conditions on $C\alpha$.

$$\begin{aligned} C\alpha &= [1 \quad 0 \quad 0]^T \\ [0 \quad 1 \quad 0 \quad 0]\alpha &= 0. \end{aligned} \tag{2.67}$$

The sliding motion is indicated with a thicker line, Figure 2.15 (right).

## The case *rsrs*

With this vectorfield pattern there is a possibility for sliding both on $\mathcal{S}_2^+$ and $\mathcal{S}_2^-$.

***Relay one faster:*** $\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_1}{\varepsilon_2} = 0$ A faster relay one again leads back to the *rrrr* solution with one vertex on $(0, \gamma\varepsilon_2)$. See Figure 2.16 (left).

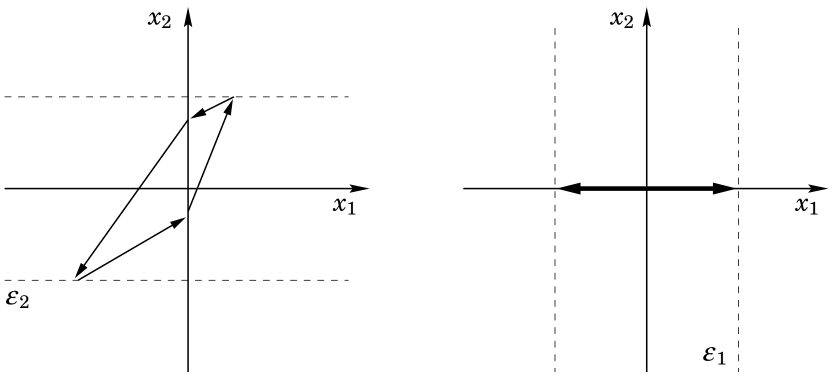

**Figure 2.16** Motion close to the subspace $\mathcal{S}_{12}$ for the case *rsrs*. Faster relay one (left) and faster relay two (right).

***Relay two faster:*** $\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_2}{\varepsilon_1} = 0$ This lead to sliding on $\mathcal{S}_2^{\pm}$. The balance equation in the $x_2$ direction, third row in $C\alpha = 0$, is now replaced with two equations. One for $\mathcal{S}_2^{+}$ and one for $\mathcal{S}_2^{-}$,

$$\begin{aligned} f_{12}\alpha_1 + f_{42}\alpha_4 &= 0 \\ f_{22}\alpha_2 + f_{32}\alpha_3 &= 0, \end{aligned} \tag{2.68}$$

where the $f_{i2}$ are the components in the $x_2$ direction, see Figure 2.16 (right). Equation 2.68 together with the remaining equations in $C\alpha = 0$ now give a unique solution for $\alpha$ and thus for $\dot{x} = V\alpha$.

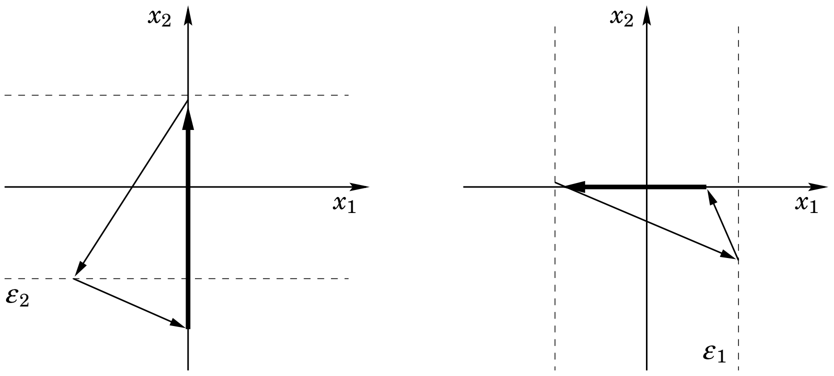

**Figure 2.17** Motion close to the subspace $\mathcal{S}_{12}$ for the case *rrss*. Faster relay one (left) and faster relay two (right).

### The case *rrss*

Now there is possible sliding on $\mathcal{S}_1^-$ and $\mathcal{S}_2^+$. The relative speeds of the relays decide which vectorfield that is not used.

***Relay one faster:*** $\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_1}{\varepsilon_2} = 0$ Vectorfield $f_1$ never comes into use. The equation $\alpha_1 = 0$ is an additional constraint, which together with the $C\alpha = 0$ condition give a unique velocity $\dot{x} = V\alpha$. See Figure 2.17 (left).

***Relay two faster:*** $\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_2}{\varepsilon_1} = 0$ Now it is vectorfield $f_2$ that is not used. The additional constraint needed to get unique sliding dynamics is thus $\alpha_2 = 0$. See Figure 2.17 (right).

### The case *rsss*

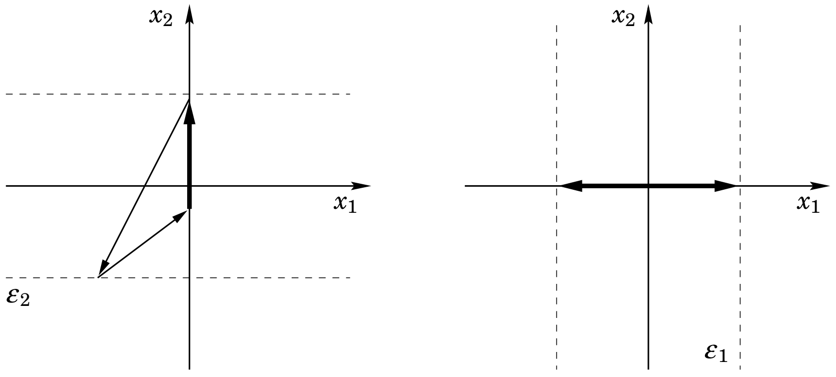

**Figure 2.18** Motion close to the subspace $\mathcal{S}_{12}$ for the case *rsss*. Faster relay one (left). Faster relay two (right).

In the last of the six cases the possible sliding sets are $\mathcal{S}_2^-$, $\mathcal{S}_2^+$ and $\mathcal{S}_1^-$. As usual whether there is sliding or not depends on the relative speeds of the relays.

***Relay one faster:*** $\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_1}{\varepsilon_2} = 0$ Vectorfield $f_1$ is never used and thus $\alpha_1 = 0$. Sliding occurs on $\mathcal{S}_1^-$. See Figure 2.18 (left).

***Relay two faster:*** $\lim_{\varepsilon_1,\varepsilon_2\to 0} \frac{\varepsilon_2}{\varepsilon_1} = 0$ Sliding is possible both on $\mathcal{S}_2^-$ and $\mathcal{S}_2^+$. Now there are two equations for the $x_2$ direction. One for sliding on $\mathcal{S}_2^-$ and another for sliding on $\mathcal{S}_2^+$,

$$\begin{aligned} f_{12}\alpha_1 + f_{42}\alpha_4 &= 0 \\ f_{22}\alpha_2 + f_{32}\alpha_3 &= 0. \end{aligned}$$

In total there are four equations and thus unique sliding dynamics.

## 2.5 Summary

This chapter treats the problem of sliding or fast mode changes. The theory for first order sliding on one sliding surface is well-known but higher order sliding and sliding on multiple sliding surfaces are not so often treated in the control literature.

It is shown that higher order sliding is not an unusual phenomena and a theorem for stability and the dynamics of second orders sliding is presented. Is is further shown that for a specific system class there can be no sliding of degree three or higher.

Sliding on multiple sliding surfaces leads in general to ambiguous dynamics. If the difference in relative switching speeds for two switching surfaces is large then it is possible to define unique dynamics.

# 3

# Simulation of hybrid systems

## 3.1 Introduction

Today, most investigations of hybrid systems are done by simulation. Even though much research effort is put into the the field of hybrid systems theory, this situation is not likely to change in the near future. To support theory and help the designer of control systems the single most important tool is, and will be, a good simulation tool. There has always been a strong link between the control and simulation communities. On one hand simulation is an extremely important tool for almost every control engineer. On the other hand, simulation of control systems often pushes the limits of what simulation packages can do. This is because of the nature of control systems. The controlled plant could be virtually anything described by a set of equations, inequalities, difference equations and differential equations.

In addition to the normal requirements, a simulation tool for hybrid systems must be able to detect and initialize discrete events. At discrete instances the continuous states can be reset or the dynamics definitions can be changed. There are several simulation packages claiming that they allow for the mixture of continuous variables and discrete variables and that they thus are suitable for simulating hybrid systems, see Otter and Cellier (1995). Few of them are, however, strong general purpose simulation tools. Typically, they are customized to a specific problem domain and have difficulties modeling systems from other domains. This is a serious drawback since systems are becoming more complex and are often composed of parts from many different domains. A manufacturing company using several subsystems might require that the components come with a simulation model. To be able to interconnect the models it is then required that they are encoded using the same simulation language or that the simulation tool is capable of handling several descriptions.

Other hybrid simulators consist of fixes applied to continuous time simulation packages that were not originally built to simulate hybrid systems. Currently, simulation performance for general hybrid systems is poor and unpredictable.

### Model deficiencies

It is not always the case that a mathematical model can be simulated as it stands. Even just entering a model into a simulation tool is yet one more layer of approximation. This should be added to the approximations already done when doing a mathematical model of a physical system and later to the approximations done during implementation of the control system. Especially important for hybrid systems is the approximation of the transitions between the discrete modes. The modeling of switches can drastically change the result, both in simulation and for the final implementation. Most users just plug in their models and study the result. This method does not work for hybrid systems. There is often no way for a user to know if the simulation results are to be trusted. Many simulation results are today wrong by definition. If there is no well-defined solution how could the result be right?

An important issue for hybrid systems is simulation of so called sliding mode behavior, i.e. when there might be infinite many mode switches in a finite time interval. In this chapter the models are hybrid systems consisting of continuous time differential equations and relays. The ideal relay yields ambiguous solutions to discontinuous differential equations and simulation results are hence difficult or impossible to evaluate. The relay is a natural way to model discrete switches between otherwise continuous systems. It can, for example, model switches between different vectorfields, parameter sets, or controllers.

### Fast mode changes

It is a non-trivial problem to detect a possibility for fast mode changes in certain variables for certain parameter values and certain initial conditions by just looking at the equations. Thus it is difficult for the user to know that the model is not good enough. On the other hand it is non-trivial for the simulation program to resolve the problem when it has detected fast mode changes. Today, the users are warned about algebraic loops and inconsistencies in the model. It is not considerable overhead to also check for possible sliding modes.

The problem could be solved by a semi-automatic procedure. The simulation package detects the problem and prompts the user for additional modeling information. The most important feature is of course that the simulation program notices that there is a problem. This can be done during compilation of the models or, if it is parameter dependent, during simulation. To further aid the user, the tool should tell what variables are involved in such fast cycles and suggest new induced modes. The construction of a suitable model for simulations will typically be an iterative process with interaction between the user and the simulation tool.

Some ideas of how under-modeling of sliding modes can be detected is presented in Section 3.4.

## Implementation

Most of the discussion in this chapter is about simulation, but of course similar problems are present when it comes to implementation of the hybrid control systems. In real-time implementation, as in simulation, there is no ideal switching between the controllers. If the discrete transitions are implemented with relays they can often be approximated with hysteresis functions in the analysis. If the implementation is done with transistors a better approximation might be a to approximate the relay with a smooth function. Also note that nonzero sampling times, quantization effects and unmodeled dynamics in actuators and measuring devices introduce hysteresis-like behavior. It is important to have detailed knowledge about the actual implementation to be able to calculate the sliding dynamics. Understanding these concepts is of course very important for the actual implementor of hybrid systems.

EXAMPLE 3.1—QUANTIZATION EFFECTS CAN MAKE SMOOTH FUNCTIONS LOOK LIKE RELAYS

Consider the following system, which is a variant of the system in Example 2.1 discussed in Chapter 2.

$$\begin{aligned}\dot{x} &= (\cos(u) - 0.8)x \\ \dot{y} &= -\sin(u) \\ u &= \frac{2}{\pi}\arctan((y_{ref} - y)/\varepsilon). \end{aligned} \tag{3.1}$$

With limited resolution in the measurement of $y_{ref} - y$ this system is unstable if the gain $\frac{1}{\varepsilon}$ is low. With a higher gain the system becomes stable. This corresponds to choosing the Filippov convex definition or the Utkin equivalent control definition for the sliding dynamics. □

## 3.2 Definitions of solutions to differential equations

The users of simulation programs often have little control over how the discrete states are changed. It is thus difficult to know if the result is well-defined by the local models $f_1, \ldots, f_n$ and the transitions $\sigma_{ij}$. Some simulation tools have sophisticated event handlers but this is not good enough to detect model deficiencies. Important issues are if the model is detailed enough in terms of

- number of local models $f_i$
- well-defined transitions $\sigma_{ij}$ between the models

These problems are related to the definition of solutions to differential equations. Depending on the definition of a solution the simulation result could be incorrect or correct. Possible definitions of solutions to the differential equation $\dot{x} = f(x,t)$ could be

1 A function $x(t)$ such that $\frac{dx}{dt} = f(x,t)$, almost everywhere. This includes continuous and discontinuous vectorfields but not chattering.

2 A more general definition such as the one used in Filippov (1988), Clarke (1990) and Paden and Sastry (1987). There, $x(t)$ is called a solution to $\dot{x} = f(x,t)$ if $\dot{x}(t) \in K[f](x), a.e.$, where

$$K[f](x) \overset{\Delta}{=} \cap_{\delta>0} \cap_{\mu N=0} \overline{co}\{f(B(x,\delta) - N, t)\} \tag{3.2}$$

and $\cap_{\mu N=0}$ denotes intersection over all sets $N$ of Lebesgue measure zero. $B(x,\delta)$ is a ball in $\mathbb{R}^n$ with radius $\delta$, $x \in \mathbb{R}^n$.

EXAMPLE 3.2—SLIDING MODES – TRANSITION EQUATIONS HAVE TO BE REFINED
Looking back on the examples in the previous chapter it was seen that in order to get a well-defined solution, additional modes had to be introduced. To define the dynamics of these modes, the transition equations were refined. Solution definition 1 above does not capture sliding behavior. The solution definition 2 finds the sliding mode as a solution, however, it is not unique, c.f. Filippov solutions in Chapter 2. □

EXAMPLE 3.3—A 'WRONG' DEFINITION MIGHT MISS A SOLUTION
Mattsson (1996) has an interesting example that illustrates the problem with hybrid system modeling. It is a second order example where the

control signal $u$ is a function of the coordinate $x$.

$$\frac{dx}{dt} = u^2$$
$$\frac{dy}{dt} = 1 - u^2, \quad u = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases} . \tag{3.3}$$

The given model is not enough to uniquely define a solution. If the transition is modeled with a continuous function $u_\varepsilon = u * \phi_\varepsilon$ then the system gets stuck on the line $x = 0$. If the transition is modeled with a hysteresis function the trajectories go right through the $x = 0$ axis.

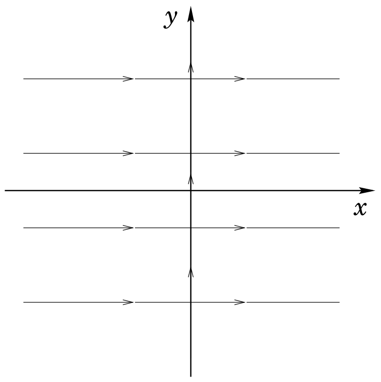

**Figure 3.1** Phase plane for Example 3.3. The vectorfield is vertical for $x = 0$ and horizontal for $x \neq 0$. The solution $x \equiv 0, \dot{y} = 1$ is not considered a solution according to definition 2.

The subspace $x = 0$ has Lebesgue measure zero and is hence disregarded in the definition 2 above, thus the dynamics $x = 0, \dot{y} = 1$ is not a solution according to that definition. □

The examples should make it clear that there really is no 'correct' result to some simulations unless more careful modeling of salient features is done.

## 3.3 The simulation problem

What can a state-of-the-art simulation tool do and what more is needed? The simulation problem can be divided into three main parts: modeling, compilation and simulation.

### Modeling

The modeling part consists of entering dynamic equations, the static equations, and connecting them. For a control engineer the natural unit to work with is the sub-system blocks of a block diagram. Good simulation tools suited for hybrid systems are object oriented and have support for building hierarchical models and graphically connecting model blocks.

Many simulation packages only allow entering of differential equations on explicit form $\dot{x} = f(x,t)$. Rewriting the physical models this way needs a modeling expert and is often hard or impossible. Further, this has to be done globally, i.e. encompass all sub-models, and does not allow for easy interconnection and reuse of models, see Andersson (1994). A basic demand from the industry seems to be that it should be possible to use models of the form $E(x,t)\frac{dx}{dt} = f(x,t)$.

### Compilation

Before actual simulation, connections between building blocks are made into equations, and the collection of equations and differential equations are put together in a differential-algebraic equation, a DAE, on the form $F(\dot{x},x,t) = 0$. A modern simulation tool does a lot of analysis of this DAE. The models are checked for mathematical completeness and consistency and structural analysis is done to see if the equations can be split into smaller sub-problems. The equations are sorted into a sequence of sub-problems and block lower triangular partition (BLT) decomposes the problem in computational order see Tarjan (1972).

In the compilation phase it is further decided if the initial conditions together with the equation $F(\dot{x},x,t) = 0$ have a solution.

### Simulation

If the differential-algebraic equation, $F(\dot{x},x,t) = 0$, can be transformed back to explicit form $\dot{x} = f(x,t)$ methods like explicit Runge-Kutta can be used. If not, implicit methods like DASSL, Brenan *et al.* (1989) or implicit Runge-Kutta are used.

***Hardware in the loop*** When testing control systems it is very useful to be able to replace parts of the simulation model with the actual hardware to be used in the real system. This requires that the simulation can be done in real-time. Another simulation problem in real-time is when humans are included in the loop, e.g. flight simulators.

### Simulation of hybrid systems

Special care has to be taken simulating hybrid systems because of the mix of continuous and discrete equations. Whatever mathematical description used in the modeling of a problem this will only be approximated by the simulation model. One example of a modeling problem is in building a simulation model from a description such as the Tavernini type hybrid systems

$$\begin{aligned} \dot{x} &= f(x(t), q(t)), \quad x \in \mathbb{R}^n \\ q(t) &= \nu(x(t), q(t^-)), \quad q \in Z^{m+}. \end{aligned} \tag{3.4}$$

Such a hybrid system is depicted as an automaton in Figure 3.2. The

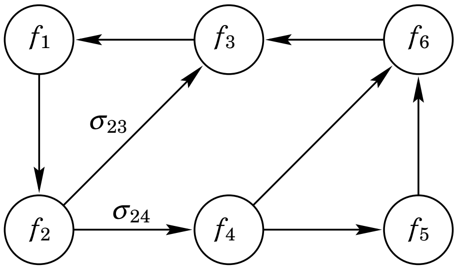

**Figure 3.2** Automaton for the model described with Equation 3.4.

transition from node $f_2$ to $f_3$ can be a model of several discrete phenomena: e.g. the vectorfield $f_i$ can change or there could be a state space jump. The simulation result could differ a lot if these changes are done continuously or abruptly i.e. how good is the simulation model approximation of the mathematical description. Another problem is if both transitions $\sigma_{23}$ and $\sigma_{24}$ are enabled simultaneously.

On the other hand it is possible that a hybrid model is derived from deliberately neglecting fast smooth transitions. In such a case the modeler does not want to explicitly specify the transitions if their definition do not alter the simulation result.

## Simulation tools

There is not a lot to choose from in terms of good general purpose simulation packages for hybrid systems. For a good review of simulation tools, see Otter and Cellier (1995). A frequently available simulation tool is SIMULINK. Two well-known object-oriented tools are Omola/OmSim and SHIFT. Recently, several simulation research groups have formed an international corporation developing the next generation simulation tool, Modelica. This project is yet in the beginning but looks very promising.

***SIMULINK*** See MATLAB (1995), has an intuitive, easy to use, interface. It is available for a broad range of computing platforms and operating systems. A drawback is that SIMULINK does not offer appropriate event handling mechanisms. This is bad for control systems in general but really critical for hybrid systems simulation. The SIMULINK providers are aware of the problem and the event handling is getting better with every new version.

***Omola/OmSim*** See Andersson (1994), is one of the general purpose tools that can be used for simulating hybrid systems. The simulation environment has features as hierarchical models, object-orientation modeling, continuous time and discrete event models. It has extensive model analysis and error checking. It uses a sophisticated mix of symbolic equation manipulation, solution approximation, zero-finding, and ODE/DAE solvers to accurately perform such mixed simulation. Omsim can handle continuous dynamics given by general DAEs, i.e. differential equations $\dot{x} = f(x, y, t)$ together with constraints $0 = g(x, y, t)$.

***SHIFT*** See Antsaklis (1997), pp 113-133, is a programming language for describing and simulating dynamic networks of hybrid automata. The functionality of SHIFT is similar to the functionality of Omola/OmSim. The main difference is that Omola/OmSim has statically interconnected objects whereas SHIFT supports evolution of interconnections. In SHIFT it is thus possible to model dynamically reconfigurable hybrid systems.The primary application of SHIFT has been automatic control of vehicles and highway systems.

***Modelica*** See Mattsson and Elmqvist (1997), is an international effort to standardize simulation tools. Modelica has a standardized functionality and standardized integration routines. The work started in the continuous time domain since there is a common mathematical framework in the form of differential-algebraic equations. The first goal was to collect knowledge and construct a unified modeling language. The short time goal is to be able to simulate DAEs with some discrete event features. The simulation environment is independent of the physical domain and should be able to model systems from different engineering fields.

## 3.4 Structural detection of fast mode changes

In a simulation environment such as Omsim, see Andersson (1994), it is possible to do structural analysis of the equations before simulation. Such a structural analysis can be achieved by efficient graph methods. Distinction is then made only between zero and nonzero coefficients. An example of such structural analysis is block-lower triangular (BLT) partition of the problem with respect to the variables. This has been used to detect algebraic loops of minimal dimension, see Tarjan (1972) and Duff *et al.* (1990). Another example is Pantelides' algorithm, see Pantelides (1988) and Mattsson and Söderlind (1993) that is used to determine suitable forms of integration of high-index DAE problems. A BLT partitioning can determine whether the output of a relay structurally influences the input to the same relay. Hence there might exist a fast sliding mode involving the dynamics between the output and the input of the relay. This can be nontrivial to see, if the system is part of a much larger problem. Since efficient graph methods that analyze dependencies exist it is negligible overhead work to check for the structural possibility of fast switching.

Before simulation it can be checked if sliding *might* occur and (minimal) sliding loops can be determined. Since structural analysis only gives necessary conditions it is not guaranteed that switching actually will occur during simulation. The structure analysis is done to determine if fast switching *may* occur for problems with one or multiple relays.

To get some insight and to illustrate the method a small example with two states and two relays is analyzed.

EXAMPLE 3.4—STRUCTURAL DETECTION OF FAST MODE CHANGES
Consider the following hybrid system on explicit form

$$\begin{aligned}
\dot{x}_1 &= f_1(x_1, u_1) \\
\dot{x}_2 &= f_2(u_2) \\
y_1 &= h_1(x_2) \\
y_2 &= h_2(x_1) \\
u_1 &= \operatorname{sgn}(y_1) \\
u_2 &= \operatorname{sgn}(y_2).
\end{aligned}$$

The actual form of the functions $f_i$ and $h_i$ does not matter in this discussion since only structural dependencies are analyzed.

***Structure Jacobian*** The structure Jacobian is built in the following way. The horizontal lines are the equations and differential equations defining relations between the variables. Each variable has a column and if a variable enters into a relation it is marked with a star on that line. After adjoining the equations $\dot{x}_i = \frac{d}{dt}x_i$ the corresponding structure Jacobian is given by

| | $\dot{x}_1$ | $x_1$ | $\dot{x}_2$ | $x_2$ | $u_1$ | $y_1$ | $u_2$ | $y_2$ |
|---|---|---|---|---|---|---|---|---|
| $f_1$ | $\star$ | $\star$ | | | $\star$ | | | |
| $f_2$ | | | $\star$ | | | | $\star$ | |
| $h_1$ | | | | $\star$ | | $\star$ | | |
| $h_2$ | | $\star$ | | | | | | $\star$ |
| $\text{sgn}_1$ | | | | | $\star$ | $\star$ | | |
| $\text{sgn}_2$ | | | | | | | $\star$ | $\star$ |
| $x_1$ | $\star$ | $\star$ | | | | | | |
| $x_2$ | | | $\star$ | $\star$ | | | | |

Fast mode switching is possible if there is a loop from the output of a relay to the input of the same relay. The possible sliding motion is confined to the subspace defined by the input equations of the relays active in the loop.

***A directed graph description*** A directed graph, see Jantzen (1994) of the system is built in the following way: Define one node for each variable $x_i$, each derivative $\dot{x}_i$, each relay input $y_i$ and each relay output $u_i$. Each equation (row in the structure Jacobian) defines directed arcs between the nodes. For example, the $f_1$ function defines arcs from $u_1$ and $x_1$ to $\dot{x}_1$, the output relation $h_1$ defines an arc from $x_2$ to $y_1$ and the $\text{sgn}_1$ function connects the relay input $y_1$ to the output $u_1$. The last equations connect $\dot{x}_i$ to $x_i$. If higher derivatives are present then there is one more node for each order of differentiation.

***Loop detection*** In this example there is a loop from the output of relay one, $u_1$, to the input of the relay, $y_1$, of the form $u_1 - \dot{x}_1 - x_1 - y_2 - u_2 - \dot{x}_2 - x_2 - y_1 - u_1$, which is easy to find using a straight forward graph algorithm. There are several graph methods for detecting loops, see Jantzen (1994). The conclusion drawn is that there is a possibility for sliding on the subspace $\{x : y_1(x) = y_2(x) = 0\}$. □

## Relation to Chapter 2

The structural detection method is now applied to some familiar examples. The examples illustrate how first order sliding, higher order sliding and multiple surface sliding can be detected.

EXAMPLE 3.5—IVHS – STRUCTURAL DETECTION OF THE SLIDING MODE.
In Chapter 2 it was seen that in this example there was a first order sliding mode.

$$\begin{aligned}\dot{x}_1 &= \cos(u) = f_1(u)\\ \dot{x}_2 &= -\sin(u) = f_2(u)\\ y &= x_2 = h(x)\\ u &= \operatorname{sgn}(y).\end{aligned}$$

The corresponding structure Jacobian is given by

| | $\dot{x}_1$ | $x_1$ | $\dot{x}_2$ | $x_2$ | $u$ | $y$ |
|---|---|---|---|---|---|---|
| $f_1$ | $\star$ | | | | $\star$ | |
| $f_2$ | | | $\star$ | | $\star$ | |
| $h$ | | | | $\star$ | | $\star$ |
| sgn | | | | | $\star$ | $\star$ |
| $x_1$ | $\star$ | $\star$ | | | | |
| $x_2$ | | | $\star$ | $\star$ | | |

There is a loop from the relay output, $u$, to the relay input, $y$ of the relay to the input of the form $u - \dot{x}_2 - x_2 - y - u$ and thus possibly sliding on the set where the relay input function is zero, here $x_2 = 0$. There is only one differentiated variable in the loop and the possible sliding is thus of order one, i.e transversal sliding. □

EXAMPLE 3.6—IVHS WITH FILTER DYNAMICS
A third equation, modeling filter dynamics, was added and second order sliding was observed for this example.

$$\begin{aligned}\dot{x}_1 &= \cos(u) = f_1(x,u)\\ \dot{x}_2 &= -\sin(u) = f_2(x,u)\\ \dot{x}_3 &= -ax_3 + x_2 = f_3(x,u)\\ y &= h(x_3)\\ u &= \operatorname{sgn}(y).\end{aligned}$$

The structure Jacobian is somewhat larger but it is still possible to do structure analysis by hand.

|  | $\dot{x}_1$ | $x_1$ | $\dot{x}_2$ | $x_2$ | $\dot{x}_3$ | $x_3$ | $u$ | $y$ |
|---|---|---|---|---|---|---|---|---|
| $f_1$ | $\star$ |  |  |  |  |  | $\star$ |  |
| $f_2$ |  |  | $\star$ |  |  |  | $\star$ |  |
| $f_3$ |  |  |  | $\star$ | $\star$ | $\star$ |  |  |
| $h$ |  |  |  |  |  | $\star$ |  | $\star$ |
| sgn |  |  |  |  |  |  | $\star$ | $\star$ |
| $x_1$ | $\star$ | $\star$ |  |  |  |  |  |  |
| $x_2$ |  |  | $\star$ | $\star$ |  |  |  |  |
| $x_3$ |  |  |  |  | $\star$ | $\star$ |  |  |

There is obviously no loop including $x_1$ and $u$. The possible sliding loop from the relay output, $u$, to the relay input, $y$, is on the form $u - \dot{x}_2 - x_2 - \dot{x}_3 - x_3 - y - u$. The loop involves only two differentiated variables and the possible sliding is here of at least second order. The sliding set is given by $\{x : x_2 = 0, x_3 = 0\}$, i.e. $y = 0$ and $\dot{y} = 0$. □

The next example illustrates that also simultaneous sliding on multiple sliding surfaces can be detected. If the sliding loops from several relays are interconnected, i.e share variables, then there can be simultaneous sliding on the intersections of the relay input sets.

EXAMPLE 3.7—TWO-RELAY SYSTEMS

This is the two-relay case from the previous chapter. In this example it was seen that there were possible sliding on two subspaces and also simultaneous sliding on their intersection. Without explicitly specifying the function the dynamics were on the form

$$\begin{aligned}
\dot{x}_1 &= f_1(u_1, u_2)\\
\dot{x}_2 &= f_2(u_1, u_2)\\
\dot{x}_3 &= f_3(u_1, u_2)\\
y_1 &= h_1(x_1)\\
y_2 &= h_2(x_2)\\
u_1 &= \mathrm{sgn}_1(y_1)\\
u_2 &= \mathrm{sgn}_2(y_2).
\end{aligned}$$

The structure Jacobians for this example is

| | $\dot{x}_1$ | $x_1$ | $\dot{x}_2$ | $x_2$ | $\dot{x}_3$ | $x_3$ | $y_1$ | $y_2$ | $u_1$ | $u_2$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $f_1$ | $\star$ | | | | | | | | $\star$ | $\star$ |
| $f_2$ | | $\star$ | | | | | | | $\star$ | $\star$ |
| $f_3$ | | | $\star$ | | | | | | $\star$ | $\star$ |
| $h_1$ | | $\star$ | | | | | $\star$ | | | |
| $h_2$ | | | | $\star$ | | | | $\star$ | | |
| $\text{sgn}_1$ | | | | | | | $\star$ | | $\star$ | |
| $\text{sgn}_2$ | | | | | | | | $\star$ | | $\star$ |
| $x_1$ | $\star$ | $\star$ | | | | | | | | |
| $x_2$ | | | $\star$ | $\star$ | | | | | | |
| $x_3$ | | | | | $\star$ | $\star$ | | | | |

There is one loop for relay one: $u_1 - \dot{x}_1 - x_1 - y_1 - u_1$. A second loop: $u_1 - \dot{x}_2 - x_2 - y_2 - u_2$ is detected for relay two. Furthermore both $u_1$ and $u_2$ appear in the $f_1$ and $f_2$ equations and hence there are cross-couplings. From this structure analysis it then follows that there are possibly five new modes needed. They correspond to the sets $\mathcal{S}_1^{\pm}$, $\mathcal{S}_2^{\pm}$ and $\mathcal{S}_{12}$. $\square$

## 3.5 New modes - new dynamics

The addition of new modes can be done semi-automatically as the simulation tool detects a loop. If the user supplies more detailed equations for the transitions, new modes can be added and the dynamics of the new modes can be calculated. The generation of necessary new modes are illustrated with an algorithm valid for the two-relay case.

ALGORITHM 3.1—MODE INDUCEMENT ALGORITHM

1 Calculate the sliding sets associated with the relays, $\mathcal{S}_1^{\pm}$ and $\mathcal{S}_2^{\pm}$.

2 Check for intersections, $\mathcal{S}_{12}$.

3 Generate induced modes for each sliding set and each intersection of sliding sets. For the two-relay case five induced modes are generated.

4 Remove modes that are not attractive, i.e. have vectorfields pointing away from the sliding set associated with the mode.

In general, it is difficult to do point 4 in the algorithm above before simulation. Further, the investigation is not necessary at all if the initial conditions are such that the trajectory never enters a sliding set. Perhaps a better way is to generate new modes as they are needed during simulation as illustrated in the next example.

EXAMPLE 3.8
To illustrate this method consider the two-relay problem with a vectorfield pattern as in Figure 2.9. On the set $\mathcal{S}_2^+ = \{x : x_2 = 0, x_1 > 0\}$ there is a sliding mode not originally modeled. On that set the solution is not well-defined. Fig. 3.3 shows the mode addition procedure step by step. To begin

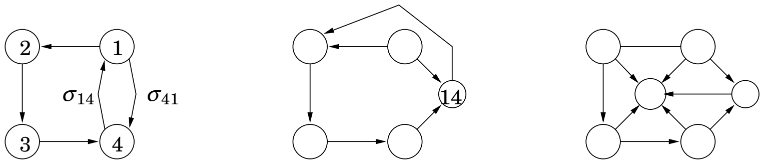

**Figure 3.3** New modes are added as they are needed.

with there are four modes in the model, one for each quadrant. Hitting the sliding set $\mathcal{S}_2^+$ the system starts to change mode very fast. Evaluating the transition equations $\sigma_{14}$ and $\sigma_{41}$ it is seen that both are enabled. A new mode denoted 14 for the set $\mathcal{S}_2^+$ together with new transition equations, $\sigma_{1,14}$, $\sigma_{4,14}$ and $\sigma_{14,2}$ are defined. This new extended model is now possible to simulate until $\mathcal{S}_{12}$ is reached. At $\mathcal{S}_{12}$ there is again fast mode changes and a new mode with new dynamics has to be introduced. □

The dynamics in the new modes can be derived from Filippov and Utkin definitions. Which one to choose depends on the additional modeling information supplied by the user. The Utkin equivalent control dynamics can be determined by substituting the relay equations, for the relays in the detected loop, with the equations $y_1(x) = y_2(x) = \dot{y}_1(x) = \dot{y}_2(x) = 0$ and solving for $u$, (if this is possible, otherwise higher derivatives of $y_1(x)$ and/or $y_2(x)$ must be computed). The resulting $u$ might be non-unique or non-existent. The Filippov convex definition is in general harder to compute. There are $2^k$ vectorfields in the equation

$$\dot{x} = \sum_{u \in \{-1,1\}^m} \alpha_u f(x,u) \tag{3.5}$$

and possibly non-unique solutions. The non-uniqueness problem can sometimes be solved by specifying relative speeds for the relays involved in the fast mode change loop.

If the simulation problem has an implicit DAE formulation then more work is required to derive a solution to the sliding dynamics.

## 3.6 Summary

This chapter illustrates some of the problems present when simulating hybrid system. The definitions of what constitutes a solution to a differential equation has to be considered in the analysis of the simulation models.

A method based on structural analysis for determining the possibility of fast mode changes was sketched. The structural detection method was applied to some examples from Chapter 2.

New modes could be added to the simulation models semi-automatically and the dynamics in the new modes can be defined using additional modeling information.

The construction of suitable models for simulation will typically be an iterative process with interaction between the user and the simulation tool.

# 4 Hybrid control system design

## 4.1 Introduction

As discussed in earlier chapters there can be several reasons to use a hybrid controller. Many hybrid controllers give very good performance but they are difficult to analyze. This chapter begins with a discussion about how Lyapunov theory can be extended to cover hybrid systems as well. An example of a natural extension that fails is presented. The main part of the chapter presents a design method for hybrid control systems that guarantees stability. The method is based on local controllers and non-smooth Lyapunov functions. Some advantages with this method are that

- It is possible to enlarge the stability region and a larger system class can be stabilized.
- Complicated control problems can be divided into local tasks.
- Local modeling and local controllers simplify design.

## 4.2 Stability

The step from one-controller systems to systems where a set of controllers can act on the process is a great step. It is a lot more complicated to prove stability for the general hybrid control system. The standard Lyapunov theory can be extended and in some cases be used to prove stability. This section contains some modifications to Lyapunov theory. Some intuitive modifications are not correct for fast mode switching systems.

## Lyapunov functions

Lyapunov's theorem for one-controller systems is recited here for convenience. The notation follows Khalil (1992). Consider the system

$$\dot{x} = f(x, u, t). \tag{4.1}$$

It is of fundamental interest to know if the system described by equation 4.1 is stable when a specific control law $u(x,t)$ is applied. Lyapunov theory a powerful tool for this. The main idea is the following: find a candidate Lyapunov function $V(x,t)$ and prove that the time derivative $\dot{V}$ is negative definite.

THEOREM 4.1—LYAPUNOV STABILITY
The differential equation

$$\dot{x}(t) = f(t, x(t)), \quad x \in \mathbb{R}^n, \ t \in \mathbb{R}, \tag{4.2}$$

where $f$ is Lipschitz continuous, is uniformly exponentially stable if there exists a continuously differentiable function $V(t,x)$ and positive numbers $\alpha_1, \alpha_2, \alpha_3$ such that

$$\begin{aligned} \alpha_1 |x|^2 < V(t,x) < \alpha_2 |x|^2 \\ \frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(t,x) \leq -\alpha_3 |x|^2 \end{aligned} \tag{4.3}$$

for all $x$ and $t$. □

This powerful theorem works fine when only one controller acts on the process. Theorem 4.1 is not easily generalized to the hybrid control system with several control laws. A fairly intuitive idea to gradually go from one stabilizing controller to another does not work. Equation 4.3 does not hold for linear combinations of control laws as shown in Example 4.1 even though there is a Lyapunov function for each separate controller.

EXAMPLE 4.1—LINEAR COMBINATION OF CONTROL LAWS
The triple integrator system

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u \\ y &= x_1 \\ u_1 &= -3x_1 - 2x_2 - 2x_3 \\ u_2 &= -0.37x_1 - 0.67x_2 - 0.67x_3, \end{aligned} \tag{4.4}$$

is stable if controlled by either controller $u_1$ or controller $u_2$. A linear combination $u = \frac{1}{4}(u_1 + 3u_2)$ will however lead to an unstable system. □

## Non-smooth Lyapunov functions

The Lyapunov theory will now be extended to cater for multi-controller systems. To begin with the differentiability condition is relaxed.

A non-smooth Lyapunov function is a Lyapunov function where the constraint $\dot{V}(x,t) < 0$ is replaced with

$$V(x(t_2), t_2) < V(x(t_1), t_1), \quad \text{if } t_2 > t_1.$$

Differentiability of $V(x,t)$ is not required. All that is needed is that $V(x,t)$ is decreasing in $t$.

In Shevitz and Paden (1994) the basic Lyapunov stability theorems are extended to the non-smooth case using Filippov solutions and Clarke's generalized gradient. Similar ideas in Paden and Sastry (1987) allow the authors to use a slightly modified Lyapunov theory to prove stability for variable structure systems in a nice way.

## Further generalizations

Lately there have been several attempts to generalize Lyapunov's stability results to multi-controller systems. A common way of doing it is to define a non-smooth Lyapunov function and then have a switching method that makes the non-smooth Lyapunov function decrease. Certain, natural, conjectures for a Lyapunov stability theorem are not valid in the presence of sliding modes. In a later section it will be shown how additional requirements on the hybrid controller can guarantee stability also in the presence of infinitely fast switching.

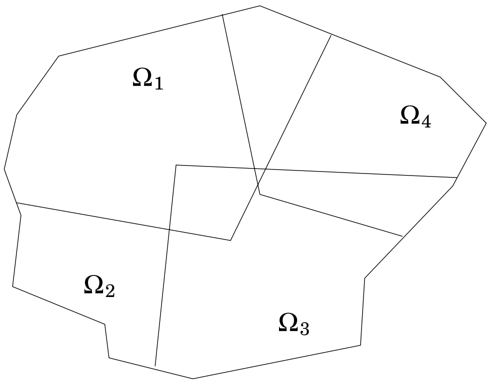

**Figure 4.1** The state space $\Omega$ is divided into regions $\Omega_i$. The regions can be overlapping and $\Omega = \cup_i \Omega_i$.

CONJECTURE 4.1—HYBRID LYAPUNOV STABILITY (WRONG)
Several control laws $u_i = c_i(x,t)$ are used to control a plant $\dot{x} = f(x,u,t)$. Each controller $c_i$ can be used in a region $\Omega_i$. The whole state space $\Omega$ is the union of regions $\Omega_i$ i.e. $\Omega = \cup_i \Omega_i$, where the regions $\Omega_i$ are allowed to overlap. Associated with each controller-region pair $(c_i, \Omega_i)$ is a function $D_i(x,t)$ such that

$$\frac{\partial D_i}{\partial t} + \frac{\partial D_i}{\partial x} f(x, u_i, t) \leq 0$$

if controller $c_i$ is used. A Lyapunov function candidate $D$ is now built by patching together local $D_i$ functions. The conjecture is that if switching from controller $c_i$ to controller $c_j$ is done only when $D_j \leq D_i$ then the global function $D$ is non increasing and the closed loop system is stable. □

As shown in the next example, this conjecture is *not* true if infinitely fast switching between the controllers is allowed.

EXAMPLE 4.2—A COUNTER EXAMPLE
Consider a simple switching system of the form

$$\dot{x}(t) = A_i x(t)$$
$$i(x) = \begin{cases} 1, & x_1 \geq 0 \\ 2, & x_1 < 0, \end{cases}$$

where the continuous dynamics are given by

$$A_1 = \begin{bmatrix} -5 & 1 \\ -10 & 1 \end{bmatrix}, \qquad A_2 = \begin{bmatrix} -5 & -1 \\ 10 & 1 \end{bmatrix}.$$

Both linear subsystems are exponentially stable and it is straight forward to construct a piecewise quadratic Lyapunov function candidate from locally decreasing functions $D_i$, where

$$D_i(x) = x^T P_i x$$

and

$$P_1 = \begin{bmatrix} 2.65 & -1.275 \\ -1.275 & 0.775 \end{bmatrix}, \qquad P_2 = \begin{bmatrix} 2.65 & 1.275 \\ 1.275 & 0.775 \end{bmatrix}.$$

This candidate Lyapunov function fulfills the conditions of Conjecture 4.1 in both regions. Since the candidate function is continuous across the switching manifold, Conjecture 4.1 suggests that the system should be stable. A simulation of the system, shown in Figure 4.2, shows that the system is unstable. The system reaches the switching manifold $x_1 = 0$ in finite time, after which the state is forced to an unstable motion on that switching manifold. The problem leading to instability in this example is

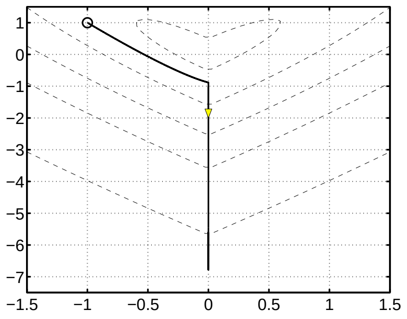

**Figure 4.2** Simulated system with unstable sliding mode (full line), and level curves of a continuous piecewise quadratic Lyapunov function (dashed line).

fast switching between $\dot{x} = A_1 x$ and $\dot{x} = A_2 x$ on the manifold $x_1 = 0$ □

***Sliding mode dynamics for Counter example 4.2*** In Chapter 2 it was seen that the different definitions of sliding dynamics coincided for affine systems. Either definition can thus be used for calculating the sliding dynamics in this example. The switching manifold is given by $C^T x = 0$ and the conditions for sliding give

$$\dot{y} = \alpha C^T A_1 x + (1-\alpha) A_2 x = [-5 \quad 2\alpha - 1]\, x = 0$$

Since this equation has a solution for $x_1 = 0$ for $\alpha = 1/2$, sliding can occur. The equivalent dynamics along the sliding manifold are thus

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \frac{1}{2}(A_1 + A_2) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -5 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ x_2 \end{bmatrix}, \qquad (4.5)$$

which are clearly unstable.

***Modified stability conditions*** What additional conditions are needed in order to guarantee stability in the case of infinitely fast switches? As in the false Conjecture 4.1, associate each controller $c_i$ with a decreasing function $D_i$. Choose any switching scheme such that a switch from $c_i$ to $c_j$ occurs only when $D_j \leq D_i$. Define $W$ as a global non-smooth Lyapunov function by patching together the local $D_i$ functions, $W = \alpha_i D_i$ where $\alpha_i = 1$ if controller $c_i$ is used and 0 otherwise. Using controller $c_i$ gives the equations $W = D_i$ and $\dot{x} = f_i$. As long as no switching occurs it is easily seen that,

$$W(x(t_2)) < W(x(t_1)), \quad \text{when } t_2 > t_1.$$

The critical part is to study the case when the solution trajectory is on a manifold where $D_i = D_j$. On this manifold the non-smooth Lyapunov function is $W = D_i = D_j$ and fast mode changes can occur. Further the dynamics can be written $\dot{x} = \alpha_i f_i + \alpha_j f_j$, where $\alpha_i$ and $\alpha_j$ are chosen in order to stay on the manifold. It is necessary to show that $W(x)$ is decreasing, i.e

$$\nabla D_i \cdot \dot{x} = \nabla D_i \cdot (\alpha_i f_i + \alpha_j f_j) < 0.$$

If all scalar products $\nabla D_i \cdot f_j < 0$ then $W(x)$ is decreasing for any combinations of $\alpha_i$ and $\alpha_j$. This is a sufficient but probably not a necessary condition. The condition can be extended to cater for fast switching between more than two controllers.

## 4.3 A Lyapunov theory based design method

This section describes a method for designing stable hybrid control systems. The method works with a set of controllers coupled to a set of Lyapunov functions. The emphasis of the design method is to guarantee stability. There will be some indication on what to do to improve performance.

### Problem description

The hybrid systems in this section are modeled as systems composed of a collection of continuous time systems $\{f_i\}$, $i = 1 \ldots n$, described by ordinary differential equations, and a static logic system that switches between the different continuous time systems. A common practical case is to have one continuous time process and several controllers

$$\begin{aligned} \dot{x} &= f(x,t,u) \\ u &= c_i(x,t) \\ i(t) &\in \{1,\ldots,n\}. \end{aligned} \tag{4.6}$$

A system $\{f_i\}$ is then a composition of the process and the $i$th controller,

$$\dot{x} = f_i(x,t) = f(x,t,c_i(x,t)).$$

Only the case of complete state information is considered here. If the systems $\{f_i\}$ are linear then this is a piecewise linear system. A discussion of such systems of low dimensions are found e.g. in Åström (1968). Discrete piecewise linear systems are analyzed in e.g. Sontag (1981).

If it is required that the control signal passed on to the process is smooth then it can be pre-filtered by a low-pass filter. This filter can be seen as part of the process. Lyapunov functions then have to be constructed for the combined filter-process system.

### Key idea

The key idea is to associate each subsystem $\{f_i\}$ with a separate Lyapunov function and to construct the switching logic device in such a way that the composite system is stable. This is done by constructing a global Lyapunov function for the composite system. There are several difficulties because the Lyapunov-like functions dealt with do not have smooth derivatives.

Three small examples will illustrate the method. The first example shows the importance of choosing the switch functions properly. Different choices can result in stable or unstable closed loop systems. The second example shows what happens in the case of sliding modes. The third example is the standard experiment of swinging up an inverted pendulum. In Åström and Furuta (1996) it is shown that a nice strategy for doing this is obtained by combining energy control with a stabilizing linear strategy. The results in this section can be used to derive a switching strategy that guarantees global stability.

### Combinations of Lyapunov functions

The Lyapunov theorem is first extended and a similar result is proved regarding combinations of several Lyapunov functions. The process is described with the general nonlinear Equation 4.6

Assume that the System 4.6 and the control law $u = c(x,t)$ is given together with a set of Lyapunov functions $V_1, \dots, V_n$. An interesting question is to find out what kind of combinations of $V_1, \dots, V_n$ are Lyapunov functions.

LEMMA 4.1—ONE CONTROLLER - SEVERAL LYAPUNOV FUNCTIONS
Let $\{V_i(x,t)\}$ be a set of given non-smooth Lyapunov functions for the system (4.6) when using the controller $u = c(x,t)$. Then the functions

$$\begin{aligned} W_1(x,t) &= \min_i [V_i(x,t)] \\ W_2(x,t) &= \max_i [V_i(x,t)] \end{aligned} \tag{4.7}$$

are also non-smooth Lyapunov functions.

***Proof*** The proof follows the ordinary proof of Lyapunov's stability theorem, see e.g. Khalil (1992), except that the time derivative, $\dot{W}$, is not used. In fact all that is needed is that $W_1(x(t),t)$ decreases monotonously with $t$. This is true since

$$\begin{aligned} W_1(x(t_2),t_2) &= \min_i [V_i(x(t_2),t_2)] \\ &< \min_i [V_i(x(t_1),t_1)] = W_1(x(t_1),t_1). \end{aligned}$$

The proof for $W_2$ is similar. □

***Remark*** More involved structures can be used for example

$$W = \min(\max(V_1,V_2), \max(V_3, \min(V_4,V_5))).$$

The function $W$ is a non-smooth Lyapunov function if $V_1,\dots,V_5$ are.

## Combining several control laws

Lemma 4.1 is now extended to the case where a set of controllers $c_1,\dots,c_n$ act on the system. As mentioned in the the introduction there are many different reasons for using several controllers. Some additional reasons are

- The controller $c_i$ stabilizes the system only in a certain domain $\Omega_i$,
- The performance obtained by $c_i$ is only good inside the domain $\Omega_i$.

A new method guaranteeing (global) stability when switching between the controllers will be developed.

The following four assumptions are made:

A1 The $\Omega_i$:s are open sets.

A2 For each pair ($c_i$, $\Omega_i$) there is a ($C^1$) Lyapunov function $V_i$.

A3 The separate $\Omega_i$:s cover the whole state space $\Omega$, i.e.

$$\Omega = \bigcup_{i=1}^{n} \Omega_i.$$

A4 At any forced transition from $c_i$ to $c_j$ the corresponding Lyapunov functions are equal, i.e. $V_i(x,t) = V_j(x,t)$.

A forced transition is a switch from controller $c_i$ to controller $c_j$ because that controller $c_i$ is no longer admissible as the system leaves region $\Omega_i$.

## Switching strategy

The idea is to use the controller corresponding to the smallest Lyapunov function. A problem that was not present in the one-controller case is the possibility of a sliding mode behavior. This can happen if more than one controller has the same Lyapunov function value and that value is the smallest.

DEFINITION 4.1—THE MIN-SWITCHING STRATEGY
Let $f_i(x,t)$ be the right-hand side of Equation 4.6 when control law $c_i$ is used. Use a control signal $u^*$ such that,

$$\dot{x} = \sum_{i=1}^{n} \alpha_i f_i(x,t). \tag{4.8}$$

where $\alpha_i \geq 0$ satisfies $\sum \alpha_i = 1$ and $\alpha_i = 0$ if either $x \notin \Omega_i$ or if $V_i(x,t) > \min_j[V_j(x,t)]$. If only one controller achieves the minimum then $\alpha = 1$ for that controller and all the other $\alpha_i$ are zero. □

It is assumed that the sliding dynamics satisfy Equation 4.8.

***Remark*** The $\alpha_i$ are in general not unique. The $\alpha_i$ depend on how many controllers achieve the minimum at the same time and also on the definition of sliding mode dynamics used.

Introduce $S_{ij} = V_i - V_j$. The set $S_{ij}(x,t) = 0$ is called the switching surface associated with the controller pair $c_i$ and $c_j$. To maintain a sliding motion on the surface $S_{ij}(x,t) = 0$ using the min-switch strategy it is necessary that

$$\begin{aligned} (V_i)_t + \nabla V_i \cdot f_j &< (V_j)_t + \nabla V_j \cdot f_j \\ (V_j)_t + \nabla V_j \cdot f_i &< (V_i)_t + \nabla V_i \cdot f_i. \end{aligned} \tag{4.9}$$

Otherwise the system will leave the surface $S_{ij}(x,t) = 0$. Consult Figure 4.3 for directions of the vector fields. Note that only the case with transversal vector fields are treated here.

## A Lyapunov theorem for hybrid control systems

THEOREM 4.2—MINIMUM OF LYAPUNOV FUNCTIONS
Let the system be given by Equation 4.6 and assume that A1–A4 hold. Introduce $W$ as

$$W = \min(V_1, V_2, \dots, V_n).$$

The closed loop system is stable with $W$ as a non-smooth Lyapunov function if the min-switch strategy in Definition 4.1 is used. □

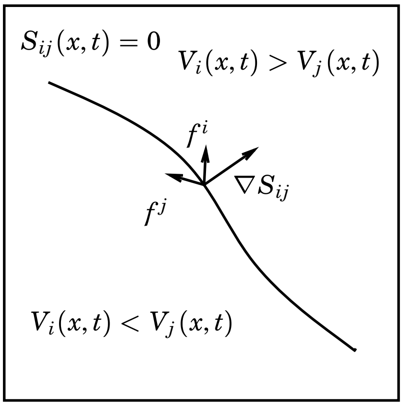

**Figure 4.3** The figure shows a phase plane close to the switching surface $S_{ij}(x,t) = V_i(x,t) - V_j(x,t) = 0$. The control law corresponding to the smallest Lyapunov function is used. Under certain conditions, see Equation 4.9, there will be a sliding motion on the switching surface.

***Proof*** Assume that $V_1(x,t) = \min_j[V_j(x,t)]$, for an open interval in $t$, and that $\alpha_1 > 0$. Then in this interval

$$\begin{aligned} \frac{d}{dt}W(x,t) &= (V_1)_t + \nabla V_1 \dot{x} \\ &= (V_1)_t + \sum_{i=1}^{n} \alpha_i \nabla V_1 \cdot f_i \\ &< \sum_{i=1}^{n} \alpha_i((V_i)_t + \nabla V_i \cdot f_i) \\ &< 0. \end{aligned}$$

The first inequality follows from the fact that if $\alpha_i > 0$ then

$$(V_1)_t + \nabla V_1 \cdot f_i < (V_i)_t + \nabla V_i \cdot f_i.$$

This is a sliding condition associated with the switching surface $S_{1i} = 0$. The last inequality follows from the fact that $V_i$ is a Lyapunov function for controller $c_i$, $i = 1, \dots, n$. If there is no sliding, only one controller is in use, then $\alpha_j = 0, i \neq j$.

The idea to use the minimum of Lyapunov functions is also used in Ferron (1996) and in Caines and Ortega (1994) where stability for a certain class of fuzzy controllers is analyzed.

## Controller design

The design of the hybrid control system is now reduced to designing sub-controllers $c_i$ with Lyapunov functions $V_i$. The Lyapunov functions as well as the sub-controllers can be of different types. Some design methods, such as LQG, provides a Lyapunov function with the control law. Special care has to be taken when a controllers validity region $\Omega_i \neq \Omega$.

***Lyapunov function transformations*** The switching method gives stability only. To achieve better performance it can be necessary to change the location of the switching surfaces. This can to some degree be achieved by different transformations. One example is transformations of the form

$$\tilde{V}_i = g_i(V_i), \tag{4.10}$$

where $g_i(\cdot)$ are monotonously increasing functions.

For some controller design methods it is possible to add constraints guiding the switching surfaces to certain regions. This is easier to accomplish if the sub-controllers are stabilizing in all $\Omega$.

## Examples

This section contains three examples where the design method described above is applied.

EXAMPLE 4.3—SWITCHED CONTROL

The first example illustrates the well-known fact that two controllers that stabilize a system can not be combined arbitrarily to give a stable controller and illustrates how the switching strategy defined above works. Assume that two controllers, $c_1$ and $c_2$, that give stable closed loop systems are found. The closed loop systems are

$$\dot{x} = \begin{bmatrix} -1 & 5 \\ 0 & -1 \end{bmatrix} x = f_1(x), \qquad \dot{x} = \begin{bmatrix} -1 & 0 \\ -5 & -1 \end{bmatrix} x = f_2(x).$$

***Ad-hoc switching strategies*** Switching between the controllers $c_1$ and $c_2$ can result in a stable or an unstable system depending on how the switching is done. For simplicity define the new coordinates

$$z = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} x.$$

Now use controller $c_1$ when $z_1 \cdot z_2 > 0$ and else controller $c_2$. Figure 4.4 shows the trajectories for three different ad-hoc switch strategies corresponding to $\alpha = [45°, 30°, 32.8°]$. The different choices result in a stable system, an unstable system and a system with a limit cycle.

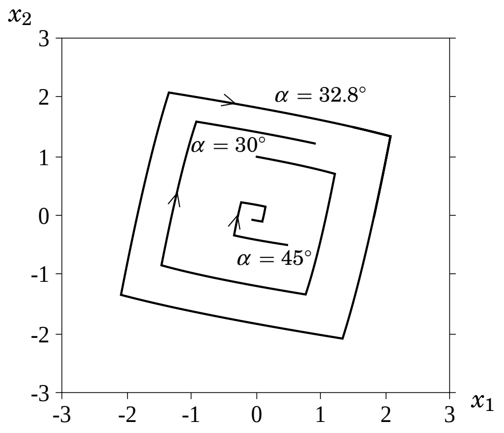

**Figure 4.4** Simulation of Example 4.3 with $\alpha$-strategies. The stability of the ad-hoc switching strategy depends on $\alpha$. The switching strategy defined by the Lyapunov functions in Equation 4.11 corresponds to $\alpha = 45°$, which gives a stable system.

***Stabilizing switching strategy*** To illustrate how the stability theorem works, pick two functions $V_1$ and $V_2$ that are Lyapunov functions when applying $c_1$ and $c_2$ respectively, for example,

$$V_1 = x_1^2 + 10x_2^2, \qquad V_2 = 10x_1^2 + x_2^2. \tag{4.11}$$

Define

$$W(x) = \min(V_1(x), V_2(x)) \tag{4.12}$$

and use the min-switching strategy above, i.e. control with $c_1$ if $V_1 < V_2$ and else with $c_2$. Now $W(x)$ is a non-smooth Lyapunov function.

With this choice of Lyapunov functions the switch surface is given by the equation

$$S(x) = V_1(x) - V_2(x) = 9x_2^2 - 9x_1^2 = 0.$$

Hence $S$ consists of the two lines, $S_1 : x_2 = x_1$ and $S_2 : x_2 = -x_1$. An inspection of the vector fields on these lines gives that all trajectories go straight through them and no sliding motion occurs.

The switching strategy given by the Lyapunov functions in equation 4.11 and Theorem 4.2 is equivalent to the case $\alpha = 45°$ above. The resulting system is stable as can be seen in Figure 4.4. □

EXAMPLE 4.4—SLIDING MOTION
This second example illustrates the case where more than one Lyapunov function achieves the minimum.

Two different control laws $u_1$ and $u_2$ are applied to the same system. Each control law gives a stable closed loop system

$$\dot{x} = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix} x = f(x, u_1), \qquad \dot{x} = \begin{bmatrix} -1 & -1 \\ 1 & -1 \end{bmatrix} x = f(x, u_2).$$

Two functions $V_1$ and $V_2$ are Lyapunov functions when applying $u_1$ and $u_2$ respectively are found

$$V_1 = 2x_1^2 + x_2^2, \qquad V_2 = x_1^2 + 2x_2^2. \tag{4.13}$$

Define

$$W(x) = \min(V_1(x), V_2(x)), \tag{4.14}$$

and use the switching strategy from Theorem 4.2 i.e. control with $u_2$ if $V_1 > V_2$ and else with $u_1$. Now $W(x)$ is a non-smooth Lyapunov function. In this example the switching surface consists of the two lines $S_1 : x_1 = x_2$

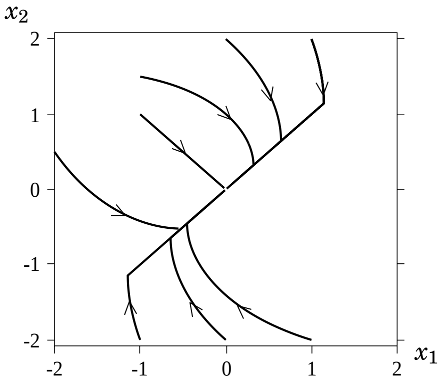

**Figure 4.5** Simulation of Example 4.4 for several values of initial conditions.

and $S_2 : x_1 = -x_2$. As usual the vector fields $f^-$ and $f^+$ are inspected on the surface $S(x) = V_1(x) - V_2(x) = x_2^2 - x_1^2 = 0$.

$$\nabla S^T = \begin{bmatrix} -2x_1 & 2x_2 \end{bmatrix}, \qquad f^- = \begin{bmatrix} -x_1 + x_2 \\ -x_1 - x_2 \end{bmatrix}, \qquad f^+ = \begin{bmatrix} -x_1 - x_2 \\ x_1 - x_2 \end{bmatrix}$$

along $S_1$ we have, $\nabla S_1^T \cdot f^- = -4x_2^2$ and $\nabla S_1^T \cdot f^+ = 4x_2^2$. Along $S_2$ it holds that, $\nabla S_2^T \cdot f^- = 4x_2^2$ and $\nabla S_2^T \cdot f^+ = -4x_2^2$. Thus $S_1$ is attractive and $S_2$ is not. After $S_1$ is reached the rest of the motion will be on $S_1$. For sample trajectories for different initial conditions see Figure 4.5. □

EXAMPLE 4.5—INVERTED PENDULUM
In this section a hybrid controller for the inverted pendulum is designed with the method above. The inverted pendulum experiment is depicted in Figure 4.5.

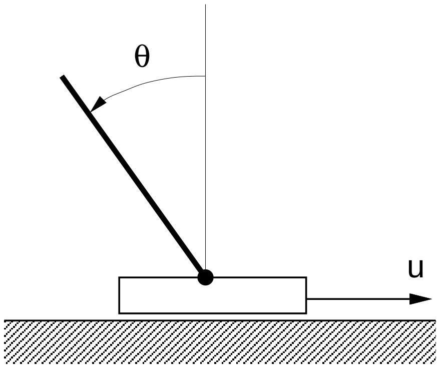

**Figure 4.6** The inverted pendulum in Example 4.5.

***Equations of motion*** The dynamics for the inverted pendulum are

$$J_p\ddot{\theta} - mgl\sin\theta + mul\cos\theta = 0, \tag{4.15}$$

where $m$ is the mass, $l$ is the distance from the pivot to the center of mass, $g$ is the gravitational constant and $u$ is the acceleration of the pivot. The angle $\theta$ is 0 when the pendulum is in an upright position.

Two different controllers are used. The first controller is an energy controller that will swing-up the pendulum. The second controller is a state feedback controller that will hold it in the upright position. The energy controller is stabilizing for all initial conditions, $\Omega_E = \Omega$ whereas the feedback controllers is stabilizing only within a certain region, $\Omega_{FB}$. The performance of the feedback controller is better than the energy controller in the upright position.

***Energy Control*** The pendulum energy is

$$E = \frac{1}{2}J_p\dot{\theta}^2 + mgl(\cos\theta - 1), \tag{4.16}$$

where the potential energy has been chosen to give zero energy when the pendulum is stationary in the upright position. The energy controller

$$u_E = \text{sat}_{ng}(kE\text{sign}(\dot{\theta}\cos\theta)), \tag{4.17}$$

where *ng* is the maximum amplitude of the controller, is described in the paper by Åström and Furuta (1996). The function $V_E = E^2 + \Delta$ (where $\Delta$ is a positive constant) can be used as a Lyapunov function since

$$\begin{aligned}\frac{dV_E}{dt} &= 2E\frac{dE}{dt}\\ &= 2E(J_p\dot{\theta}\ddot{\theta} - mgl\dot{\theta}\sin\theta)\\ &= 2E(-mul\dot{\theta}\cos\theta)\\ &= 2E(-ml\dot{\theta}\cos\theta)\text{sat}_{ng}kE\text{sign}(\dot{\theta}\cos\theta) \le 0.\end{aligned} \tag{4.18}$$

$\dot{V}_E$ is zero only when $\dot{\theta} = 0$ or $\theta = \pm\frac{\pi}{2}$ and the only such point that is also an equilibrium point of the closed loop system is $\theta = \dot{\theta} = 0$. Hence the closed loop system is globally stable.

One drawback with this energy controller is that it is not very *efficient* close to the equilibrium point. When the energy is close to 0 the system can still be far from the point ($\theta = 0$, $\dot{\theta} = 0$). Therefore it is preferred only to use this controller in a domain $\Omega_1$ where its Lyapunov function is large.

***Linear Feedback Control*** The second controller is a linear feedback controller given by

$$u_{FB} = l_1\theta + l_2\dot{\theta}.$$

A function on the form

$$V_{FB}(x) = \theta^2 + \gamma\dot{\theta}^2$$

is a Lyapunov function for some value of $\gamma$. Not all values of $\gamma$ give Lyapunov functions but there is still some design freedom. Note that $V_{FB}$ is a Lyapunov function within a specific domain only, $\Omega_2$, in the ($\theta$, $\dot{\theta}$)-plane. The size and shape of the domain $\Omega_2$ depends on the choice of $\gamma$.

***Min-control*** The control law $u_{FB}$ is used whenever $V_{FB} < V_E + \Delta$ and $(\theta,\ \dot{\theta}) \in \Omega_2$, otherwise $u_E$ is used. The constant $\Delta$ is added to assure that the energy controller is not switched back in at the upright position. the constant $\Delta$ should be in the interval $(0, V_{FB}(\partial(\Omega_{FB})))$, where $V_{FB}(\partial(\Omega_{FB}))$ is the value of $V_{FB}$ on the stability boundary for the linear feedback controller. Figure 4.7 shows a swing-up and catch of the pendulum. It is captured by the linear controller at the point $t = 7.34$, $\theta = 6.1$, $\dot{\theta} = 0.18$. □

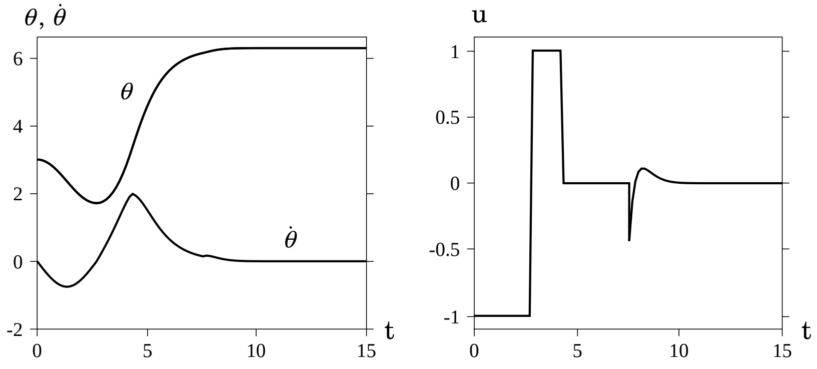

**Figure 4.7** Simulation of the pendulum using the switching strategy based on $\min(V_E, V_{FB})$. Only one switch occurs, at time $t = 7.34$, from the control law $u_E$ to $u_{FB}$. The system is caught and stabilized in the upright position.

## 4.4 Summary

This chapter introduced a design method for hybrid control systems that gives closed loop stability. The design method is based on local controllers and local Lyapunov functions. A global Lyapunov function $W$ is build from the local Lyapunov function. A switching strategy is then chosen as to guarantee that the non-smooth Lyapunov function $W$ is decreasing.

The method was applied to three examples and some modifications to avoid chattering were discussed.

Lyapunov theory is not easily extended to hybrid systems. A fairly natural stability conjecture was proven to be wrong.

# 5

# Experiments with hybrid controllers

## 5.1 Introduction

This chapter describes two experiments with hybrid control systems. To illustrate the theoretical results and to get some experience of practical issues the simple hybrid controller of the last chapter was implemented. The experiments were performed in two different software environments but with similar basic control algorithms. Some general points of programming languages for control systems will also be discussed.

## 5.2 A hybrid tank controller

In process control it is common practice to use PI control for steady state regulation and to use manual control for large set-point changes. In this experiment it is tried to combine the steady state regulation with a minimum time controller for the set-point changes.

First the process and the process model are introduced. Then the hybrid controller will be motivated, and the design for the sub-controllers is presented. It is shown that the use of this hybrid controller improves performance compared with a pure PID controller. Both good response to set-point changes and good disturbance rejection are achieved.

### The Process

The process to be controlled consists of two water tanks in series, see Figure 5.1. The goal is to control the level ($x_2$) of the lower tank and

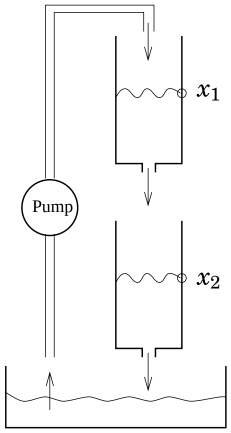

**Figure 5.1** The double-tank process.

indirectly the level $(x_1)$ of the upper tank. The two tank levels are both measurable. Choosing the level of tank $i$ as state $x_i$ the following state space description is derived

$$\dot{x} = f(x,u) = \begin{bmatrix} -\alpha_1\sqrt{x_1} + \beta u \\ \alpha_1\sqrt{x_1} - \alpha_2\sqrt{x_2} \end{bmatrix}, \tag{5.1}$$

where the inflow $u$ is the control variable. The inflow can never be negative and the maximum flow is $\bar{u} = 27 \cdot 10^{-6}\ m^3/s$. Furthermore, in this experimental setting the tank areas and the outflow areas are the same for both tanks, giving $\alpha_1 = \alpha_2$. The square root expression is derived from Bernoulli's energy equations.

## The Controller

As mentioned above a controller structure with two sub-controllers and a supervisory switching scheme will be used. The time-optimal controller is used when the states are far away from the reference point. Coming closer the PID controller will automatically be switched in to replace the time-optimal controller. At each different set-point the controller is redesigned, keeping the same structure but using reference point dependent parameters. Figure 5.2 describes the algorithm with a Grafcet diagram, see David and Alla (1992). The Grafcet diagram for the tank controller consists of four states. Initially no controller is in use. This is the **Init** state. **Opt** is the state where the time-optimal controller is active and **PID** is the state for the PID controller. The **Ref** state is an intermediate state used for calculating new controller parameters before switching to a new time-optimal controller.

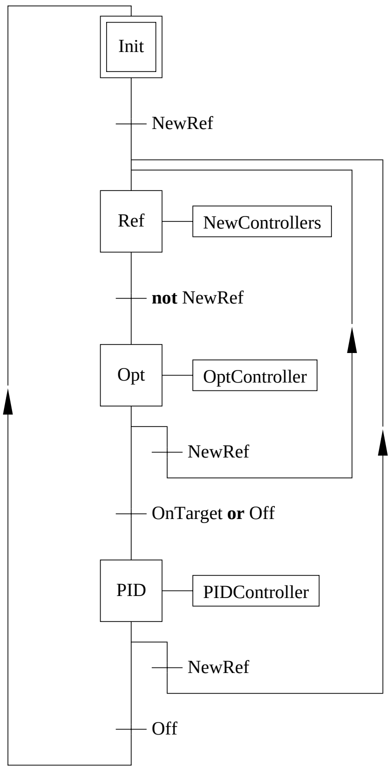

**Figure 5.2** A Grafcet diagram describing the control algorithm.

The sub-controller designs are based on a linearized version of Equation 5.1

$$\dot{x} = \begin{bmatrix} -a & 0 \\ a & -a \end{bmatrix} x + \begin{bmatrix} b \\ 0 \end{bmatrix} u, \tag{5.2}$$

where the parameter $b$ has been scaled so that the new control variable $u$ is in $[0, 1]$. The parameters $a$ and $b$ are functions of $\alpha$, $\beta$ and the reference level around which the linearization is done. It is later shown how the neglected nonlinearities affect the performance. To be able to switch in the PID controller, a fairly accurate knowledge of the parameters is needed.

***PID controller design*** A standard PID controller on the form

$$G_{PID} = K(1 + \frac{1}{sT_I} + sT_d)$$

is used. The design of the PID controller parameters $K$, $T_d$ and $T_I$ is based on the linear second order transfer function,

$$G(s) \quad = \frac{ab}{(s+a)^2},$$

derived from Equation 5.2. Let the desired closed loop characteristic equation be

$$(s+\alpha\omega)(s^2+2\zeta\omega s+\omega^2).$$

The parameters $(\alpha,\omega,\zeta) = (1.0, 0.06, 0.7)$ are chosen for a reasonable behavior, both in case of set-point changes and under load disturbances. For

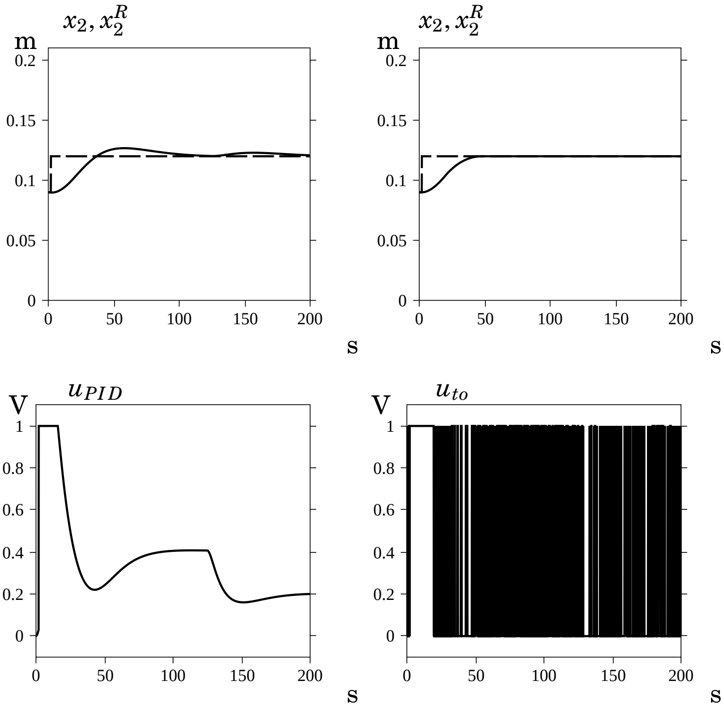

**Figure 5.3** Pure PID (left) and pure time-optimal control (right). The PID controller has a smooth control signal but gives a slow response with an overshoot. The time-optimal controller gives a fast response but the control signal is unusable in practice

some systems it is possible to get a smaller overshoot by set-point weighting. Figure 5.3 left shows the set-point and load disturbance responses for the PID controller. When implementing the real-time version of the PID algorithm a low-pass filter is used on the derivative part.

***Time-optimal controller design*** The theory of optimal control is well established, see Lewis (1986) and Leitman (1981). This theory is applied to derive minimum time strategies to bring the system as fast as possible from one set-point to another. The Pontryagin maximum principle is used to show that the time-optimal control strategy for the System 5.1 is of bang-bang nature. The time-optimal control is the solution to the following optimization problem

$$\max J = \max \int_0^T -1 \cdot dt \tag{5.3}$$

under the constraints

$$\begin{aligned} x(0) &= [x_1^0 \quad x_2^0]^T \\ x(T) &= [x_1^R \quad x_2^R]^T \\ u &\in [0,1], \end{aligned}$$

together with the dynamics in Equation 5.2. The Hamiltonian, $H(x,u,\lambda)$, for this problem is

$$H = -1 + \lambda_1(-a\sqrt{x_1} + bu) + \lambda_2(a\sqrt{x_1} - a\sqrt{x_2}),$$

with the adjoint equations, $\dot{\lambda} = -\frac{\partial H}{\partial x}$,

$$\begin{bmatrix} \dot{\lambda}_1 \\ \dot{\lambda}_2 \end{bmatrix} = \begin{bmatrix} -\frac{a}{2\sqrt{x_1}} & \frac{a}{2\sqrt{x_1}} \\ 0 & -\frac{a}{2\sqrt{x_2}} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}. \tag{5.4}$$

To derive the optimal control signal the complete solution to these equations is not needed. It is sufficient to note that the solutions to the adjoint equations are monotonous. This together with the switching function from Equation 5.4, i.e. $\sigma = \lambda_1 bu$, give the optimal control signal sequence that minimizes $H(u)$. Possible optimal control sequences are

$$\langle 0,1\rangle, \langle 1,0\rangle, \langle 0\rangle, \langle 1\rangle.$$

For linear systems of order two, there can be at most one switch between the maximum and minimum control signal value (it is assumed that the tanks never are empty $x_i > 0$).

The switching times are determined by the new and the old set-points. In practice it is preferable to have a feedback loop instead of pre-calculated switching times. Hence an analytical solution for the switching curves is needed. For the linearized equation it is possible to derive the switching curve

$$x_2(x_1) \quad = \quad \frac{1}{a}[(ax_1 - b\bar{u})(1 + \ln(\frac{ax_1^R - b\bar{u}}{ax_1 - b\bar{u}})) + b\bar{u}],$$

where $\bar{u}$ takes values in $\{0, 1\}$. The time-optimal control signal is $u = 0$ above the switching curve and $u = 1$ below, for switching curves see Figure 5.4.

The fact that the nonlinear system has the same optimal control sequence as the linearized system makes it possible to simulate the nonlinear switching curves and to compare them with the linear switching curves. Simulation is done in the following way: initialize the state to the value of a desired set-point and simulate backwards in time.

Note that the linear and the nonlinear switching curves are quite close for the double-tank model, see Figure 5.4. The diagonal line is the set of equilibrium points, $x_1^R = x_2^R$. Figure 5.4 shows that the linear switching curves are always below the nonlinear switching curves. This will cause the time-optimal controller to switch either too late or too soon. It is

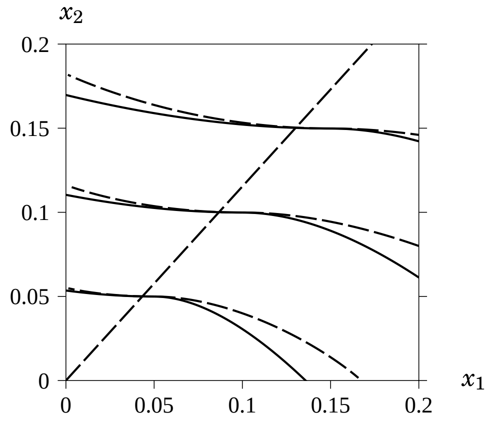

**Figure 5.4** Linear (full) and nonlinear (dashed) switching curves for different set-points. Above the switching lines the minimum control signal is applied. Below the lines the maximum control signal is used

not necessary to use the exact nonlinear switching curves since the time-optimal controller is only used to bring the system close to the new set-point. When sufficiently close, the PID controller takes over.

## Stabilizing switching-schemes

As seen in Chapter 4 it can happen that switching between stabilizing controllers may lead to an unstable closed loop system. It is therefore necessary to have a switching scheme that guarantees stability. Consider the system

$$\begin{aligned} \dot{x} &= f(x,t,u_i) \\ u_i &= c_i(x,t), \end{aligned} \tag{5.5}$$

where the $c_i(x,t)$ represent different controllers. In a hybrid control system different controllers are switched in for different regions of the state space or in different operating modes. There exist some switching schemes that guarantee stability. One of these is the min-switch strategy described in Chapter 4. In this application only two controllers are used.

***Lyapunov function modifications*** From a control designer's point of view the design of a hybrid control scheme using the *min-switching strategy* can be reduced to separate designs of $n$ different control laws and their corresponding Lyapunov functions. To improve performance it is often convenient to change the location of the switching surfaces. This can, to some degree, be achieved by different transformations of the Lyapunov functions. One example was transformations of the form, $\tilde{V}_i = g_i(V_i)$, where $g_i(\cdot)$ are monotonously increasing functions.

In some cases there can be very fast switching, chattering, between two or more controllers having the same value of their respective Lyapunov function. The hybrid controller is still stabilizing but it not desired behavior in a practical implementation. One way to avoid this chattering is to add a constant $\Delta$ to the Lyapunov functions that are switched out and subtract $\Delta$ from the Lyapunov functions that are switched in. This works as a hysteresis function. For two controllers with Lyapunov functions $V_1$ and $V_2$ the equations are $V_1 = V_1 + \Delta$ and $V_2 = V_2$ if controller two is in use and $V_1 = V_1$ and $V_2 = V_2 + \Delta$ if controller one is controlling the process. This guarantees that a controller is used for a time period $t > 0$ before it is switched out. It is easily shown that the hybrid controller is globally stabilizing with this addition.

## Simulations

In this section some different switching methods are evaluated. In all simulations a switching surface for the time-optimal controller based on the linearized equations is used.

All simulations have been done in the Omola/Omsim environment Andersson (1994), which supports the use of hybrid systems.

***Pure time-optimal and pure PID control*** The first simulation in Figure 5.3, shows control of a linearized system using either a time-optimal controller or a PID controller. The PID controller is tuned aggressively to give the same rise time as the minimum time controller. In practice, more conservative tuning must me made, otherwise measurement noise will generate large control actions. Note that PID control gives a large overshoot. The time optimal controller works fine until the level of the lower tanks reaches its new set-point. Then the control signal starts to chatter between its minimum and maximum value.

***A simple switching strategy*** A natural switching strategy would be to pick the best parts from both PID control and time-optimal control. One way to accomplish this is to use the time-optimal controller when far away from the equilibrium point and the PID controller when coming closer.

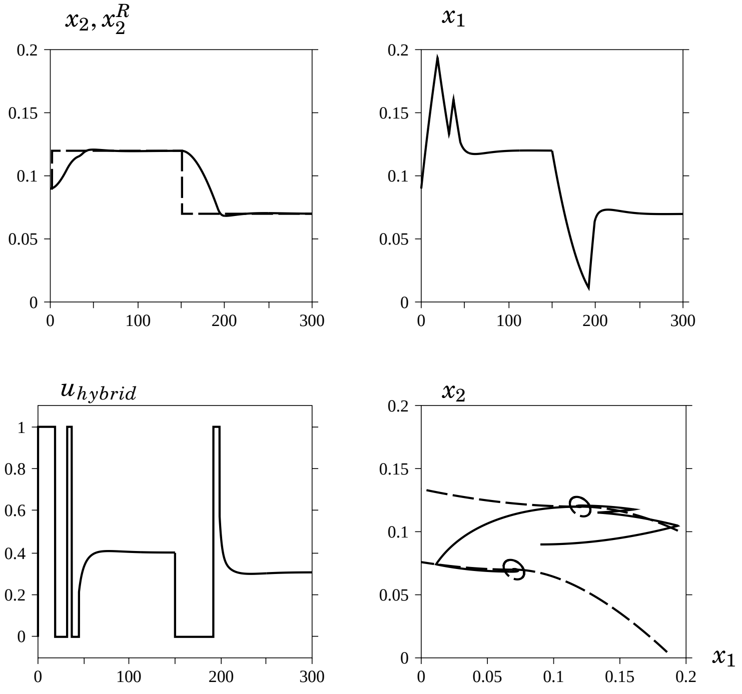

**Figure 5.5** Simulation of the simple switching strategy. Lower left figure shows the control signal. The time-optimal controller makes one extra min-max switch because of the nonlinearity. Catching regions are shown in lower right sub-figure.

As a measure of closeness the function $V_{close}$ is used,

$$V_{close} = \begin{bmatrix} x_1^R - x_1 \\ x_2^R - x_2 \end{bmatrix}^T P(\theta, \gamma) \begin{bmatrix} x_1^R - x_1 \\ x_2^R - x_2 \end{bmatrix},$$

where

$$P(\theta,\gamma) = \begin{bmatrix} \cos^2\theta + \gamma\sin^2\theta & (1-\gamma)\sin\theta\cos\theta \\ (1-\gamma)\sin\theta\cos\theta & \sin^2\theta + \gamma\cos^2\theta \end{bmatrix}.$$

The switching strategy here is to start with the time-optimal controller and then switch to the PID controller when $V_{close} < \rho$. The size and shape of the catching region may be changed with the $\gamma$ and $\theta$ parameters. The $P$ matrix above gives ellipsoidal catching curves. In this simulation switching back to the time-optimal controller is not allowed until there is a new reference value. See Figure 5.2 for a graphical description of the algorithm. The simulation results in Figure 5.5, show how the best parts from the sub-controllers are used to give very good performance.

***Lyapunov theory based switching*** In this third simulation set the *min switching strategy* that guarantees stability for the linearized system is used. The two Lyapunov functions are defined as

$$\begin{aligned} V_{PID} &= \begin{bmatrix} x_1^R - x_1 \\ x_2^R - x_2 \\ x_3^R - x_3 \end{bmatrix}^T P(\theta,\gamma) \begin{bmatrix} x_1^R - x_1 \\ x_2^R - x_2 \\ x_3^R - x_3 \end{bmatrix} \\ V_{TO} &= \textit{time left to reach new set-point} \end{aligned}$$

$$P(\theta,\gamma) = \begin{bmatrix} \cos^2\theta + \gamma_1\sin^2\theta & (1-\gamma_1)\sin\theta\cos\theta & p_{13} \\ (1-\gamma_1)\sin\theta\cos\theta & \sin^2\theta + \gamma_2\cos^2\theta & p_{23} \\ p_{13} & p_{23} & \gamma_2 \end{bmatrix}.$$

The state $x_3$ is the integrator state in the PID controller and $x_3^R$ is its steady state value. As in the previous simulation set the parameters $\gamma$ and $\theta$ are used to shape the catching region. The new state $x_3$ is preset to its value at the new equilibrium point, i.e. $x_3^R$, any time there is a set-point change. This state is then not updated until after the first switch to PID control. Using this method a similar two-dimensional catching region as in the previous simulation set is constructed. The simulation results are presented in Figure 5.6.

This supervisory scheme may lead to two types of chattering behavior. One is only related to the time-optimal controller and is due to the nonlinearities. The nonlinear switching curve lies above the linear, see Figure 5.4. That causes the trajectory of the nonlinear system to cross the linear switching curve and the control signal goes from 0 to 1 somewhat too late or too early. One way to remove this problem is to introduce

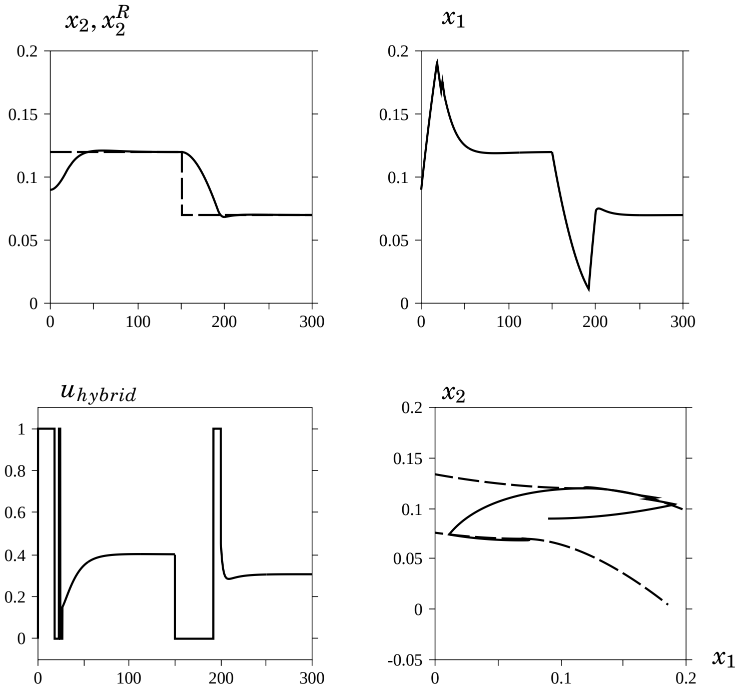

**Figure 5.6** Lyapunov based switching. Similar result as in Figure 5.5

a hysteresis when going from minimum to maximum control signal in the time optimal controller. There can also be chattering between the PID and the time-optimal controller if their corresponding Lyapunov functions have the same value. One solution to this problem is to add and remove the constant $\Delta$ as discussed in the section on Lyapunov functions modifications.

## Experiments

The theory and the simulations have also been verified by experiments. For simplicity only the switching strategy in Sec. 5.2 is implemented. Figure 5.7 shows the results of that experiment with the double-tanks.

The measurements from the lab process have a high noise level as can be seen in Figure 5.7. A first order low-pass filter

$$G_f(s) = \frac{1}{s+1}$$

is used to eliminate some of it. To further reduce the impact of the noise,

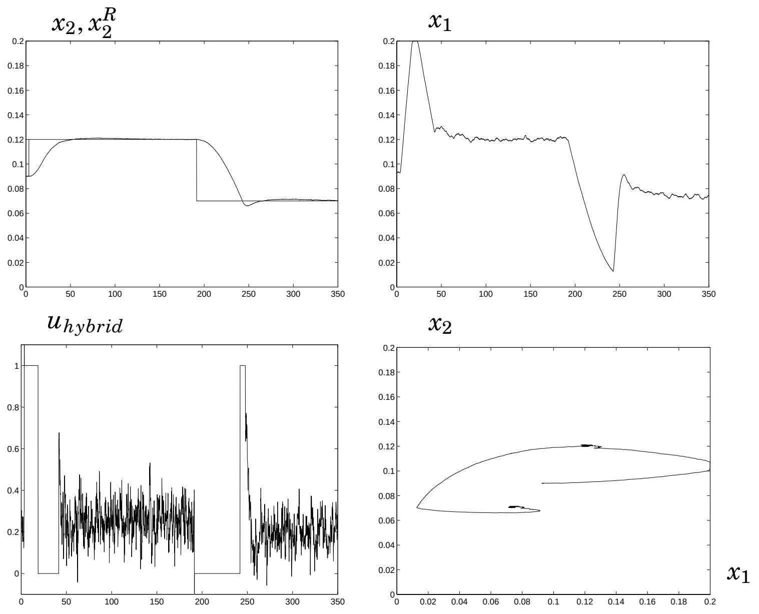

**Figure 5.7** Lab experiment. The simple switching strategy is used. The noise level is rather high.

a filter is added to the derivative part of the PID controller in a standard way.

The parameters in the simulation model were chosen to match the parameters of the lab process. It is thus possible to compare the experimental results directly with the simulations. Comparison of Figure 5.5 and Figure 5.7 shows the close correspondence between simulation and experimental results.

***Model mismatch*** During experiments it was found that the difference between the linear and the nonlinear switching curves were not so important. It resulted in a few more $\langle min, max \rangle$ switches before reaching the target area where the PID controller took over. However, a good model of the static gain in the system is needed. If there is a large deviation it cannot be guaranteed that the equilibrium points are within the catching regions. The catching region is defined as a ellipsoid around the theoretical equilibrium points.

***Adaptation*** It is not difficult to incorporate some process parameter estimation functions in the code. This would get a good estimate of the static gain.

Points on the switching curves could be stored in a table instead of using the analytical expressions. If the switch is too early or too late because of the nonlinearities then this information could be used to update the switching curve table.

## Implementation issues

The traditional way of implementing real-time systems using languages such as C or ADA gives very poor support for algorithms expressed in a state machine fashion. The need for a convenient way to implement hybrid systems is evident. The programming language must allow the designer to code the controller as a state-machine, as a period process, or as a combination of both. The later alternative is particularly useful in the case where the controller is divided into several controllers sharing some common code. A typical example of this is a state feedback controller for a plant, where it is not possible to measure all states. To get information about the non-measurable states an observer or a filter can be used. Typically this information is needed by the whole set of controllers, and thus the controller itself can be implemented as a hybrid system, consisting of one global periodic task that handles the filtering of process data, and a set of controllers that can be either active or inactive. Many of the hybrid controllers from Chapter 4 have the same sorts of demands on the programming language.

Implementing complex control algorithms puts high demands on the software environment. Automatic code generation and verification are needed together with advanced debugging and testing facilities.

***PAL and* Pålsjö** PAL Blomdell (1997) is a dedicated control language, which supports a mixture of periodic and sequential algorithms. Furthermore, the language supports data-types such as polynomials and matrices, which are extensively used in control theory. The implementation of the hybrid controller for a double-tank system see Section 5.2 is written in PAL. PAL has an run-time environment PÅLSJÖ, Eker and Blomdell (1996), that is well suited for experiments with hybrid control. PÅLSJÖ was developed to meet the needs for a software environment for dynamically configurable embedded control systems. PÅLSJÖ features: rapid prototyping, code re-usability, expandability, on-line configurability, portability,and efficiency

For a more exhaustive description of PAL and PÅLSJÖ see the licenciate thesis by Eker (1997).

***PAL Code*** This section presents some of the special features in PAL. The full PAL code can be found in Appendix A. PAL has all the usual functions in a programming language as well as some special features for implementing control systems. External functions can be imported or defined in PAL.

```
function ln(r : input real) : real;external "ln";

  function Switch(z1 : input real;z3 : input real;ubar : input real : real;
  begin
    result := 1.0/a * ((a * z1 - b * ubar) * (1.0 + ln((a * z3 - b * ubar)/
                (a * z1 - b * ubar))) + b * ubar);
  end Switch;
```

The code in the **calculate** and **update** blocks are common for all controller modes. In these blocks typically inputs are read and input signals are filtered.

```
calculate
begin
  x1 := (1.0 - c * h) * x1 + c * h * y1;
  x2 := (1.0 - c * h) * x2 + c * h * y2;
  xref := yref * Kc;
end calculate;

update
begin
  Vpid := gamma1 * ((x1 - xref) * (x1 - xref)+
              gamma2 * (x2 - xref) * (x2 - xref));
  yold := x2;
end update;
```

The structure as seen in the Grafcet diagram is built from **steps** and **transitions**

```
step Opt;
  activate OptController;
end Opt;

step PID;
  activate PIDController;
end PID;

transition from Opt to PID when OnTarget or Off;
```

For each **step** there is an **action** defining what is actually to be done in that step. The PID controller **action** is

```
action PIDController;
  usat : real;
begin
  OnTarget := false;
```

```
    $e := xref - x2;$
    $P := K * e;$
    $D := d1 * D + d2 * (yold - x2);$
    $u := P + I + D;$
    $usat := Sat(umin, umax, u);$
    $v := u;$
    $I := I + i1 * e + w1 * (usat - u);$
  **end** PIDController;
```

Notice the transparency in the description of continuous and discrete part. The switching between modes are explicitly given as transitions.

### Summary

A hybrid controller for a double-tank system has been designed and implemented. Both simulations and real experiments were presented.

It was shown that a hybrid controller, consisting of a time-optimal controller together with a PID controller gives very good performance.

The controller is easy to implement. It gives, in one of its forms, a guaranteed closed loop stability.

It solves a practical and frequent problem. Many operators go to manual control when handling start-up and set-point changes.

It is fairly easy to combine this method with a tuning experiment that gives a second order model of the system. From the tuning experiments it is easy to automatically generate a PID controller and an approximate time-optimal controller.

The model mismatch was not serious. The nonlinearity leads to some additional ⟨*min, max*⟩ switches.

## 5.3 A heating/ventilation problem

The ultimate goal was to implement the fast set-point control algorithms on an existing control system and solve a real problem. Several industries were contacted to find suitable test cases that were relevant and not too complicated. It was found that there were good examples in heating, ventilation and air conditioning.

The company Diana Control AB was interested in a cooperation and fortunately they could supply both an interesting problem and a control system. In a school in Klippan, see Figure 5.8, they use a combined heating and ventilation system where pre-heated air is blown into the classrooms. Specifically the control problem is that they want to have two different settings for the ventilation air temperature, one during the day and one during the night. Another related problem is that when the system has

**Figure 5.8** The school in Klippan.

been shut down for some reason it is necessary to start from considerably lower temperatures. The air is heated in a heat-exchanger and the incoming air has outdoor temperature, which can be very low in Sweden during the winter.

The present PID controller was tuned very conservatively to be able to handle these cold starts without too large overshoots. This was exactly the problem type that the fast set-point hybrid controller was designed for. A time-optimal controller handling set-point changes and system starts could be added to the existent PID controller.

***The ventilation air temperature control loop*** Figure 5.9 shows the ventilation process as it is graphically represented in the Diana control system. The control signal, $u$, is the set-point for the opening of the valve. Hot water flows through the valve to a heat-exchanger and the air is heated. The output of the system is the air temperature, $T$, after the fan. This heated air then goes to the classrooms in the school and the final temperatures in the classrooms are controlled with radiators on a separate control system.

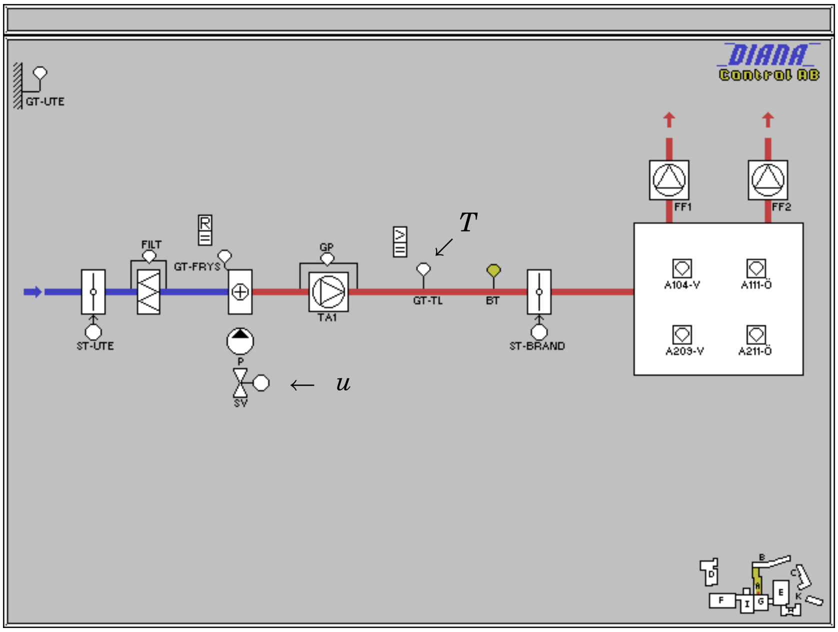

**Figure 5.9** The air temperature control loop. The water flow to the heat-exchanger is controlled by a valve via the control signal $u$. The output is the air temperature $T$ after the fan.

## Multiple modeling and design steps

It was a requirement not to disturb the school kids too much with heavy oscillations in the temperature. This was not a problem as only the system identification and the final tests actually affected the kids. The modeling and controller design problem was solved in different steps

- identification of a model,
- building a simulation model,
- design of the hybrid controller,
- implementation on a DIANA ADP 2000 and testing against a real-time simulation model,
- final testing at the school.

## System identification

The identification data was logged on the Diana ADP 2000 running in manual control mode and then uploaded over the modem connection. System identification data were collected by exciting the system with the input profile sketched in Figure 5.10. The only measurable state of the

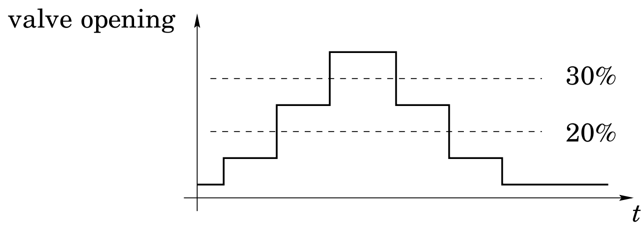

**Figure 5.10** Input profile for the system identification. Valve opening in % as a function of time.

system is the outlet air temperature. During normal use the reference temperature varies between 17 and 22 degrees centigrade. The goal of the experiment was also to be able to handle the start-up phase from considerable lower temperatures. Therefore the transfer function was estimated at a much lower temperature as well.

The actual modeling was done by fitting step responses of several predefined low order models to the process data. Then the parameters in the predefined models were optimized to match the input-output data.

The identification data was separated into several sets where each set covered a set-point. In fact there were actually two sets for each set-point, one for raising the temperature and one for lowering the temperature. There was no significant difference in the models estimated from raising or lowering the temperature or at different set-points. Basically the only difference was in the static gain. The identification experiments gave that

$$G(s) = \frac{b}{(s+a_1)(s+a_2)} = \frac{0.03}{(s+0.01)(s+0.05)}, \tag{5.6}$$

was a reasonable model of the system. During the experiments the parameter $b$ had a variation of $\pm 20\%$. At least part of the variation in this parameter is due to the outdoor temperature changes. To raise the temperature more energy needs to be added to the outdoor air if it is colder.

The chosen model is complicated enough to capture the dynamics of the system and still simple enough so it is possible to get an analytical solution to the time-optimal control problem.

***A state space representation*** There is some freedom in choosing a state space representation for the transfer function in Equation 5.6. On the real process only one state, the temperature, is measurable and to simplify the filtering to be used later on in the real-time implementation, the second state was chosen as the derivative of the temperature. A, for simulation and implementation, suitable state-space representation is thus

$$\begin{aligned}\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} &= \begin{bmatrix}0 & 1\\ -a_1a_2 & -(a_1+a_2)\end{bmatrix} x + \begin{bmatrix}0\\ b\end{bmatrix} u\\ y &= [\,1 \quad 0\,]\,x.\end{aligned} \tag{5.7}$$

The parameters $a_1, a_2$ are almost constant and only the gain $b$ contributes to the process variations as a function of the working conditions.

## Controller design

The sub-controllers are again a PID controller and a time-optimal controller. Most of the work done for the double-tank controller could be re-used.

***PID control*** The PID controller parameters were derived with a pole placement design method. The closed loop dynamics were chosen to

$$(s+\alpha\omega)(s^2+2\zeta\omega s+\omega^2). \tag{5.8}$$

The parameters in the closed loop characteristic equation and their corresponding PID parameters were

$$\begin{aligned}[\,\omega \quad \zeta \quad \alpha\,] &= [\,0.025 \quad 1.0 \quad 1.0\,]\\ [\,K \quad T_i \quad T_d\,] &= [\,0.0375 \quad 72 \quad 4.44\,].\end{aligned} \tag{5.9}$$

The controller parameters were chosen to give a well damped closed loop system but with approximately the same speed as the open system. The controller good be more aggressively tuned than the currently used PID controller as the error and control signal never are large.

***Time-optimal control*** The problem of set-point response can be viewed as a purely deterministic problem: to change the process from one state to another in shortest possible time, possibly without any overshoot subject to constraints on the control signal. The constraints are typically bounds on the control signal or its rate of change. This is an optimal control problem. For a wide class of systems the solution is of bang-bang character where the optimal control signal switches between its extreme values. For problems with a two dimensional state space the strategy can be expressed in terms of a switching curve that separates the state space in the subsets where the control signal assumes either its high or low value. This changes the control principle from feed-forward to feedback.

Equilibrium points and switching curves around those are easily calculated from the state space representation in Equation 5.7.

## Simulations

In practical applications it can be an advantage to approximate the switching curves with simpler functions. To investigate the behavior for such approximations the switching curves for System 5.7 were approximated with five straight lines. As can be seen in Figure 5.11 the overall performance did not deteriorate very much. The main difference from using the theoretically correct switching strategy is that a few extra $\langle min, max\rangle$ are needed. Simulations to test the method's robustness to process variations were also done. There were no significant decrease in performance for process parameter variations of $\pm 50\%$.

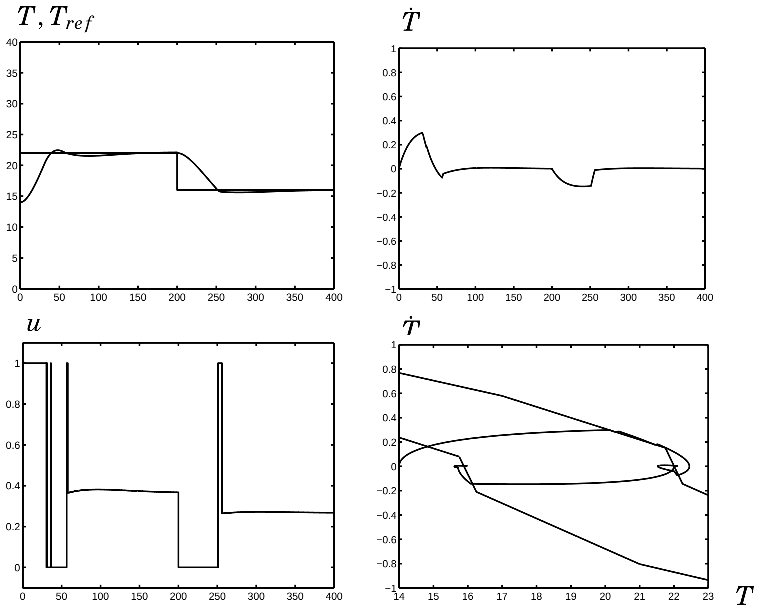

**Figure 5.11** Simulation of the ventilation model for the school in Klippan. The affine approximation of the ideal switching curves is seen together with two set-point change trajectories in the last sub-plot.

## Implementation

The hybrid controller was implemented in FORTH on a Diana ADP 2000 and tested against a process simulation using real-time SIMNON, see Elmqvist *et al.* (1990).

***Diana ADP 2000*** The control system, Diana ADP 2000, manufactured by Diana Control AB is mainly used in heating-ventilation applications. The system is flexible, modular and well-suited for testing of new software.

It is possible to install new controller code without having to rebuild the whole system for input/output handling, logging etc.

***Control system hardware*** The controller hardware Diana ADP 2000 is modular. One unit can itself have several control loops and on large installations several Diana ADP 2000 can be interconnected in a network. In the network configuration one of the control units is a master and the other are slaves. This master unit could be reached over a modem connection and from the master it is then possible to communicate with all the others. Over the telephone line and via a user interface it is for every controller possible to: log data, change parameters, and down-load new code.

***Control system software*** The programming, both the real-time operating system and the application programs, is done in FORTH. Amongst its good properties are that it is both interactive and compiled. Structure can be built with the notion of 'words'. Words are executable functions or procedures that can be arranged in a hierarchical structure. The Diana ADP 2000 comes with some predefined objects and functions such as

- Mathematical functions
- Logical functions
- Event-handling functions
- Network communications functions
- I/O-handling functions
- Predefined objects: controllers, timers etc
- Alarm functions

## Field test

Finally the controller was tested at the school in Klippan. The results are shown in Figure 5.12. As were expected there was a few extra $\langle min, max \rangle$ switches due to approximation of the switching curve and probably also due to unmodeled dynamics and nonlinearities. The $b$ parameter were estimated from steady state values on the air temperature and the control valve position.

A similar set-point change for the controller currently in use at the school showed that it took approximately twice as long time to reach the new set point. However, the main advantage with the proposed method is that it will not give a large overshoot even if started from a very low temperature.

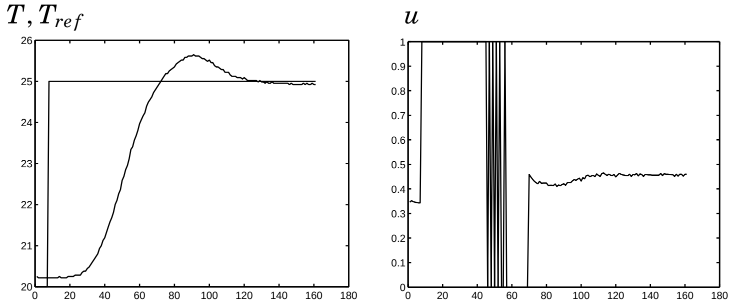

**Figure 5.12** A 5 degree set-point change using the hybrid controller. Air temperature (left) and control signal (right).

## Summary

The simple hybrid controller was tested on a fast set-point problem on a school. Simulations indicated that the theoretically derived switching curves could be approximated with simple functions without to much performance deterioration.

Measurements at the school showed that the process variations were not significant and the variation in steady state gain could be estimated from input output data before or during a set-point change.

A fairly crude model of the system together with approximative switching curves gave significantly improvements for set-point changes. The speed was doubled without large overshoots.

At this first test, no attempts were made to smooth the control signal. The extra $\langle min, max\rangle$ switches could have been avoided with a hysteresis function.

# 6

# Concluding remarks

There is currently a large interest in hybrid systems. This is motivated both by applications and by intellectual curiosity. The problems are challenging because they lead to models that are a mixture of continuous and discrete phenomena. Traditionally, the analysis of mixed systems has been either to discretize the continuous dynamics or continualize the discrete dynamics. The development of hybrid systems offers hope that complex systems of mixed discrete and continuous dynamics can be dealt with as they are. This is very attractive since then more detailed and accurate models of the real world can be analyzed directly without ad hoc approximations.

Mixing discrete and continuous dynamics can lead to very complex behavior. Often the complexity of the problem forces researchers to focus one one aspect at a time, e.g. modeling, analysis, simulation, implementation or design. For hybrid systems it may be impossible to achieve complete deterministic control. The control system designer would perhaps be able to develop a controller that ensures that all dynamics are driven to some local region, but he may not be able to specify how the system will execute the control action. It may be that the control, except for the final local control, must be done qualitatively. Some of the present hierarchical design methods use this philosophy.

Hybrid system problems require a multidisciplinary approach. Mapping control problem to automata and using graph theoretical methods to construct control algorithms can be mixed with more traditional methods for analysis of control systems. Hybrid system models soon grow out of hand and must be analyzed with computer tools. A big problem in hybrid control is that the analyzing and verifying tools do not yet match the complexity of the theory. Simulation is used heavily but it is not reliable because most simulation packages are not designed to handle hybrid systems.

Thus, the price to pay for using hybrid systems is complexity and difficult analysis. Is there then something useful in this for a control engineer that makes it worth the effort? Based on the work I have done in this thesis I believe that much can be gained. Even if my investigations are restricted to problems of limited complexity it gives some interesting views on hybrid control. One example, the simple controller that combines a fast set-point response with good regulation, has been taken all the way from conceptual design via analysis to practical implementation.

***Design:*** A systematic methodology for design of hybrid systems is highly desirable. In this thesis I have developed a methodology where conventional control designs are used in separate regions. The controllers and the design methodologies used in the different regions can be different as long as each controller is associated with a Lyapunov function. The global objectives can then be met by switching between the controllers.

***Fast mode changes:*** Dynamics are not well defined if the control design methods lead to fast mode changes. The dynamics depend on the salient features of the implementation of the mode switch. More detailed information could be needed to get a unique solution. The switching problem is not well understood. In this this thesis I derive a theorem for the stability of second order switching together with the resulting dynamics. Another switching problem that I solve is switching with two relays that work on different time scales.

***Simulation:*** The current simulation packages have problems modeling and simulating hybrid systems. One of the problems is fast mode changes. In Chapter 3, I show how possible fast mode changes can be found before or during simulation. The necessary analysis work is a very small overhead for a modern simulation tool. The problem of choosing suitable dynamics and extending the simulation model for such cases can be solved semi-automatically.

***Experiments:*** To get some experience from practical problems with hybrid control the switching strategy is implemented in two different software environments. The attempt to add a time-optimal controller to an existing PID controller on a commercial controller was successful.

To further test the method, approximations of the theoretically derived controller strategy was used to check if a simpler implementation of the switching strategy could be used while maintaining good performance. Approximations of the switching curve with a few straight lines proved sufficient even in the case of model uncertainties.

## Future work

The field of hybrid control is vast and still in its infancy and researchers have only investigated small parts of it so far. For hybrid control to be really useful in practice it is absolutely necessary that simulation and analyzing tools get better to support the control engineer.

The main issue as I see it is to find modeling methods that scale up to large systems while avoiding a combinatorial increase of complexity.

# A

# Proof of Theorem 2.1

The basic idea is that $\mathcal{S}_{xy} = \{x = 0, y = 0\}$ is a locally stable subspace if the series $\rho_k$ is decreasing, where $\rho_k$ are the intersections with the $y = 0$ plane for $x > 0$, see Figure A.1.

The analysis was inspired by Filippov (1988) pp. 234-238, where a similar analysis for a two dimensional case is done. The following analysis allows additional dynamics through the variables in $z$. The additional directions $z \in \mathbb{R}^{n-2}$ are not shown in Figure A.1.

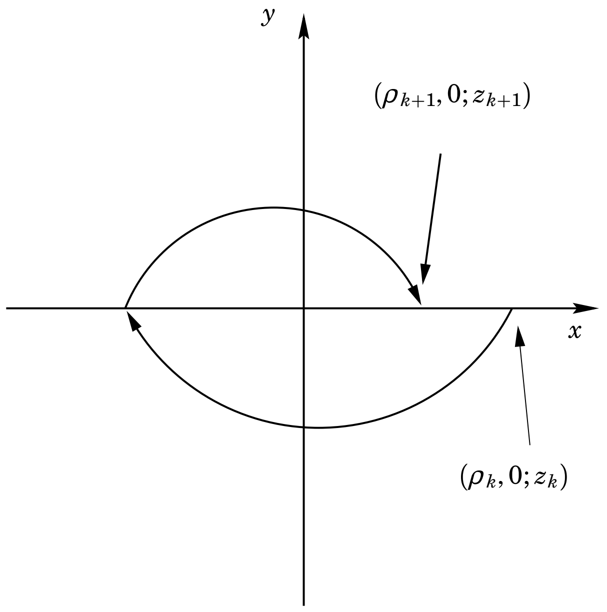

**Figure A.1** The $x$-$y$-plane. The $z$-directions, $z \in \mathbb{R}^{n-2}$, are omitted for simplicity. The intersections with the $y = 0$ plane for $x > 0$ are denoted $\rho_k$.

Straightforward series expansions of $x(t), y(t), z(t), P(x,y;z), Q(x,y;z)$ and $R(x,y;z)$ around $(0,0;z)$ show that a trajectory that starts in $(-\rho,0;z)$ hits the plane $y=0$ in $(\rho + A^{+}\rho^2 + O(\rho^3), 0, z + O(\rho))$. Following the trajectory backwards in time from $(-\rho,0;z)$ shows that the previous intersection with the $y=0$ plane is at $(\rho + A^{-}\rho^2 + O(\rho^3), 0, z + O(\rho))$.

In the calculation below it is assumed that a state transformation has been done that achieves $Q(0,0;z) \equiv 0, \forall z$, so also $Q_z(0,0;z) = 0$ and $Q_{zz}(0,0;z) = 0$. In the following equations all higher order terms are omitted as soon as they no longer are needed. The dynamics on each side of the plane $y=0$ are

$$\begin{aligned} \dot{x} &= P^{\pm}(x,y;z) \\ \dot{y} &= Q^{\pm}(x,y;z) \\ \dot{z} &= R^{\pm}(x,y;z). \end{aligned} \tag{A.1}$$

First series expansions of $x(t,\rho)$, $y(t,\rho)$, $z(t,\rho)$ and $\tilde{t}(\rho)$ are done, where $\tilde{t}(\rho)$ is the time to the next intersection with $y=0$. The index of the constants $x_{ij}$ are chosen such that the index $i$ indicates order in $t$ and the index $j$ indicates order in $\rho$.

$$\begin{aligned} x &= -\rho + t(x_{10} + x_{11}\rho + O(\rho^2)) + t^2(x_{20} + O(\rho)) + O(t^3) \\ y &= t(y_{11}\rho + y_{12}\rho^2 + O(\rho^3)) + t^2(y_{20} + y_{21}\rho + O(\rho^2)) + t^3(y_{30} + O(\rho)) \\ z &= t(z_{10} + z_{11}\rho + O(\rho^2)) + t^2(z_{20} + O(\rho)) + O(t^3) \\ \tilde{t} &= t_1\rho + t_2\rho^2 + O(\rho^3). \end{aligned} \tag{A.2}$$

Now differentiate with respect to time and identify the coefficients in terms of partial derivatives. Up to second order terms have been used for $x$ and $z$. Third order terms are needed for the expansion of $y$. It is assumed that $t_1$ above is nonzero and thus that $\rho$ and $t$ are of the same order of magnitude.

$$\begin{aligned} \dot{x} =& x_{10} + x_{11}\rho + 2tx_{20} = P + P_x(-\rho + tx_{10}) + P_y \cdot 0 + P_z t z_{10} \\ \dot{z} =& z_{10} + z_{11}\rho + 2tz_{20} = R + R_x(-\rho + tx_{10}) + R_y \cdot 0 + R_z t z_{10} \\ \dot{y} =& y_{11}\rho + y_{12}\rho^2 + 2ty_{20} + 2ty_{21}\rho + 3t^2 y_{30} \\ =& Q + Q_x(-\rho + t(x_{10} + x_{11}\rho + O(\rho^2)) + t^2 x_{20}) + Q_y(t y_{11}\rho + t^2 y_{20}) \\ &+ \frac{1}{2}(-\rho + tx_{10})Q_{xx}(-\rho + tx_{10}) + (-\rho + tx_{10})Q_{xy} \cdot 0 \\ &+ (-\rho + tx_{10})Q_{xz}(tz_{10}) + \frac{1}{2}0 \cdot Q_{yy} \cdot 0 + 0 \cdot Q_{yz}(tz_{10}). \end{aligned} \tag{A.3}$$

This gives the following equations where the right hand sides should be evaluated at $(0,0^+;z)$. The $x_{ij}$ constants are

$$\begin{aligned} 1: &\quad x_{10} = P \\ \rho: &\quad x_{11} = -P_x \\ t: &\quad 2x_{20} = P_x x_{10} + P_z z_{10} = P_x P + P_z R. \end{aligned} \tag{A.4}$$

The $z_{ij}$ constants are

$$\begin{aligned} 1: &\ z_{10} = R \\ \rho: &\ z_{11} = -R_x \\ t: &2z_{20} = R_x x_{10} + R_z z_{10} = R_x P + R_z R, \end{aligned} \tag{A.5}$$

and finally the $y_{ij}$ constants are

$$\begin{aligned} 1: &\quad 0 = Q \\ \rho: &\quad y_{11} = -Q_x \\ \rho^2: &\quad y_{12} = \frac{1}{2}Q_{xx} \\ t: &\quad 2y_{20} = Q_x x_{10} = Q_x P \\ t\rho: &\quad 2y_{21} = Q_x x_{11} + Q_y y_{11} - Q_{xx} x_{10} - Q_{xz} z_{10} \\ &\qquad = -Q_x P_x - Q_y Q_x - Q_{xx} P - Q_{xz} R \\ t^2: &\quad 3y_{30} = Q_x x_{20} + Q_y y_{20} + \frac{1}{2} x_{10} Q_{xx} x_{10} + x_{10} Q_{xz} z_{10} \\ &\qquad = \frac{1}{2} Q_x (P_x P + P_z R) + \frac{1}{2} Q_y Q_x P + \frac{1}{2} P Q_{xx} P + P Q_{xz} R. \end{aligned} \tag{A.6}$$

Now look for the $\tilde{t}$ that solves $y(\tilde{t}) = 0$.

$$\begin{aligned} y(\tilde{t}) =& [t_1\rho + t_2\rho^2 + O(\rho^3)] \cdot [y_{11}\rho + y_{12}\rho^2 + O(\rho^3)] \\ &+ [t_1\rho + t_2\rho^2 + O(\rho^3)]^2 \cdot [y_{20} + y_{21}\rho + O(\rho^2)] \\ &+ [t_1\rho + t_2\rho^2 + O(\rho^3)]^3 \cdot [y_{30} + O(\rho)] = 0 \end{aligned} \tag{A.7}$$

This gives a solution for $t_1$ and $t_2$.

$$\begin{aligned} \rho^2: &\quad t_1 y_{11} + t_1^2 y_{20} = 0 \\ \rho^3: &\quad t_2 y_{11} + t_1 y_{12} + t_1^2 y_{21} + t_1^3 y_{30} + 2t_1 t_2 y_{20} = 0. \end{aligned} \tag{A.8}$$

Now $t_1$ and $t_2$ can be calculated in terms of $P$, $Q$, $R$ and partial derivatives.

$$\begin{aligned}
t_1 &= -\frac{y_{11}}{y_{20}} = \frac{2Q_x}{Q_xP} = \frac{2}{P} \\
t_2 &= -\frac{t_1y_{12} + t_1^2y_{21} + t_1^3y_{30}}{y_{11} + 2t_1y_{20}} \\
&= -\frac{1}{-Q_x + 2\frac{2}{P}\frac{1}{2}Q_xP}[\frac{2}{P}\frac{1}{2}Q_{xx} + (\frac{2}{P})^2\frac{1}{2}(-Q_xP_x - Q_yQ_x - Q_{xx}P - Q_{xz}R) \\
&\quad + (\frac{2}{P})^3\frac{1}{6}(Q_x(P_xP + P_zR) + Q_yQ_xP + PQ_{xx}P + 2PQ_{xz}R)] \\
&= \frac{-1}{Q_x}[\frac{Q_{xx}}{P} + \frac{2}{P^2}(-Q_xP_x - Q_yQ_x - Q_{xx}P - Q_{xz}R) \\
&\quad + \frac{4}{3P^3}(Q_x(P_xP + P_zR) + Q_yQ_xP + PQ_{xx}P + 2PQ_{xz}R)]. \qquad (A.9)
\end{aligned}$$

Evaluate $x$ at $\tilde{t}$ to find the next $\rho$:

$$\begin{aligned}
x(\tilde{t}) &= -\rho + \tilde{t}(x_{10} + x_{11}\rho) + \tilde{t}^2x_{20} \\
&= -\rho + (\frac{2}{P}\rho + t_2\rho^2)(x_{10} + x_{11}\rho) + (\frac{2}{P}\rho + t_2\rho^2)^2x_{20} \\
&= -\rho + (\frac{2}{P}\rho + t_2\rho^2)(P - P_x\rho) + (\frac{2}{P}\rho + t_2\rho^2)^2\frac{1}{2}(P_xP + P_zR) \\
&= \rho + \rho^2(t_2P + \frac{2P_zR}{P^2}). \qquad (A.10)
\end{aligned}$$

With the expression for $t_2$ above, this finally results in:

$$x(\tilde{t}) = \rho + \rho^2\frac{2}{3}[\frac{P_x + Q_y}{P} - \frac{Q_{xx}}{2Q_x} + \frac{(Q_xP_z - PQ_{xz})R}{Q_xP^2}] = \rho + \rho^2 \cdot \frac{2}{3}A,$$

where the functions in A are evaluated at $(0, 0^+, z)$. This can be compared with Filippov's result in Filippov (1988) for the two dimensional case, i.e. without the $z$ dynamics:

$$A = [\frac{P_x + Q_y}{P} - \frac{Q_{xx}}{2Q_x}].$$

If the above calculations are repeated with $(\rho, 0^-; z)$ as a starting point then it is seen that the next passage at $y = 0$ is made at the point $(-\rho + \frac{2}{3}A\rho^2, 0; z)$. This calculation is equivalent to following a trajectory backwards in time from $(-\rho, 0; z)$ to $(\rho - \frac{2}{3}A\rho^2, 0; z)$. Introduce $A^+$ and $A^-$ for the value of $A$ when $A$ is evaluated on each side of $y = 0$,

$$\begin{aligned}
A^+ &= A(0, 0^+; z) \\
A^- &= A(0, 0^-; z).
\end{aligned}$$

Finally to investigate stability consider

$$\begin{aligned} \rho &\overset{f}{\rightarrow} -\rho + \frac{2}{3}A^- \cdot \rho^2 + O(\rho^3) \\ -\rho &\overset{g}{\rightarrow} \rho + \frac{2}{3}A^+ \cdot \rho^2 + O(\rho^3). \end{aligned}$$

The iterative equation for the intersection with $y = 0, x > 0$ are thus

$$\rho_{k+1} = (g \circ f)(\rho_k) = \rho_k + \frac{2}{3}\rho_k^2 \cdot (A^+ - A^-) + O(\rho_k^3).$$

The sliding dynamics on the subspace $\mathcal{S}_{yx}$ are defined by:

$$\dot{z} = \alpha^+ R^+(0,0;z) + \alpha^- R^-(0,0;z), \tag{A.11}$$

where $\alpha^+$ and $\alpha^-$ are uniquely defined by

$$\begin{aligned} \alpha^+ + \alpha^- &= 1 \\ \alpha^+ P^+ &= \alpha^- P^-. \end{aligned} \tag{A.12}$$

Equation A.11 is a necessary condition for staying on the subspace $\mathcal{S}_{yx}$.

# B

# The chatterbox theorem

The notation 'chatterbox' was introduced in Seidman (1992) where he proves a similar theorem but uses constant vectorfields and relays with time hysteresis. The notation from Chapter 2 is used where the vector-

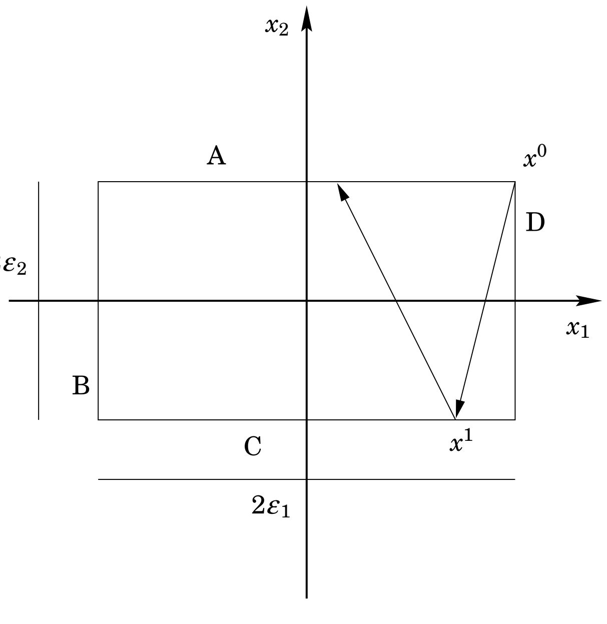

**Figure B.1** The chatterbox in the $x_1$-$x_2$-plane with $\varepsilon_2$ smaller than $\varepsilon_1$. The boundaries of the chatterbox are denoted A, B, C and D.

fields $f_1(x) \dots f_4(x)$ used in quadrants $1 \dots 4$ are nonlinear functions of $x$. Relays with hysteresis are used and switching from one vectorfield to another is done at the boundaries A, B, C and D. For simplicity only the $x_1$-$x_2$-plane is shown in Figure B.1 and as before the rest of the dynamics are collected in $x_3$. To begin with relay two is the fastest and thus $\lim_{\varepsilon_1 \to 0} \frac{\varepsilon_2}{\varepsilon_1} = 0$. The distanced traveled using any vectorfield $f_i$ is proportional to time which gives $O(t) = O(\varepsilon)$.

The Filippov convex definition of a solution on $\mathcal{S}_1^+$ is denoted $F_{14}^C$ and a corresponding notation is used for the sliding dynamics on $\mathcal{S}_1^-$, $\mathcal{S}_2^+$ and $\mathcal{S}_2^-$. With those prerequisites the following theorem can be proved

THEOREM B.1—THE CHATTERBOX THEOREM
If relays with different hysteresis $\varepsilon_1$ and $\varepsilon_2$ are used and $\varepsilon_1 \to 0$ and $\varepsilon_2/\varepsilon_1 \to 0$ then the limiting dynamics on $\mathcal{S}_{12}$ can be uniquely defined from calculating the Filippov convex solutions $F_{14}^C$, $F_{23}^C$ on $\mathcal{S}_2^{\pm}$ and then take a convex combination of $F_{14}^C$ and $F_{23}^C$ as to have the dynamics $\dot{x} = [0, 0; \dot{x}_3]^T$. □

***Proof*** Starting at $x^0 = [\varepsilon_1, \varepsilon_2]$ (the $z$ coordinates will be omitted in the equations) and using Taylor expansions gives the next intersection with the chatterbox at $x^1$, where

$$x^1 = x^0 + tf_1(x^0) + O(\varepsilon_2^2) \tag{B.1}$$

the intersection with the boundary C occurs at time

$$t_1 = \frac{-2\varepsilon_2}{f_{12}} + O(\varepsilon_2^2),$$

where $f_{12} < 0$. On this boundary the relay switches to vectorfield $f_4$. That vectorfield is used until the boundary A is hit, which occurs after additional time

$$t_4 = \frac{2\varepsilon_2}{f_{42}} + O(\varepsilon_2^2),$$

where $f_{42} > 0$. The time between two intersections with the boundary A, denoted $t_{14}$ is hence

$$t_{14} = t_1 + t_4 = 2\varepsilon_2(\frac{-1}{f_{12}} + \frac{1}{f_{42}}) + O(\varepsilon_2^2). \tag{B.2}$$

The distance traveled in the $x_1$ direction during $t_{14}$ is

$$t_1 f_{11} + t_4 f_{41} = 2\varepsilon_2(\frac{-f_{11}}{f_{12}} + \frac{f_{41}}{f_{42}}) + O(\varepsilon_2^2).$$

The boundary B is hit after a number $f_1$-$f_4$ cycles. The number of such cycles, $n_{14}$, satisfies

$$\frac{-2\varepsilon_1}{2\varepsilon_2(\frac{-f_{11}}{f_{12}}+\frac{f_{41}}{f_{42}})+O(\varepsilon_2)}-1 \leq n_{14} \leq \frac{-2\varepsilon_1}{2\varepsilon_2(\frac{-f_{11}}{f_{12}}+\frac{f_{41}}{f_{42}})+O(\varepsilon_2)}+1. \tag{B.3}$$

The velocity, $\dot{x}_1$ in the $x_1$ direction is hence within the interval

$$\frac{-2\varepsilon_1}{(n_{14}+1)t_{14}} \geq \dot{x}_1 \geq \frac{-2\varepsilon_1}{(n_{14}-1)t_{14}}. \tag{B.4}$$

From equation B.2–B.4 the upper and lower bounds on $\dot{x}_1$, become

$$\begin{aligned}\frac{-2\varepsilon_1}{(n_{14}\pm 1)t_{14}} &= \frac{-2\varepsilon_1}{2\varepsilon_1\frac{(\frac{-1}{f_{12}}+\frac{1}{f_{42}})+O(\varepsilon_2)}{(\frac{f_{11}}{f_{12}}-\frac{f_{41}}{f_{42}})+O(\varepsilon_2)} \pm 2\varepsilon_2(\frac{-1}{f_{12}}+\frac{1}{f_{42}})+O(\varepsilon_2)}\\ &= \frac{1}{\frac{f_{42}-f_{12}}{f_{11}f_{42}-f_{12}f_{41}}(1+O(\varepsilon_2)) \pm \frac{\varepsilon_2}{\varepsilon_1}(\frac{-1}{f_{12}}+\frac{1}{f_{42}})+O(\varepsilon_2^2/\varepsilon_1)}.\end{aligned}$$

Compare this with the Filippov convex combination, $F_{14}^C$, where $F_{14,1}^C$ denotes the velocity in the $x_1$ direction

$$\begin{aligned} F_{14}^C &= \alpha f_1 + (1-\alpha)f_4, \quad \alpha = \frac{f_{42}}{f_{42}-f_{12}}\\ F_{14,1}^C &= \frac{f_{11}f_{42}-f_{12}f_{41}}{f_{42}-f_{12}}. \end{aligned} \tag{B.5}$$

This shows that the velocity $\dot{x}_1$ in the negative direction can then be written as

$$\dot{x}_1^- = F_{14,1}^C + O(\varepsilon_2) + O(\varepsilon_2/\varepsilon_1). \tag{B.6}$$

A similar calculation using $f_2$ and $f_3$ for $\dot{x}_1 > 0$ gives

$$\dot{x}_1^+ = F_{23,1}^C + O(\varepsilon_2) + O(\varepsilon_2/\varepsilon_1). \tag{B.7}$$

Now, moving back and fourth in the $x_1$ direction leads to the equations

$$t_{14} = \frac{-2\varepsilon_1}{F_{14,1}^C}, \quad t_{23} = \frac{2\varepsilon_1}{F_{23,1}^C}. \tag{B.8}$$

The resulting dynamics in the $x_3$ directions can be written as

$$\dot{x}_3 = \frac{t_{14}}{t_{14}+t_{23}}\dot{x}_1^- + \frac{t_{23}}{t_{14}+t_{23}}\dot{x}_1^+ + O(\varepsilon_1) \tag{B.9}$$

$$\dot{x}_3 = \dot{x}_{1423,3} + O(\varepsilon_2) + O(\varepsilon_2/\varepsilon_1) + O(\varepsilon_1) \tag{B.10}$$

Letting $\varepsilon_1$ and $\varepsilon_2/\varepsilon_1$ goto zero gives

$$\lim_{\varepsilon_1 \to 0} \lim_{\varepsilon_2/\varepsilon_1 \to 0} \dot{x}_3 = \dot{x}^C_{1423,3}, \tag{B.11}$$

i.e the solution obtained by taking the Filippov convex combination solution first over the $(x_2 = 0)$-plane and then over the $(x_1 = 0)$-plane.

If the order is reversed, i.e letting $\varepsilon_2 \to 0$ and $\varepsilon_1/\varepsilon_2 \to 0$, then the resulting dynamics are $\dot{x}^C_{1234,3}$. In general $\dot{x}^C_{1234,3} \neq \dot{x}^C_{1423,3}$.

# C

# Pal code for the double-tank experiment

The full PAL code for the hybrid controller used for level control in the double-tank experiment is presented in this section. The controller is written as a Grafcet with four states, see Figure 5.2.

```
module regul;

  function ln(r : input real) : real;external "ln";

  function sqrt(r : input real) : real;external "sqrt";

  block Tank

    y1, y2 : input real;
    OnTarget := false, NewRef := false, Off := false : boolean;
    h : sampling interval;
    a := 1.0, b := 1.0, yref := 0.50 : real;
    K := 5.0, Ti := 60.0, Tr := 60.0, Td := 10.0 : real;
    gamma1 := 1.0, gamma2 := 1.0, Vpid := 100.0 : real;
    e := 0.0, yold := 0.0, P := 0.0, I := 0.0, D := 0.0 : real;
    d1 := 1.0, d2 := 1.0, i1 := 1.0, w1 := 1.0 : real;
    umin := 0.0, umax := 1.0 : real;
    x1 := 5.0, x2 := 5.0, xref := 5.0,, u := 0.0 : real;
    Ku := 0.000027, Kc := 5.0, region := 0.1 : parameter real;
    omega := 0.04, zeta := 0.7, alpha := 1.0, N := 10.0 : parameter real;
    c := 2.0, aa := 0.00000707, AA := 0.00273 : parameter real;
    v := 0.0 : output real;

    function Switch(z1 : input real;z3 : input real;ubar : input real : real;
    begin
      result := 1.0/a * ((a * z1 - b * ubar) * (1.0 + ln((a * z3 - b * ubar)/
                (a * z1 - b * ubar))) + b * ubar);
```

```
end Switch;

function Sat(min : input real;max : input real;x : input real ) : real;
begin
   if $x < min$ then
      $result := min$;
   elsif $x > max$ then
      $result := max$;
   else
      $result := x$;
   end if;
end Sat;

calculate
begin
   $x1 := (1.0 - c * h) * x1 + c * h * y1$;
   $x2 := (1.0 - c * h) * x2 + c * h * y2$;
   $xref := yref * Kc$;
end calculate;

update
begin
   $Vpid := gamma1 * ((x1 - xref) * (x1 - xref)+$
               $gamma2 * (x2 - xref) * (x2 - xref))$;
   $yold := x2$;
end update;

initial step Init;
   activate OffController;
end Init;

step Ref;
   activate NewControllers;
end Ref;

step Opt;
   activate OptController;
end Opt;

step PID;
   activate PIDController;
end PID;

transition from $Init$ to $Ref$ when $NewRef$;

transition from $Ref$ to $Opt$ when  not $NewRef$;

transition from $PID$ to $Init$ when $Off$;
transition from $Opt$ to $PID$ when $OnTarget$ or $Off$;

transition from $PID$ to $Ref$ when $NewRef$;

transition from $Opt$ to $Ref$ when $NewRef$;

action OffController;
begin
  $v := 0.0$;
  $Vpid := 100.0$;
  $Off :=$ false;
end OffController;

action NewControllers;
begin
  $a := aa/AA * sqrt(9.81/2.0/xref)$;
  $b := Ku/AA * Kc$;
  $K := (omega * omega * (1.0 + 2.0 * alpha * zeta) - a * a)/b/a$;
  $Ti := b * K * a/alpha/omega/omega/omega$;
  $Td := 1.0/a/b/K * (omega * (alpha + 2.0 * zeta) - 2.0 * a)$;
  $d1 := Td/(Td + N * h)$;
  $d2 := K * N * d1$;
  $i1 := h * K/Ti$;
  $w1 := h/Tr$;
  $NewRef :=$ false;
end NewController;

action OptController;
begin
  if $Vpid < region$ then
    $OnTarget :=$ true;
    $e := xref - x2$;
    $P := K * e$;
    $D := d1 * D + d2 * (yold - x2)$;
    $I := aa/AA * sqrt(2.0 * 9.81 * xref * Kc)/b$;
  end if;
  $u := 0.0$;
  if $x1 > xref$ and $x2 < Switch(x1, xref, umin)$ then
    $u := umax$;
  end if;
  if $x1 < xref$ and $x2 < Switch(x1, xref, umax)$ then
    $u := umax$;
  end if;
  $v := u$;
end OptController;
```

```
    **action** PIDController;
      $usat$ : **real**;
    **begin**
      $OnTarget$ := **false**;
      $e := xref - x2$;
      $P := K * e$;
      $D := d1 * D + d2 * (yold - x2)$;
      $u := P + I + D$;
      $usat := Sat(umin, umax, u)$;
      $v := u$;
      $I := I + i1 * e + w1 * (usat - u)$;
    **end** PIDController;

    **procedure** Ref05();
    **begin**
      $yref := 0.05$;
      $NewRef$ := **true**;
    **end** Ref10;

    **procedure** Ref15();
    **begin**
      $yref := 0.11$;
      $NewRef$ := **true**;
    **end** Ref11;

    **procedure** Stop();
    **begin**
      $Off$ := **true**;
    **end** Stop;

  **end** Tank;

**end** regul.
```

# Bibliography

Alur, R., C. Courcoubetis, T. A. Henzinger, and P.-H. Ho (1993): "Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems." Technical Report TR93-1343. Cornell University, Computer Science Department.

Alur, R. and D. L. Dill (1994): "A theory of timed automata." *Theoretical Computer Science*, **126:2**, pp. 183–235.

Andersson, M. (1994): *Object-Oriented Modeling and Simulation of Hybrid Systems*. PhD thesis ISRN LUTFD2/TFRT--1043--SE.

Antsaklis, P. J. (1997): "Hybrid systems IV: Papers related to the fourth international conference on hybrid systems, held in ithaca, N.Y, October 12–14, 1996."

Antsaklis, P. J., K. M. Passino, and S. J. Wang (1991): "An introduction to autonomous control systems." *IEEE Control Systems Magazine*, **11:4**, pp. 5–13.

Åström, K. (1968): *Lectures on Nonlinear Systems*, chapter 3. In Swedish.

Åström, K. and K. Furuta (1996): "Swinging up a pendulum by energy control." In *IFAC World Congress*. San Fransisco.

Åström, K. J. (1992): "Autonomous control." In Bensoussan and Verjus, Eds., *Future Tendencies in Computer Science, Control and Applied Mathematics*, vol. 653 of *Lecture Notes in Computer Science*, pp. 267–278. Springer-Verlag.

Åström, K. J. and K.-E. Årzén (1993): "Expert control." In Antsaklis and Passino, Eds., *An Introduction to Intelligent and Autonomous Control*, pp. 163–189. Kluwer Academic Publishers.

Åström, K. J. and T. Hägglund (1995): *PID Controllers: Theory, Design, and Tuning*, second edition. Instrument Society of America, Research Triangle Park, NC.

Åström, K. J. and B. Wittenmark (1995): *Adaptive Control*, second edition. Addison-Wesley, Reading, Massachusetts.

Blomdell, A. (1997): "The Pålsjö algorithm language.". Master's thesis, Department of Automatic Control, Lund Institute of Technology.

Bode, H. (1945): *Network Analysis and Feedback Amplifier Design*. D. van Nostrand, New York.

Branicky, M. (1995): *Studies in Hybrid Systems: Modeling, Analysis, and Control*. PhD thesis, LIDS, Massachusetts Institute of Technology.

Brenan, K. E., S. L. Campbell, and L. R. Petzold (1989): *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. North-Holland, New York.

Brockett, R. W. (1983): "Asymptotic stability and feedback stabilization." In Brockett *et al.*, Eds., *Differential Geometric Control Theory*, pp. 181–191.

Brockett, R. W. (1990): "Formal languages for motion description and map making." In Brocket, Ed., *Robotics*. American Math. Soc., Providence, Rhode Island.

Brockett, R. W. (1993): "Hybrid models for motion control systems." In Trentelman and Willems, Eds., *Essays on Control: Perspectives in the Theory and its Application*. Birkhäuser.

Brooks, R. A. (1990): "Elephants don't play chess." In Maes, Ed., *Designing Autonomous Agents*. MIT Press, Cambridge, Massachusetts.

Caines, P. and R. Ortega (1994): "The semi-lattice of piecewise constant controls for non-linear systems: A possible foundation for fuzzy control." In *NOLCOS 95*. Tahoe, CA.

Caines, P. and Y.-J. Wei (1995): "Hierarchical hybrid control systems." In *Block Island Workshop*. Rhode Island.

Cassandras, C. G. (1993): *Discrete Event Systems: Modeling and Performance Systems*. Irwin.

Clarke, F. H. (1990): *Optimization and Nonsmooth Analysis*. SIAM.

David, R. and H. Alla (1992): *Petri Nets and Grafcet: Tools for modelling discrete events systems*. Prentice-Hall.

Daws, C., A. Olivero, S. Tripakis, and S. Yovine (1996): "The tool KRONOS." In Alur *et al.*, Eds., *Hybrid Systems III*, vol. 1066 of *Lecture Notes in Computer Science*, pp. 208–219. Springer-Verlag.

Duff, I. S., A. M. Erisman, and J. K. Reid (1990): *Direct methods for sparse matrices.* Clarendon Press, Oxford.

Eker, J. (1997): *A Framework for Dynamically Configurable Embedded Controllers.* Lic Tech thesis ISRN LUTFD2/TFRT--3218--SE.

Eker, J. and A. Blomdell (1996): "A structured interactive approach to embedded control." In *4th Intelligent Robotic System*, pp. 191–197. Lisbon, Portugal.

Elmqvist, H., K. J. Åström, T. Schönthal, and B. Wittenmark (1990): *Simnon User's Guide.* SSPA, Göteborg, Sweden.

Emelyanov, S. V. (1967): *Variable Structure Control Systems.* Oldenburger Verlag, Munich, FRG.

Ezzine, J. and A. H. Haddad (1989): "Controllability and observability of hybrid systems." *Int. J. Control,* **49:6**, pp. 2045–2055.

Ferron, E. (1996): "Quadratic stabilizability of switched systems via state and output feedback." Technical Report CICS-P-468. Center for intelligent control systems, MIT, Cambridge, Massachusetts, 02139, U.S.A.

Feuer, A., G. G. C., and S. M.. (1997): "Potential benefits of hybrid control for linear time invariant plants." In *Proceedings of the 1997 American Control Conference.* Albuquerque, New Mexico, USA.

Filippov, A. F. (1988): *Differential Equations with Discontinuous Right-hand Sides.* Kluwer, Dordrecht.

Halbwachs, N. (1993): *Synchronous programming of reactive systems.* Kluwer Academic Pub.

Henzinger, T. A., P. W. Kopke, A. Puri, and P. Varaiya (1995a): "What's decidable about hybrid automata?" Technical Report TR95-1541. Cornell University, Computer Science.

Henzinger, T. H., P.-H. Ho, and H. Wong-Toi (1995b): "A user guide to HyTech." *Lecture Notes in Computer Science,* **1019**.

Itkis, U. (1976): *Control Systems of Variable Structure.* Halsted Press, Wiley, New York.

Jantzen, J. (1994): *Digraph Approach to Multivariable Control.* Electrical Power Engineering department, Technical University of Denmark, Lyngby, Denmark.

Johansson, K. H. and A. Rantzer (1996): "Global analysis of third-order relay feedback systems." Report ISRN LUTFD2/TFRT--7542--SE.

Johansson, M., J. Malmborg, A. Rantzer, B. Berhardsson, and K.-E. Årzén (1997): "Modeling and control of fuzzy, heterogeneous and hybrid systems." In *Proceedings of the SiCiCa 1997*.

Johansson, M. and A. Rantzer (1997): "On the computation of piecewise quadratic Lyapunov functions." In *Proceedings of the 36th IEEE Conference on Decision and Control*. San Diego, USA.

Johansson, M. and A. Rantzer (1998): "Computation of piecewise quadratic Lyapunov functions for hybrid systems." *IEEE Transactions on Automatic Control*, April. Special issue on Hybrid Systems. To Appear.

Khalil, H. K. (1992): *Nonlinear Systems*. Macmillan, New York.

Kiendl, H. and J. J. Rüger (1995): "stability analysis of fuzzy control systems using facet functions." *Fuzzy Sets and Systems*, **70**, pp. 275–285.

Kuipers, B. and K. J. Åström (1991): "The composition of heterogeneous control laws." In *Proceedings of the 1991 American Control Conference*, pp. 630–636. Boston, Massachusetts.

Lafferiere, G. (1994): "Discontinuous stabilizing feedback using partially defined Lyapunov functions." In *Conference on Decision and Control*. Lake Buena Vista.

Lafferiere, G. and S. E. (1993): "Remarks on control Lyapunov functions for discontinuous stabilizing feedback." In *Conference on Decision and Control*. Texas.

Leitman, G. (1981): *The Calculus of Variations and Optimal Control*. Plenum Press, New York.

Lewis, F. L. (1986): *Optimal Control*. Wiley.

Malmborg, J., B. Berhardsson, and K. J. Åström (1996): "A stablizing switching scheme for multi-controller systems." In *Proceedings of the 1996 Triennal IFAC World Congress, IFAC'96*, vol. F, pp. 229–234. Elsevier Science, San Francisco, California, USA.

Malmborg, J. and B. Bernhardsson (1997): "Control and simulation of hybrid systems." *Nonlinear Analysis, Theory, Methods and Applications*, **30:1**, pp. 337–347.

Malmborg, J. and J. Eker (1997): "Hybrid control of a double tank system." In *IEEE Conference on Control Applications*. Hartford, Connecticut.

Malmborg, J. and M. Johansson (1994): "Fuzzy heterogeneous control." In *FALCON Meeting/EUFIT Conference*. Aachen.

MATLAB (1995): *MATLAB SIMULINK, toolboxes*. The Mathworks, Cochituate Place, 24 Prime Park Way, Natick, MA, USA, version 4.2, volume vi edition. 1 computer laser optical disk, graphics quick reference + 1 MATLAB quick reference.

Mattsson, S. and H. Elmqvist (1997): "Modelica – an international effort to design the next generation modeling language." In *IFAC Symp. on Computer Aided Control Systems Design, CACSD'97*. Gent, Belgium.

Mattsson, S. and G. Söderlind (1993): "Index reduction in differential-algebraic equations." *SIAM Journal of Scientific and Statistical Computing*, **14:3**, pp. 677–692.

Mattsson, S. E. (1996): "On object-oriented modeling of relays and sliding mode behaviour." In *IFAC'96, Preprints 13th World Congress of IFAC*, vol. F, pp. 259–264. San Francisco, California.

Middleton, R. H. (1991): "Trade-offs in linear control systems design." *Automatica.*, **27:2**, pp. 281–292.

Morse, A. (1995a): "Control using logic-based switching." In *Block Island Workshop*. Rhode Island.

Morse, A. S. (1995b): "Control using logic-based switching." In Isidori, Ed., *Trends in Control. A European Perspective*, pp. 69–113. Springer.

Nerode, A. and W. Kohn (1992): "Models for hybrid systems: automata, topologies, stability." Technical Report. Mathematical Sciences Institute, Cornell University.

Nijmeijer, H. and A. van der Schaft (1990): *Nonlinear Dynamical Control Systems*. Springer-Verlag.

Otter, M. and F. E. Cellier (1995): "Software for modeling and simulating control systems."

Paden, B. and S. Sastry (1987): "A calculus for computing Filippov's differential inclusion with application to the variable structure control of robot manipulators." *IEEE Trans. on Circuits and Systems*, **34:1**, pp. 73–82.

Pantelides, C. C. (1988): "The consistent initialization of differential-algebraic systems." *Siam J. of Scientific and Statistical Computing*, **9:2**, pp. 213–231.

Peleties, P. and R. DeCarlo (1989): "A modeling strategy with event structures for hybrid systems." In *Proc. 28th CDC*, pp. 1308–1313. Tampa, FL.

Ramadge, P. J. G. and W. M. Wonham (1989): "The control of discrete event systems." *Proceedings of the IEEE; Special issue on Dynamics of Discrete Event Systems*, **77, 1**, pp. 81–98.

Seidman, T. I. (1992): "Some limit results for relays." In *WCNA 92.* Tampa, USA.

Shevitz, D. and B. Paden (1994): "Lyapunov stability theory of nonsmooth systems." *IEEE Trans. on Automat. Contr.*, **39:9**, pp. 1910–1914.

Sontag, E. (1981): "Nonlinear regulation: The piecewise linear approach." *IEEE Transactions on Automatic Control.*

Stiver, J. and P. Antsaklis (1992): "Modeling and analysis of hybrid control systems." In *Conference on Decision and Control.* Tucson, Arizona, USA.

Sugeno, M. and T. Takagi (1983): "Multi-dimensional fuzzy reasoning." *Fuzzy Sets and Systems*, **9:3**, pp. 313–325.

Tarjan, R. H. (1972): "Depth first search and linear graph algorithms." *Siam J. of Comp.*, **1**, pp. 146–160.

Tavernini, L. (1987): "Differential automata and their discrete simulators." *Nonlinear Analysis, Theory, Methods and Applications*, **11:6**, pp. 665–683.

Tittus, M. (1995): *Control Synthesis for batch Processes.* PhD thesis, Chalmers University of Technology.

Utkin, V. (1978): *Sliding Mode and Their Application in Variable Structure Systems.* MIR Publishers, Moscow.

Utkin, V. I. (1977): "Variable structure systems with sliding modes." *IEEE Transactions on Automatic Control*, **AC-22**, pp. 212–222.

ISSN 0280–5316
ISRN LUTFD2/TFRT--1050--SE