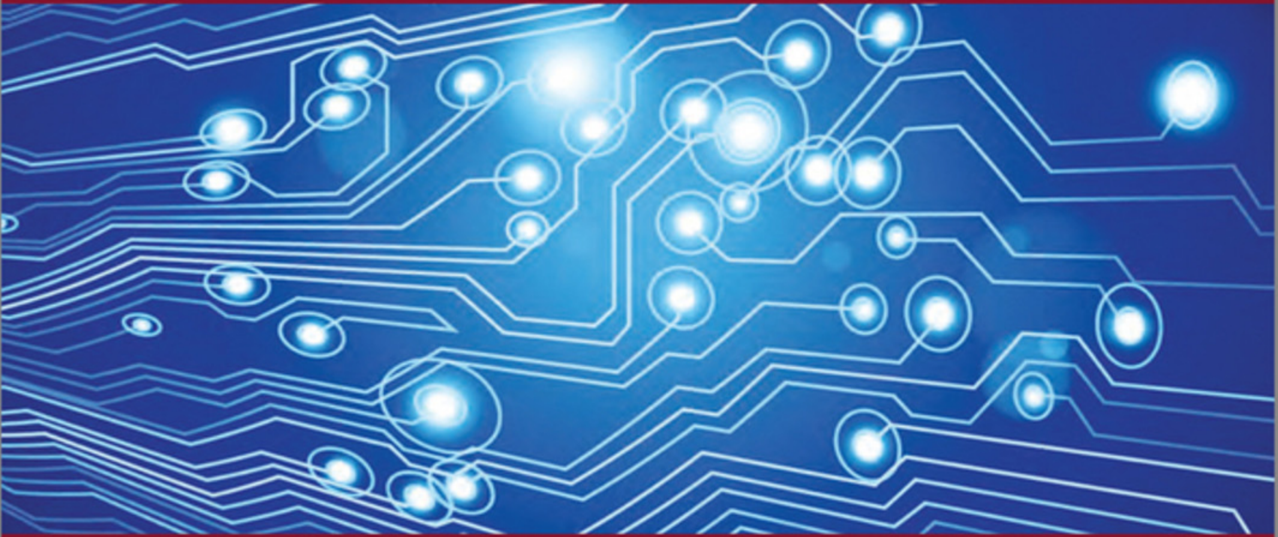
FOCUS

SYSTEMS ENGINEERING SERIES



Control of Switching Systems by Invariance Analysis

Application to Power Electronics

Laurent Fribourg and Romain Soulat

iSTE

WILEY

Control of Switching Systems by Invariance Analysis

FOCUS SERIES

Series Editor Bernard Dubuisson

Control of Switching Systems by Invariance Analysis

Application to Power Electronics

Laurent Fribourg
Romain Soulat

WILEY

First published 2013 in Great Britain and the United States by ISTE Ltd and John Wiley & Sons, Inc.

ISTE Ltd
27-37 St George's Road
London SW19 4EU
UK

www.iste.co.uk

John Wiley & Sons, Inc.
111 River Street
Hoboken, NJ 07030
USA

www.wiley.com

Library of Congress Control Number: 2013941774

British Library Cataloguing-in-Publication Data
A CIP record for this book is available from the British Library
ISSN: 2051-2481 (Print)
ISSN: 2051-249X (Online)
ISBN: 978-1-84821-606-8

FSC
www.fsc.org
MIX
Paper from
responsible sources
FSC® C013604

Printed and bound in Great Britain by CPI Group (UK) Ltd., Croydon, Surrey CR0 4YY

Contents

Preface

Switched systems are embedded devices widespread in industrial applications such as power electronics and automotive control. They consist of continuous-time dynamical subsystems and a rule that controls the switching between them. Under a suitable control rule, the system can improve its steady-state performance and meet essential properties, such as safety and stability, in desirable operating zones. We show in this book that such controller synthesis problems are related to the construction of appropriate invariants of the state space, which approximate the limit sets of the system trajectories. We present several approaches of invariant construction based on techniques of state space decomposition and backward/forward fixed-point computation, and perform them directly on the continuous state space, or indirectly on discrete abstractions. All these methods are illustrated in a number of case studies, mainly taken from the field of power electronics.

Acknowledgments

Most of this work was produced in the framework of projects BOOST and BOOST2, which are cooperations between Systems et Application des Technologies de l'Information et de l'Energie (SATIE) Electronics Laboratory and Laboratoire Specification et Verification (LSV) Computer Science Laboratory at ENS Cachan. We are grateful to Institut Farman, ENS Cachan, and Centre National de la Recherche Scientifique (CNRS) for supporting these interdisciplinary projects.

Bertrand Revol (SATIE) was the co-leader of projects BOOST and BOOST2 and initiated the cooperation with LSV by proposing concrete case studies of control of devices of power electronics. Chapter 5 is based on technical reports [FEL 12a, FEL 12b], which have been written in collaboration of Revol and his colleagues Gilles Feld, Denis Labrousse and Stéphane Lefebvre.

When coming to the design and implementation of the tool of decomposition (Appendix 5), Ulrich Kühne implemented an interface with PPL [PPL 13] which allows us to calculate a sequence of nested invariants, and visualize it as an animation.

Finally, we are most grateful to Étienne André for his very helpful comments on an earlier draft of the manuscript.

Introduction

In recent years, there has been an increasing interest in applying renewable energy in electricity generation and transportation. In particular, much effort has been devoted to the improvement of robust and flexible control techniques of power converters in order to increase reliability and safety of operation. Due to their practical feasibility to achieve a high performance as well as natural digital implementation in signal processors, *switched controllers* are the most common type of controller to have been applied to power converters (see [CER 09, LIB 03, SUN 05]). Systems equipped with switched controllers are constituted of two parts: first, a family of continuous subsystems or *modes*; second, a *switching signal* that controls the selection of these modes. The switching signal can be state dependent and/or time dependent.

With respect to classical systems, an interest of switching controllers arises from the existence of systems that cannot be asymptotically stabilized by a single continuous feedback control law [BRO 83]. However, with switched systems, the steady-state operating condition is typically a *periodic solution* or *limit cycle*, not an equilibrium point. The relevant stability notion is *asymptotic orbital stability* or *practical stability*, which studies the conditions under which the system state evolves within certain subsets of the state space [LAS 61]. The problem of stabilization of switched dynamical systems is thus much more difficult in general than in classical control theory. In particular, instability phenomena can occur even when all the modes, taken separately, are stable.

Although the general control theory of switched systems is very difficult, special cases of these problems arise frequently in restricted contexts associated with control design, and may be simpler to solve specifically. It is thus suggested in [LIB 99] to stay in close contact with particular applications of switched systems. This is indeed what happens to the authors of this book: as researchers in LSV Computer Science Laboratory, we have cooperated with researchers of SATIE Electronics Laboratory in the framework of an interdisciplinary project relevant to the control of practical examples of switched systems in the domain of power electronics. We have thus focused on switching signals that operate with a fixed switched period denoted by τ. These signals are very common because of their ease of implementation. Also a fixed-period operation avoids potentially troublesome harmonic side-effects that may arise with varying frequency operation (see [GEY 08]). There are two types of periodic switched controllers: *state-independent* controllers that cyclically apply the same sequence of modes that has been computed off-line and *state-dependent* controllers that select modes dynamically according to the regions of the states at the switching instants. Furthermore, the dynamics of each subsystem obeying Ohm's electrical laws, are governed by *affine* differential equations. These systems can thus be viewed as special cases of *hybrid systems* (see [HEN 96]) combining affine continuous dynamics and discrete transitions taking place at instants that are integer multiples of τ. Such a subclass has been recently studied by many researchers such as Antoine Girard, Giordano Pola and Paulo Tabuada (see, e.g., [TAB 09]). These systems are called "time-triggered sampled versions of switched systems". We call them here simply $\mathcal{S}^2$-systems (for sampled switched systems).

In classical control theory, we make use of Lyapunov theory in order to analyze and stabilize controlled systems. Roughly speaking, Lyapunov functions are energy functions characterizing the state of the system that decrease until they reach a 0 level, which corresponds to a level where the system is stable. These Lyapunov functions can be extended in the framework of switched systems under the form of so-called “multiple Lyapunov functions” or “common Lyapunov functions”, and plays a central role in, for example, [TAB 09]. However, there is no general method for finding appropriate Lyapunov functions, and we have preferred in this book to avoid using them. Instead, the theoretical tool that we have mainly used is based on the notion of the “(controlled) invariant” [BLA 99]. Note, however, that the two concepts of invariants and Lyapunov functions are closely related, and we can show (at least in the classical context) that the level sets of Lyapunov functions correspond to the boundaries of invariant sets, and that the converse holds.

This book focuses on the restricted class of sampled switched systems, and on methods for controlling them using invariants. We will exploit the construction of invariants in order to synthesize two main classes of controllers: *safety* controllers and *stability* controllers. Safety controllers aim at protecting the system from undesirable states, while stability controllers aim at driving the system to a steady-state operating condition. To synthesize safety controllers, we describe *indirect* methods working on an abstract discrete level, and *direct* methods working on the continuous state-space level. These methods adopt a classical *backward* computation of the reachable states. To synthesize stability controllers, we describe a method of state-space *decomposition* that allows us to construct limit-cyclic trajectories by iterated forward computation of the reachable states.

The control strategies synthesized by this method have been numerically simulated, and also successfully experimented on physical prototypes built by SATIE Electronics Laboratory. The code has been written in Octave [OCT 13], and is available at the end of this book. We hope that this book that surveys methods of the literature and presents some recent enhancements together with the description of the implementation code will be interesting for students and engineers, and will encourage them to use, experiment, adapt and improve these procedures.

This book is structured as follows. In Chapter 1, we formally define the model of $\mathcal{S}^2$-systems, and explain in Chapter 3, how to synthesize safety controllers for $\mathcal{S}^2$-systems. In Chapter 4, we explain how to synthesize stability controllers for $\mathcal{S}^2$-systems using an original procedure of state-space decomposition. We show in Chapter 5 how to apply the procedure for controlling an important application of power electronics. In Chapter 6, how to extend the procedure in order to synthesize robust safety controllers and reachability controllers is explained, and suggestions for how to use it for sensitivity analysis are given. The main results with some perspectives are reviewed in Conclusions and Perspectives. Notes citing sources and related works are given at the end of each chapter.

This book tries as much as possible to avoid too theoretical results, rather focusing on the practical ideas of control synthesis for sampled switched systems using invariance analysis. In particular, proofs of the results are usually omitted. An exception is presented in Chapter 4, where we do detail an original procedure of state-space decomposition, motivated by and applied to the analysis of concrete examples of power electronics.

1 Control Theory: Basic Concepts

This chapter presents basic concepts of control theory, which will be used in the remaining book.

In section 1.1, we present the general *control/plant* model. In section 1.2, we explain why the introduction of digital sensors and actuators in systems has fundamentally modified the issue of controlled stability. Finally, we introduce the model of *switched* systems, and explains their advantages compared with general systems (section 1.2.3). We then explain in section 1.3 how the notion of *invariant sets* can be used for proving safety and stability properties of controlled systems.

1.1. Model of control systems

A control system is generally divided into a controlled part, called a *plant*, and a *controller*. The plant is generally described as a dynamic time-invariant, possibly uncertain, system governed by equations of the form:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t), w(t)) & [1.1] \\ y(t) = g(x(t)), & [1.2] \end{cases}$$

where $x(t) \in \mathbb{R}^n$ is the *system state*, $u(t) \in \mathbb{R}^m$ is the *control input* $y(t) \in \mathbb{R}^p$ is the *output*, $w(t) \in \mathcal{W} \subset \mathbb{R}^q$ is a *disturbance* (or external input) and $\mathcal{W}$ is an assigned compact set. We will refer to $\mathbb{R}^n$ as the *state space* of the system. The general theory of control focuses on *feedback control*: the controller is fed with state signal $x(t)$ coming from the plant, and issues a control input $u(t)$ to the plant. A typical layout of a feedback control system is shown in Figure 1.1. Under general conditions (continuity for u and w, and Lipschitz property for f), the system admits a unique solution $x(t)$ on $\mathbb{R}_{\geq 0}$. Equations [1.1] and [1.2] are often simplified by disregarding $w(t)$, and assuming that $y = x$.

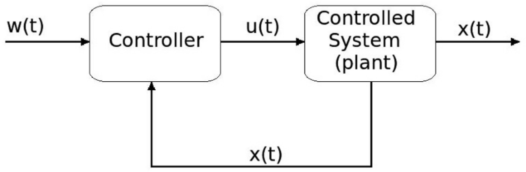


Figure 1.1. *Control/plant model*

An important subclass is the *linear time-invariant* (LTI) framework, for which [1.1] and [1.2] become: $\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) + Ew(t) \\ y(t) = Cx(t) \end{cases}$
for matrices A, B, C, E of appropriate size with constant coefficients. A *discrete-time LTI* system is a system governed by an equation of the form: $x(t+1) = Ax(t) + Bu(t) + Ew(t)$.

When a system is governed by an equation of the form $\dot{x}(t) = Ax(t)$, where A is a matrix whose eigenvalues have negative real parts, the origin is a *stable equilibrium* point to which the system converges from any initial point of $\mathbb{R}^n$. Given a plant governed by an equation of the form $\dot{x}(t) = Ax(t) + Bu(t)$ with $(A, B) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times m}$, a typical problem of linear control theory is to find a stabilizing controller governed by an equation of the form $u(t) = Kx(t)$ with $K \in \mathbb{R}^{m\times n}$. This essentially amounts to finding coefficient values of K that make the real parts of the eigenvalues of $A + BK$ negative.

1.2. Digital control systems

1.2.1. *Digitization*

With the emergence of digital computers, a control system has to handle data that come from the periodic sampling of signals. In such a context, a control system is said to be *sampled data* or *digital control system*. There, a system described by differential equations (which involve continuous-valued variables that depend on continuous time) is controlled by a discrete-time controller described by difference equations, which involve continuous-valued variables that depend on discrete time. As explained in [ANT 02], a digital control system can be divided into three parts, the plant, the interface, and the controller as shown in Figure 1.2.

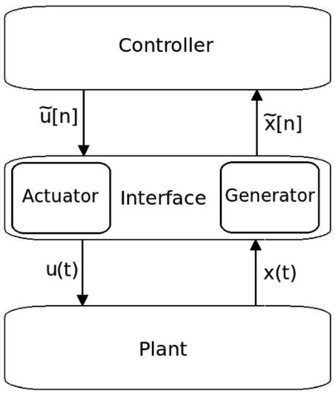


Figure 1.2. *Digital control/plant model (from [ANT 02])*

The system to be controlled (*plant*) is modeled as a time-invariant continuous-time system governed by equations [1.1] and [1.2] where, for the sake of simplicity, we disregard disturbance and assume that the output function is the identity map (i.e. we have $\dot{x} = f(x, u)$ and $y = x$).

The *controller* is a discrete event system modeled as a deterministic automaton. The action of the controller can be described by equations of the form:

$$\begin{cases} \tilde{s}[n] = \delta(\tilde{s}[n-1], \tilde{x}[n]) \\ \tilde{u}[n] = \phi(\tilde{s}[n]), \end{cases}$$

where δ is the state transition function of the controller and ϕ is the output function of the controller. Tildes are used to indicate that the particular signal is made up of symbols. The index n is here analogous to a time index in that it specifies the order of the symbols in the sequence. An argument in brackets, for example, $\tilde{x}[n]$, represents the nth symbol from a set. The input signal $\tilde{x}$ and the output signal $\tilde{u}$ associated with the controller are a sequence of symbols, rather than continuous-time signals. Note that there is *no delay* in the controller: the state transition, from $\tilde{s}[n-1]$ to $\tilde{s}[n]$, and the controller symbol, $\tilde{u}[n]$, occur immediately after the occurrence of plant symbol $\tilde{x}[n]$.

The controller and plant cannot communicate directly because each utilizes different types of signals. Thus, an *interface* is required that can convert continuous-time signals to sequences of symbols and vice versa. The interface consists of a memoryless map γ called *actuator*, and a memoryless map α called *generator*. The actuator converts a controller symbol $\tilde{u}[n]$ to a constant plant input of the form $u(t) = \gamma(\tilde{u}[n])$. Since the plant input, u, can only take on certain constant values, where each value is associated with a particular controller symbol, the plant input signal $u(t)$ is *piecewise constant*, and may change only when a controller symbol occurs. Such a piecewise continuous command signal issued by the actuator is illustrated in Figure 1.3. The generator is a function α that maps the real-valued state vector $x(t)$ of the plant into a plant symbol of the form $\tilde{x}[n] = \alpha(x(t))$. Note that $\tilde{x}$ does not change continuously, but only when a *plant event* occurs. There are two different models of plant event: in the *state-triggered* model, a plant event occurs when the plant state x crosses the boundary of two predefined state regions; in the *time-triggered* model, a plant event occurs periodically when the signal $\tilde{x}$ issued by the generator corresponds to a *periodic sampling* of the plant output x, as illustrated in Figure 1.4.

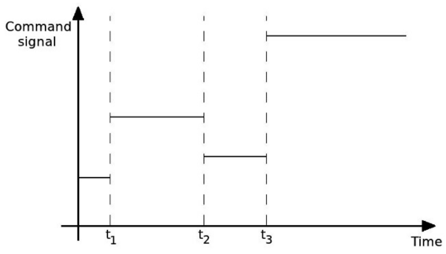


Figure 1.3. *Staircase command signal $u(t)$ issued by the actuator as it receives controller symbols $\tilde{u}[1], \tilde{u}[2], \ldots$ at time t_1, t_2, ... (from [ANT 02])*

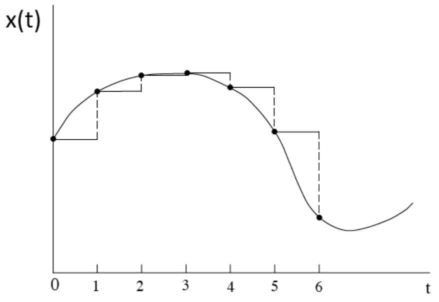


Figure 1.4. *Controller symbols $\tilde{x}[1], \tilde{x}[2], \ldots$ produced by the generator by sampling of the plant output signal $x(t)$ (time-triggered plant event model) (from [ANT 02])*

Note that, since it is assumed that there is no delay in the controller, the command signal $u(t)$ issued by the actuator is *synchronized* with the signal $x(t)$ issued by the generator. In the time-triggered model, the command $u(t)$ is therefore itself *periodic*. (In Figure 1.4, the stair length is constant and equal to τ.)

1.2.2. *Quantization*

Digitization also has an effect sometimes known as *quantization* (see, e.g., [PAT 05]). Suppose that the signal u now takes its values on a *finite* domain U, instead of a dense (possibly bounded) domain or an infinite discrete domain. This means that, in Figure 1.3, the plant input signal $\mathbf{u}(t)$ is a staircase signal that can take only a *finite* number of values. In such a situation, there are many systems (even LTI systems) for which there is no control function that ensures stabilization, that is convergence to a unique equilibrium point (see, e.g., [BRO 00]). The controller can only achieve *practical stability*, that is convergence into a bounded set instead of a single point for general stability The goal is then to synthesize controllers that are capable of steering the system to within sufficiently small neighborhoods of the equilibrium. The size of the final set within which the trajectories are confined is a measure of performance of the controlled dynamics. Hence, for a quantized system, the notion of *minimal invariant* set (once a proper notion of size has been defined) is useful for describing zones of practical stability.

1.2.3. *Switching*

A *switched system* is a digital quantized control system that consists of a finite family of continuous subsystems and a rule that controls the switching between them. More precisely, we have the following definition:

DEFINITION 1.1.– *A* switched system *is a quadruple* $\mathcal{S} = (\mathbb{R}^n, U, \mathcal{U}, \mathcal{F})$*, where* $\mathbb{R}^n$ *is the state space;* $U = \{1, \ldots, N\}$ *is the finite set of modes;* $\mathcal{U}$ *is the set of piecewise constant functions from* $\mathbb{R}_{\geq 0}$ *to* U*, continuous from the right; and* $\mathcal{F} = \{f_1, \ldots, f_N\}$ *is a collection of smooth vector fields indexed by* U*.*

A *switching signal* of $\mathcal{S}$ is a function $\mathbf{u} \in \mathcal{U}$. A piecewise $\mathcal{C}^1$ function $\mathbf{x} : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is said to be a *trajectory* of $\mathcal{S}$ if it is continuous and there exists a switching signal $\mathbf{u} \in \mathcal{U}$ such that, at each $t \in \mathbb{R}_{\geq 0}$ where the function $\mathbf{u}$ is continuous, $\mathbf{x}$ is continuously differentiable and satisfies:

$$\dot{\mathbf{x}}(t) = f_{\mathbf{u}(t)}(\mathbf{x}(t)).$$

The times at which the switching signal changes its values are called the *switching instants*. The scheme of switched systems is represented in Figure 1.5. It is easy to see that a quantized discrete-time LTI system is a particular subclass of switched systems (for which the function $f_{\mathbf{u}(t)}(x(t))$ is of the form $Ax(t) + B\mathbf{u}(t)$). However, the class of switched systems is much more general.

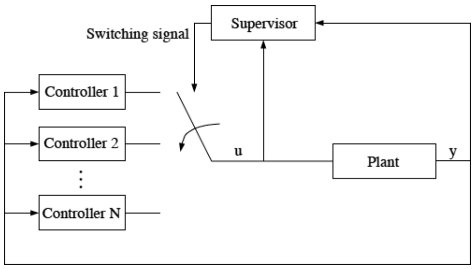


Figure 1.5. *Scheme of a switching controller*

In recent years, control techniques based on switching between different controllers, as shown in Figure 1.5, have been used in order to achieve stability and improve transient response. The importance of such control methods also arises from the existence of systems that cannot be stabilized by a single continuous feedback law (see [BRO 83]). In contrast, even if the different components of a switched system working in their proper mode have no (common) equilibrium, it is still possible to control the global system in order to make its behavior similar to those of conventional stable systems near equilibrium (see, e.g., [BUI 05]). Switched systems have thus found numerous applications switching power converters and many other fields (see [LIB 99]).

Caveat

Note, however, it is possible for a switched system to be unstable even when all the subsystems are stable around a common equilibrium point. This is true even when the subsystems are linear, as illustrated in the following example (see [ANT 02]). Consider the switched system $\dot{x}(t) = A_u x(t)$, where $x \in \mathbb{R}^2$, $u \in \{1, 2\}$, and $A_1 = \begin{pmatrix} -1 & 100 \\ 10 & -1 \end{pmatrix}$, $A_2 = \begin{pmatrix} -1 & -10 \\ 100 & -1 \end{pmatrix}$, with a (state-triggered) switching signal that applies A_1 (respectively A_2) when x is in the second and fourth (respectively first and third) quadrants. Both A_1 and A_2 are stable since their eigenvalues $\lambda_{1,2} = -1 \pm j\sqrt{1{,}000}$ have negative real parts. However, their trajectories are unstable (see Figure 1.6). Such a phenomenon takes place because the intervals between the switchings of the dynamics decrease to 0 as time goes to infinity. This can be avoided by imposing a minimum duration (called *dwell time*) between two switching instants. This can be easily enforced for the class of *sampled* switched systems that we will study in this book for which switchings occur with a fixed period τ.

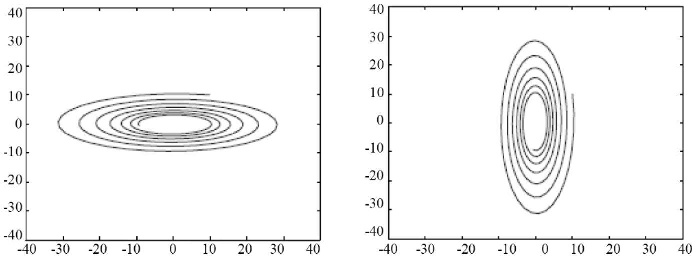


Figure 1.6. *Unstable trajectory of switched system consisting of stable subsystems (from [ANT 02])*

1.3. Control of switched systems using invariant sets

We now consider the problem of synthesizing controllers for switched systems. This results in finding a switching signal that controls the system in order to satisfy some given properties. We focus on the safety and stability properties. We explain that the controller synthesis problem is related to the construction of controlled invariant sets.

1.3.1. *Controlled invariants*

Given a dynamic system, a subset $\mathcal{I}$ of the state space is said to be *invariant* if it has the following property: if it contains the system state at some time, then it will contain it also in the future [BLA 99][1]. We have:

$$x(t) \in \mathcal{I} \Rightarrow x(t') \in \mathcal{I}, \text{ for all } t' \geq t.$$

The concept of invariance can be easily extended to the case in which a control input is present. In this case, we say that a set R is *controlled invariant* if, for all initial conditions chosen in R, we can keep the trajectory inside $\mathcal{I}$ *by means of a proper switching signal*. Let us now explain why controlled invariants are useful for proving *safety* and *stability* properties of a switched system.

1.3.2. *Safety control problem*

The safety property is typically encoded as a subset S of the continuous state space, called *safe set*. In a simple formulation, S is a box set given by the minimum and maximum values tolerated for each state variable. The associated safety properties suffice to describe typical requirements of direct-current to direct-current (DC-DC) power converters such as voltage regulation, current limitation, maximal current and voltage ripple.

Safety control problem: given a safe set S, determine whether a switching signal $\mathbf{u}$ exists such that if $x(0) \in S$, then $x(t) \in S$ for $t \geq 0$.

1 This property is often called "positively invariant" instead of just "invariant" in the literature.

Several approaches [ASA 00, TOM 00] have been proposed to solve the safety control problem. The idea of these approaches is to obtain a *controlled invariant* W that is included into S for an appropriate switching signal $\mathbf{u}$. If such a set W exists and if the initial state is in W, then the system is ensured to stay in W, hence in the safe set S. In [ASA 00], an abstract algorithm is proposed to synthesize controlled invariants using a backward iterative computation of reachable states. Furthermore, the set W computed is the *maximal controlled invariant* subset of S (it contains all other controlled invariant included into S). In [TOM 00], the controller synthesis problem is formulated as a game between controller and disturbance. We can then find Hamilton–Jacobi equations whose solutions describe the boundaries of the maximal safe set, and derive an associated maximally permissive controller. In Chapter 3, we will discuss methods to synthesize safety controllers that are adapted to the simpler context of sampled switched systems that we consider here.

1.3.3. *Stability control problem*

Given a certain region R, many controlled invariants subsets of R exist. If, instead of looking for maximal invariant subsets, we look for finding invariants as small a size as possible around a given operating point, we get a characterization of a controller with the smallest deviation from the point, and obtain a steady-state behavior with "minimum ripple" (see [SEN 03]). When periodic solutions of the system exist, we should be able to synthesize a *stability controller* that makes the trajectories converge to such periodic solutions of the system, also called *limit cycles*.

Stability control problem: given a region R, determine a switching signal $\mathbf{u}$ that makes the trajectories starting in R converge to a subregion as small as possible, ideally a limit cycle.

In Chapter 4, we will discuss a method based on a procedure of state-space decomposition, and iterated computation of forward reachable states for synthesizing stability controllers.

1.3.4. *Other controllers*

We will also give some hints to solve the problem of synthesizing *robust safety* controllers that maintain the plant in a safety region in the presence of disturbance or uncertainty, as well as *reachability controllers*, which drive the plant in finite time from an initial operating region to a desired operating region (see Chapter 6).

1.4. Notes

The common approach for stability analysis of dynamic systems is based on Lyapunov's method, which relies on the concept of a *Lyapunov function* or generalized energy function. Essentially, a Lyapunov function for an equilibrium point x_e of the system $\dot{x} = f(x)$ is a differentiable function $V(x)$ that has a strict minimum as x_e, and so that its derivative $\dot{V}(x) = \frac{\partial V(x)}{\partial x} \cdot f(x)$ along the system trajectories is negative in some neighborhoods of the equilibrium. Various converse theorems establish the existence of a Lyapunov function whenever the equilibrium point is stable (in the appropriate sense). There are fundamental connections between the notion of Lyapunov function and that of invariance. Precisely, given a Lyapunov function, its level sets are the boundaries of invariant sets. In this book, we will not use Lyapunov functions, but focus on invariant sets.

The context of section 1.2 is mainly taken from [ANT 02].

2

Sampled Switched Systems

We have seen in Chapter 1, that there are two models of plant event: the state-triggered model where a plant event occurs when the plant state crosses the boundary of two regions and the time-triggered model when a fixed time-out has been reached. In the rest of this book, we will focus on the time-triggered model. Furthermore, we will suppose that the switching instants occur *periodically*. We say that such switched systems are *sampled*, and refer to them as $\mathcal{S}^2$-systems (for “sampled switched” systems).

In this chapter, we give the formal model (section 2.1) of $\mathcal{S}^2$-systems together with illustrative examples (section 2.2). We also explain how to represent efficiently sets of states using the notion of “zonotope” (section 2.3).

2.1. Model

We are now considering a subclass of switched systems (see definition 1.1) for which the switching signal changes its values *periodically*, with a fixed period denoted by τ. The parameter τ is called the *sampling period*. We call such switched systems *sampled switched systems*, and refer to them as $\mathcal{S}^2$-systems. For an $\mathcal{S}^2$-system of sampling period τ, the control synthesis problem then amounts to finding the value of the switching signal at times τ, 2τ, ... In addition, we make assumptions that are commonly met in practice in embedded control applications of power electronics and the automotive industry. They are as follows:

– (A1) We focus on *affine* dynamics: the function $f_u(x)$ is of the form $A_u x + b_u$.

– (A2) We consider that the solution of the differential equation is *continuous*: there is no "jump" of the trajectory at the switching instants.

– (A3) We only consider the properties of the system at *switching instants* $\tau, 2\tau, \ldots$, and ignore possible state-constraint violations of the system in between.

These assumptions are classically done in power electronic systems. In power electronics, a typical circuit is a network of electrical components selected from the following three groups: ideal voltage or current sources, linear elements (e.g. resistors, capacitors, inductors and transformers) and nonlinear elements acting as switches (see [SEN 03]). At this level of abstraction, the behavior of a switch is idealized as having two discrete states: an open circuit and a short circuit. In a circuit with K switches, there are 2^K possible modes. In practice, however, not all these modes are admissible. Some of them are not feasible because of the physical characteristics of the switches, while others are banned by the designer because of safety considerations. Because of the restricted choice of circuit elements, the resulting systems have the desirable property that the continuous dynamics of each mode are linear or affine, which justifies (A1). The absence of "jump" assumed in (A2) is met in practice because of the continuity of the laws of physics. Finally, from this continuity, it follows that the constraint violations between switching instants are limited and become negligible for sufficiently small sampling periods.

Formally, we have the following definition (see [TAB 09, GIR 10a]):

DEFINITION 2.1.– *An $\mathcal{S}^2$-system Σ of sampling period $\tau \geq 0$ is a switched system $\Sigma = (\mathbb{R}^n, U, \mathcal{U}, \mathcal{F})$, where*

– the switching instants of $\mathbf{u} \in \mathcal{U}$ occur periodically at times τ, 2τ, $\ldots$

– *$\mathcal{F}$ is a set of functions $\{f_u\}_{u \in U}$ with, for any $u \in U$ and $x \in \mathbb{R}^n$, $f_u(x) = A_u x + b_u$ with $A_u \in \mathbb{R}^{n \times n}$ and $b_u \in \mathbb{R}^n$.*

In the following, it is convenient to assume that the matrix A_u governing the dynamics of mode u is invertible. This assumption is met in realistic models of physical systems[1].

Given an initial condition $x_0 \in \mathbb{R}^n$ (such that $\mathbf{x}(0) = x_0$), the trajectory is fully determined by the values $u_1, u_2, \ldots,$ of u at switching instants $\tau, 2\tau, \ldots$. These values define a switching signal $\mathbf{u}(t)$, which is constant on each interval $[k\tau, (k+1)\tau)$, for all $k \in \mathbb{N}$. Between two switching instants, the system is governed by a differential equation of the form: $\dot{\mathbf{x}}(t) = A_u \mathbf{x}(t) + b_u$ with $u \in U$. We will use $\mathbf{x}(t, x, u)$ to denote the point reached by Σ at time t (since last switching) under mode u from the initial condition x. This gives a transition relation $\rightarrow_u^\tau$ defined, for all x and x' in $\mathbb{R}^n$, by:

$$x \rightarrow_u^\tau x' \text{ iff } \mathbf{x}(\tau, x, u) = x'.$$

For a given $\mathcal{S}^2$-system Σ, the transition relation $\bigcup_{u \in U} \rightarrow_u^\tau$ will be denoted by $\rightarrow^\tau$. (In other words: $x \rightarrow^\tau x'$ means $x \rightarrow_u^\tau x'$ for some $u \in U$.)

DEFINITION 2.2.– *Given an $\mathcal{S}^2$-system Σ, a set $X \subset \mathbb{R}^n$ is* controlled invariant *if:*

$$\forall x \in X \ \exists u \in U \ \exists x' \in X \ x \rightarrow_u^\tau x'.$$

The set of successors of X via mode u, *denoted by $Post_{u,\tau}(X)$, or more simply by $Post_u(X)$, is:*

$$\{x' \mid x \rightarrow_u^\tau x' \textit{ for some } x \in X\}.$$

1 For example, if the matrix associated with a model of electrical circuit is non-invertible, it is often because some resistances have been neglected and idealized to 0.

The set of predecessors of X via mode u*, denoted by* $Pre_{u,\tau}(X)$*, or more simply by* $Pre_u(X)$*, is:*

$$\{x' \mid x' \to_u^\tau x \textit{ for some } x \in X\}.$$

Given a set $R \subset \mathbb{R}^n$*, a subset* X *of* R *is* R-invariant via mode u *if:* $Post_u(X) \subset R$*.*

PROPOSITION 2.1.– The mappings $Post_u : 2^{\mathbb{R}^n} \to 2^{\mathbb{R}^n}$ and $Pre_u : 2^{\mathbb{R}^n} \to 2^{\mathbb{R}^n}$, with $u \in U$, are affine transformations.

PROOF–. Given a mode $u \in U$, we have $\dot{\mathbf{x}}(t) = A_u\mathbf{x}(t) + b_u$ with $u \in U$ and $(A_u, b_u) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times 1}$. Therefore, we have $\mathbf{x}(\tau, x, u) = e^{A_u\tau}x + \int_0^\tau e^{A_u(\tau-t)}b_u dt = C_u x + d_u$ with $C_u = e^{A_u\tau}$ and $d_u = \int_0^\tau e^{A_u(\tau-t)}b_u dt = (e^{A_u\tau} - \mathcal{I}_n)A_u^{-1}b_u$, where $\mathcal{I}_n$ denotes the identity matrix. Hence, $Post_u$ is an affine transformation. The proof is similar for Pre_u.

As stated in assumption (A3), we will focus on the states reached by the continuous-time trajectories at switching instants $0, \tau, 2\tau, \dots$, and disregard the continuous portions of trajectories between two switching instants. This is depicted in Figure 2.1: we focus only on the segment $F = [x_1(0), x_2(0)]$ and its exact segment successor $[x_1(\tau), x_2(\tau)]$ at time τ, which corresponds to an affine image of the form $C_u(F) + d_u$, and do not construct a polyhedral over-approximation of the continuous trajectories starting from F (see steps (b) and (c)) as in [ASA 00]. Such a restriction allows us to simplify many works in the literature. The price to be paid is that, *between* two switching instants, the system may violate temporarily a desired property.

An $\mathcal{S}^2$-system can thus be seen a *discrete-time* system governed by an affine equation of the form: $x(t + \tau) = C_u x(t) + d_u$ with $C_u \in \mathbb{R}^n \times \mathbb{R}^n$ and $d_u \in \mathbb{R}^n$. Given an $\mathcal{S}^2$-system Σ and an initial point, the set of points $\{x_0, x_1, x_2, \dots\}$ corresponding to the states of the system at instants $0, \tau, 2\tau, \dots$, will be referred to as the *discrete trajectory* of Σ starting at x_0. In the figures, for facilitating visualization, consecutive points of discrete trajectories will be linked together by *straight* lines (unlike the real continuous-time trajectory portion which is exponential).

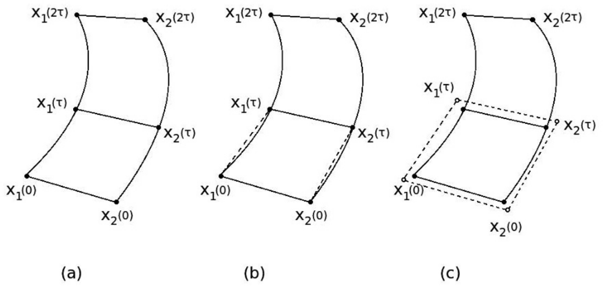


Figure 2.1. *a) A segment $F = [x_1(0), x_2(0)]$ and its exact segment successor $[x_1(\tau), x_2(\tau)]$ at time τ; b) approximating the set of continuous trajectories starting from F during τ time by convex hull; c) bloating the convex polyhedron to obtain a polyhedral over-approximation (from [ASA 00])*

Unless otherwise stated, the notation $\|\cdot\|$ will denote the Euclidean norm: for any $x \in \mathbb{R}^n$, $\|x\|$ is defined by $\|x\| = (x_1^2 + \cdots + x_n^2)^{1/2}$, where x_i is the ith component of the vector x. The exponential of any matrix $A \in \mathbb{R}^{n\times n}$ is denoted by e^A and is the analytic function $\sum_{i=0}^{\infty} \frac{1}{i!} A^i$. The ball of radius $\varepsilon \in \mathbb{R}_{\geq 0}$ centered at $x \in \mathbb{R}^n$ is denoted by $\mathcal{B}(x,\varepsilon)$ and is defined as the set of all the points $x' \in \mathbb{R}_{\geq 0}$ satisfying $\|x - x'\| \leq \varepsilon$. Similarly, $\mathcal{B}(X,\varepsilon) = \bigcup_{x\in X} \mathcal{B}(x,\varepsilon)$ for all $X \subset \mathbb{R}^n$.

DEFINITION 2.3.– *We say that a mode $u \in U$ is* contractive *if there exists $0 \leq \beta_u < 1$ such that, for all $x, y \in \mathbb{R}^n$:*

$$\|\mathbf{x}(\tau, x, u) - \mathbf{x}(\tau, y, u)\| \leq \beta_u \|x - y\|.$$

Given a subset $R \subset \mathbb{R}^n$, We say that a mode $u \in U$ is locally contractive in *R if there exists $0 \leq \beta_u < 1$ such that, for all $x, y \in R$: $\|\mathbf{x}(\tau, x, u) - \mathbf{x}(\tau, y, u)\| \leq \beta_u \|x - y\|$.*

It is easy to see that a mode u is contractive iff $\|e^{A_u \tau}\| = \beta_u$ for some $0 \leq \beta_u < 1$. This is equivalent to saying that all the eigenvalues of $A_u\tau$ have negative real parts. Similarly, a mode u is locally contractive in R

if there exists $0 \leq \beta_u < 1$ such that, for all $x \in R$, $\|e^{A_u\tau}x\| \leq \beta_u\|x\|$[2]. To show that a mode u is locally contractive in R, it suffices to show that there exists a right cone $\mathcal{C}$ that contains R such that $\exists 0 \leq \beta_u < 1, \forall x \in \mathcal{S}(0_{\mathbb{R}^n}, 1) \cap \mathcal{C},\ \|e^{A_u\tau}x\| \leq \beta_u$, where $\mathcal{S}(0_{\mathbb{R}^n}, 1)$ is the unity sphere (i.e. $\mathcal{S}(0_{\mathbb{R}^n}, 1) = \{x \in \mathbb{R}^n, \|x\| = 1\}$).

A *pattern* of the form $(u_1 \cdot u_2 \cdots u_m)$ is a finite sequence of modes $u_1, u_2, \ldots, u_m$ of U. A *k-pattern* is a pattern of length at most k. In this book, patterns will be often associated with finite paths in oriented graphs whose edges are labeled by modes. We will use the expression π^i for denoting the concatenation of i patterns equal to π, and π^* for the concatenation of π an arbitrary number of times.

The definition of successors via modes (definition 2.2) extends naturally for patterns. Formally, for $X \subset \mathbb{R}^n$ and a pattern π of the form $(u_1 \cdot u_2 \cdots u_m)$, we have: $Post_\pi(X) = Post_{u_m}(\cdots(Post_{u_1}(X))\cdots)$. We write sometimes $x \to_\pi x'$ to mean $x \to^\tau_{u_1} x_1 \to^\tau_{u_2} \cdots x_{m-1} \to^\tau_{u_m} x'$ for some $x_1, \ldots, x_{m-1} \in \mathbb{R}^n$.

Similarly, definitions of predecessors, R-invariance and (local) contractivity for modes extend naturally to those for patterns. From proposition 2.1, it follows:

PROPOSITION 2.2.– *Let π be a pattern. Then the mappings $Post_\pi\colon 2^{\mathbb{R}^n} \to 2^{\mathbb{R}^n}$ and $Pre_\pi\colon 2^{\mathbb{R}^n} \to 2^{\mathbb{R}^n}$ are affine transformations.*

The image of a convex set X by $Post_\pi$ (respectively, Pre_π) is therefore a convex set. Given a convex set $R \subset \mathbb{R}^n$ and a pattern π, in order to show that a convex subset X is R-invariant via π, it suffices to show that every vertex of X is mapped via $Post_\pi$ onto a point of R.

2.2. Illustrative examples

EXAMPLE 2.1.– (BOOST DC–DC CONVERTER).– This example is taken from [BEC 05] (see also, e.g., [GIR 10b, BUI 05, SEN 03]). This

2 Note that this requires that at least one eigenvalue of $A_u\tau$ has a negative real part.

is a Boost DC–DC converter with one switching cell (see Figure 2.2). The state of the system is $x(t) = [i_l(t)\ v_c(t)]^T$, where $i_l(t)$ is the inductor current and $v_c(t)$ the capacitor voltage. There are two operation modes depending on the position of the switching cell. When the switch is closed (mode 2), the inductor current i_l increases and energy is stored to the inductance. When the switch is open (mode 1), the energy accumulated in the inductance is transferred to the capacitor. The dynamics associated with mode u is of the form $\dot{x}(t) = A_u x(t) + b_u$ $(u = 1, 2)$ with:

$$A_1 = \begin{pmatrix} -\frac{r_l}{x_l} & 0 \\ 0 & -\frac{1}{x_c}\frac{1}{r_0+r_c} \end{pmatrix} \quad b_1 = \begin{pmatrix} \frac{v_s}{x_l} \\ 0 \end{pmatrix},$$

$$A_2 = \begin{pmatrix} -\frac{1}{x_l}(r_l + \frac{r_0 r_c}{r_0+r_c}) & -\frac{1}{x_l}\frac{r_0}{r_0+r_c} \\ \frac{1}{x_c}\frac{r_0}{r_0+r_c} & -\frac{1}{x_c}\frac{1}{r_0+r_c} \end{pmatrix} \quad b_2 = \begin{pmatrix} \frac{v_s}{x_l} \\ 0 \end{pmatrix}.$$

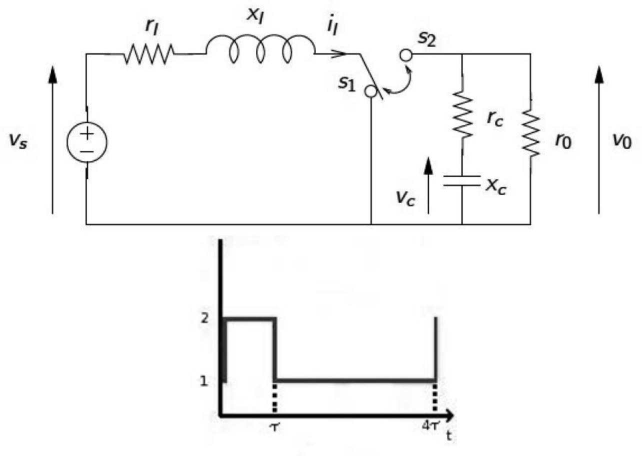


Figure 2.2. *a) Scheme of the Boost DC–DC converter; b) cell switching for pattern* $(2 \cdot 1 \cdot 1 \cdot 1)$

We will use the numerical values of [BEC 05], expressed in the per-unit system: $x_c = 70$, $x_l = 3$, $r_c = 0.005$, $r_l = 0.05$, $r_0 = 1$ and $v_s = 1$. The sampling period is $\tau = 0.5$.

A general goal of the control is to stabilize the output voltage v_0 around a desired value v_e. The range of variations of the output voltage and inductor current should be limited in order to avoid phenomena of inductor saturation and blocking voltage stress of the switch. This corresponds to the specification of a safety area S. The safety control problem is to find a strategy for deciding which sequence of patterns to apply in order to keep the state within S. An example of pattern of length 4 is illustrated in Figure 2.2: it corresponds to the application of mode 2 on $(0, \tau]$ and mode 1 on $(\tau, 4\tau]$. The control can be state-independent, consisting of the repeated application of the same sequence of patterns (computed off-line), or state-dependent, with the application of a pattern depending on the current value of the electrical state.

EXAMPLE 2.2.– (TWO-ROOM BUILDING HEATER).– This example is taken from [GIR 12]. It is a simple heating model of a two-room building. One of the rooms can be heated via a heating device. The two rooms communicate such that heat from one room can diffuse to the other. Moreover, the rooms are surrounded by an environment that has a fixed temperature. By controlling when to turn the heating device on and off, we are interested in maintaining the two rooms at a comfortable temperature. Let $T = [T_1 \; T_2]^T$ be the state variable, where T_i is the temperature of room i ($i = 1, 2$). The dynamics of the system are given by the following equation:

$$\dot{T} = \begin{pmatrix} -\alpha_{21} - \alpha_{e1} - \alpha_f u & \alpha_{21} \\ \alpha_{12} & -\alpha_{12} - \alpha_{e2} \end{pmatrix} T + \begin{pmatrix} \alpha_{e1} T_e + \alpha_f T_f u \\ \alpha_{e2} T_e \end{pmatrix},$$

where u is a mode of value 0 or 1, and the heat transfer coefficients and external temperatures are given by the values: $\alpha_{12} = 5 \times 10^{-2}$, $\alpha_{21} = 5 \times 10^{-2}$, $\alpha_{e1} = 5 \times 10^{-3}$, $\alpha_{e2} = 3.3 \times 10^{-3}$, $\alpha_f = 8.3 \times 10^{-3}$, $T_e = 10$ and $T_f = 50$. The sampling period is $\tau = 5$.

EXAMPLE 2.3.– (HELICOPTER MOTION).– This example is taken from [DIN 11]. The problem is to control a quadrotor helicopter toward a particular position on top of a stationary ground vehicle, while satisfying constraints on the relative velocity. By controlling the pitch and roll angles, we can modify the speed and the position of the helicopter. A typical problem is to find a switching rule, depending on the position and velocity of the helicopter, in order to keep the system state within a safe area, avoiding excessive speed or distance in relation to the ground vehicle. Let g be the gravitational constant, x (respectively y) the position according to x-axis (respectively y-axis), $\dot{x}$ (respectively $\dot{y}$) the velocity according to x-axis (respectively y-axis), ϕ the pitch command and ψ the roll command. The possible commands for the pitch and the roll are the following: $\phi,\psi \in \{-10, 0, 10\}$. Since each mode corresponds to a pair (ϕ,ψ), there are nine modes. The dynamics of the system are given by the equation:

$$\dot{X} = \begin{pmatrix} 0\,1\,0\,0 \\ 0\,0\,0\,0 \\ 0\,0\,0\,1 \\ 0\,0\,0\,0 \end{pmatrix} X + \begin{pmatrix} 0 \\ g\ \sin(-\phi) \\ 0 \\ g\ \sin(\psi) \end{pmatrix},$$

where X is $[x\ \dot{x}\ y\ \dot{y}]^T$. The sampling period is $\tau = 0.1$. Since the variables x and y are decoupled in the equations and follow the same equations (up to the sign of the command), it suffices to study the control for x. (The control for y is opposite.)

2.3. Zonotopes

The construction of invariant sets under the form of (union of) *polyhedral* sets is natural because polyhedra correspond to sets of linear constraints of the state space. Furthermore, they are well suited to the approximation of reachability sets and domains of attractions of dynamic systems. *Zonotopes* are a data structure which is very useful for representing and manipulating efficiently convex polytopes (see, e.g., [ALT 08, GIR 05a, KÜH 98]). They can be seen as symmetric polyhedra where a facet must be parallel to an opposing facet. The class of zonotopes is closed under linear transformation and Minkowski sum3. Furthermore, using zonotopes, it is easy to introduce

3 The Minkowski of two sets A, B is defined by $A + B = \{a + b \mid a \in A, b \in B\}$.

uncertainty or disturbance in the dynamics of the models. A zonotope is defined by a center c to which linear segments $l_i = \beta^{(i)} \cdot g^{(i)}$, $-1 \leq \beta^{(i)} \leq 1$ are added via Minkowski sum.

DEFINITION 2.4.– *A* zonotope *is a set:*

$$Z = \{x \in \mathbb{R}^n : x = c + \Sigma_{i=1}^{p} \beta^{(i)} \cdot g^{(i)},\ -1 \leq \beta^{(i)} \leq 1\}$$

with $c, g^{(1)}, \dots, g^{(p)} \in \mathbb{R}^n$.

The vectors $g^{(1)}, \dots, g^{(p)}$ are referred to as the *generators* and c as the *center* of the zonotope. It is convenient to represent the set of generators as an $n \times p$ matrix G of columns $g^{(1)}, \dots, g^{(p)}$. The notation is $< c, G >$.

A *box* (or *rectangle*) is a Cartesian product of n closed intervals. It can be seen as a zonotope of the form $< c, D >$, where c is the center of the box and D is an $n \times n$ diagonal matrix whose (i, i)th element is equal to half the size of the ith interval, for $1 \leq i \leq n$. Boxes play an important role in invariance theory (see, e.g., [PIC 08, ABA 09]).

The smallest box containing a zonotope $Z =< c, G >$ is called the *bounding box of* Z, and denoted by $\square(Z)$. We have: $\square(Z) =< c, D >$, where D is an $n \times n$ diagonal matrix whose (i, i)th element is equal to $\Sigma_{\ell=1}^{p} |G_{i,\ell}|$, for $1 \leq i \leq n$.

Given a zonotope $Z =< c, G >$, the transformation of Z via an affine function $x \mapsto Cx + d$ is a zonotope of the form $< Cc + d, CG >$. The successor set $Post_u(Z)$ of Z via a mode u can thus be simply computed using zonotopes, using matrix multiplication whose complexity is (at most) cubic.

Zonotopes allow us to compute easily over-approximations of the successor sets in order to take into account small perturbations (or uncertainties) of the system dynamics. All the dynamics of the system are now of the form $\dot{x}(t) = A_u x(t) + b_u + \varepsilon(t)$, where $\varepsilon(t)$ represents disturbance under the form of a vector belonging to a given box

$\Lambda = [-\varepsilon_1, +\varepsilon_1] \times \cdots \times [-\varepsilon_n, +\varepsilon_n]$ of $\mathbb{R}^n$, with $\varepsilon_i \geq 0$ for $i = 1, \ldots, n$. We will use $\mathbf{x}(t, x, u, \varepsilon)$ to denote the point reached by the system at time t under mode u with disturbance ε, from the initial condition x. This entails a transition relation $\rightarrow_\tau^{u,\varepsilon}$ defined, for all $x, x' \in \mathbb{R}^n$, $\varepsilon \in \mathbb{R}^n_{\geq 0}$ and $u \in U$, by:

$$x \rightarrow_\tau^{u,\varepsilon} x' \text{ iff } \mathbf{x}(\tau, x, u, \varepsilon) = x'.$$

Given a box $\Lambda = [-\varepsilon_1, +\varepsilon_1] \times \cdots \times [-\varepsilon_n, +\varepsilon_n]$ of $\mathbb{R}^n$, we define $Post_u(X, \Lambda) = \{x' \mid \exists \varepsilon \in \Lambda,\ x \rightarrow_\tau^{u,\varepsilon} x'\}$. This definition of $Post_u$ with perturbation naturally extends to $Post_\pi$ where π is a pattern. If X is given under the form of a zonotope $< c, G >$, then it is easy to compute an overapproximation of $Post_u(X, \Lambda)$. Suppose that the successor set *without* perturbation $Post_u(X)$, is an affine transformation of the form $CX + d$, we have:

LEMMA 2.1.– Consider a zonotope $X =< c, G >$, a box $\Lambda = [-\varepsilon_1, +\varepsilon_1] \times \cdots [-\varepsilon_n, +\varepsilon_n]$ of $\mathbb{R}^n$. We have:

$$Post_u(X, \Lambda) \subset < Cc + d, CG + \tau D_\Lambda >,$$

with: $D_\Lambda = \begin{pmatrix} \varepsilon_1 & 0 & \ldots & 0 \\ 0 & \varepsilon_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \varepsilon_n \end{pmatrix}$.

The process can be iterated to compute a zonotopic over-approximation of $Post_\pi(X, \Lambda)$, for any pattern π. This technique will be used in Chapter 6 in order to handle the problem of *robust safety* control and the problem of *nonlinear dynamics*.

2.4. Notes

The class of $\mathcal{S}^2$-systems is a subclass of *affine hybrid systems* [HEN 96] where the discrete transitions only happen at instants that are integer multiples of τ. Alternatively, we can consider $\mathcal{S}^2$-systems as models capturing only continuous transitions of duration τ.

The semantics of sampled switched systems given here originate from the work of Antoine Girard, Giordano Pola and Paulo Tabuada (see, e.g., [TAB 09]). We have simplified their formalism here by removing the notion of *output* and *observation* sets. What has been called *sampled switched system* here, denoted by Σ, corresponds actually in [TAB 09, Chapter 11] to the notion of a *symbolic system associated with a switched affine system* Σ, denoted by $S_\tau(\Sigma)$.

The idea of approximating the state trajectories of dynamic systems using zonotopes comes from [KÜH 98]. Using zonotopes for enclosing the uncertainty, we can avoid the *wrapping effect*, which leads to exponential fast-growing enclosures when using an iteratively naive approximation (such as the bounding box). Zonotopes have recently received many applications in the domain of hybrid systems with uncertainty, such as reachability analysis [ALT 07].

3 Safety Controllers

In this chapter, we are interested in finding controllers of an $\mathcal{S}^2$-system which make the variables of the system stay within the limits of a given area S. The problem amounts to finding a switching rule that selects a mode ensuring that the system will still be in S at the next sampling time, and so on iteratively. If we consider S as a predefined *safe* region for the operating states of the system, then such a switching rule can be seen as a *safety controller*. The problem is closely related to the problem of finding a controlled invariant subset of S. It is interesting to design a controller that is as permissive as possible because this allows us to formulate secondary control objectives in order to satisfy various performance criteria inside the safe set. This amounts to synthesizing a controlled invariant subset of S which is as large as possible, ideally *maximal*.

We first present a *direct* approach working on the original continuous state space, which makes use of a general procedure of backward fixed point computation (section 3.1). We then present an *indirect* approach working on an abstract discrete state space, which makes use of the notion of approximate bisimilarity (section 3.2). The two methods are finally applied to the three-cell converter application of power electronics (section 3.3).

3.1. Backward fixed point computation (direct approach)

Suppose we are given an $\mathcal{S}^2$-system Σ and a set S of *safe* states. Let us consider a very general approach for synthesizing a (maximal) subset S^* of S which is a *controlled invariant* (i.e., such that: $\forall x \in S^* \; \exists u \in U = \{1 \dots N\} \; \exists x' \in S^* \; x \rightarrow^{\tau}_{u} x'$). Given a set S, it is easy to show that the union of two controlled invariant subsets of S is itself a controlled invariant subset of S. Accordingly, the notion of a *maximal controlled invariant* subset of S is well-defined and corresponds to the union of all the controlled invariant subsets of S.

Consider the operator $F_S : 2^{\mathbb{R}^n} \rightarrow 2^{\mathbb{R}^n}$ defined by:

$$F_S(X) = \bigcup_{u=1\dots N} Pre_u(X) \cap S.$$

The set $F_S(X)$ contains all the states $x \in X \cap S$ for which the successors via u of x are in X. The next result states that a *maximal fixed point* of F_S exists.

PROPOSITION 3.1.– *The operator* $F_S : 2^{\mathbb{R}^n} \rightarrow 2^{\mathbb{R}^n}$ *the following points:*

1) The sequence $\{F_S^i(\mathbb{R}^n)\}_{i \geq 0}$ *is nested and decreasing.*

2) The maximal fixed point S^* *of* F_S *satisfies:*

$$S^* = \lim_{i \rightarrow \infty} F_S^i(\mathbb{R}^n) = \bigcap_{i \geq 0} F_S^i(\mathbb{R}^n);$$

3) The maximal fixed point S^* *of* F_S *is the maximal controlled invariant subset of S.*

See [TAB 09], Chapter 5 for a proof. The maximal fixed point S^* of F_S (i.e. the maximal solution of equation $X = \bigcup_{u=1\dots N} Pre_u(X) \cap S$) can be computed by iteration of F_S until a fixed point S^* is obtained. The general algorithm is of the form:

Algorithm 3.1: *Synthesis of maximal controlled invariant subset*

```
  Input: A set $S \subset \mathbb{R}^n$
  Output: A maximal invariant subset $S^*$ of $S$
1 $X^0 := S$
2 repeat
3 $X^{k+1} := X^k \cap \bigcup_{u=1 \ldots N} Pre_u(X^k)$
4 until $X^{k+1} = X^k$
5 $S^* = X^k$
```

The correctness of algorithm 3.1 relies on the fact that, at step $k \geq 0$, the set X^k is the set of starting points of trajectories of length k contained in S. Formally: $X^k = \{x \in S \mid x \rightarrow_\tau x_1 \rightarrow_\tau \cdots \rightarrow_\tau x_k \text{ for some } x_1, \ldots, x_k \in S\}$. It follows that S^* is the set of starting points of infinite trajectories contained in S, that means that S^* is the maximal invariant subset of S.

Henceforth, we suppose that the input S is given under a polyhedral form. Every set X^k can be put under the form of a finite union of polyhedral components: each polyhedral component P of X^k is obtained as $Pre_u(Q) \cap R$, where u is in U, and Q and R are themselves two polyhedral components of X^{k-1}. (Recall that the operator Pre_u is an affine transformation that maps a polyhedron into another polyhedron; see Chapter 2). The test $X^{k+1} = X^k$ is performed by testing if the vertices of the components of X^k belong to (components of) X^{k+1}, and vice versa. If the test succeeds, algorithm 3.1 terminates, and outputs a set S^* which is a union of polyhedral components.

Let us now explain how we can derive a state-dependent control strategy that allows us to always maintain the system in S^*, using a simple additional information storage in algorithm 3.1. We modify the algorithm as follows: for each polyhedron component P produced at step k (of the form $Pre_u(Q) \cap R$, with $Q, R \subset X^{k-1}$), we store the mode u with which it has been produced. If algorithm 3.1 terminates, the output S^* is given under the form of a finite set of polyhedral components together with their associated modes. The control is now as follows: when at a switching time, the system state lies in a component P of S^*, we apply the associated mode u of P; at the next switching time, the system lies in a component P' of S^*, and we apply the associated mode u', and so on iteratively. Such a control that allows us to stay in the maximal invariant S^* subset of S is said to be *maximally safe* (or *maximally permissive*). In a further step, we can refine the maximally permissive controller in order to satisfy various performance criteria inside the safe set.

EXAMPLE 3.1.– To illustrate this approach, we synthesize a control for the Boost DC–DC converter with one cell (see example 2.1 for a description of the system). We have $S = [3.0, 3.4] \times [1.5, 1.8]$ in the (i_l, v_c) plane and $\tau = 0.5$. Algorithm 3.1 terminates in two steps:

– At step 1, it produces two polyhedral components $P_1 = Pre_1(S) \cap S$ and $P_2 = Pre_2(S) \cap S$, with $X^1 = P_1 \cup P_2$.

– At step 2, the set X^2 produced is the union of eight polyhedral components $Pre_i(P_j) \cap P_k$ with $i, j, k \in \{1, 2\}$, and it can be seen that $X^2 = X^1$.

This means that $S^* = X^1 = P_1 \cup P_2$. In Figure 3.1, the uncontrollable part $S \setminus S^*$ corresponds to the "horizontal" polyhedra colored in black in the lower left and upper right parts of S. The controlled subset P_1 corresponds to the "vertical" left polyhedron and P_1 to the right polyhedron. If the system state is in P_1 (respectively P_2) at a switching time, then mode 1 (respectively 2) should be applied. Mode 1 or 2 can be arbitrarily applied when the system lies in $P_1 \cap P_2$. A controlled trajectory starting at point $x_0 = (3.01, 1.79) \in S^*$ is depicted in Figures 3.1 and 3.2. We can see that the trajectory stays within $S^* \subset S$. Algorithm 3.1 involves the computation of the predecessor operator, intersection and test of point containment for polyhedra. However, the number of polyhedral components increases exponentially at each step. A realistic implementation of algorithm 3.1 requires merging different polyhedral components into an *under-approximated* polyhedral form. This can be done using, for example, the notion of *griddy polyhedra* (see [ASA 00], i.e. sets that can be written as unions of closed unit hypercubes with integer vertices). The under-approximation process also helps the algorithm to terminate. The output S^* of the algorithm is then an invariant subset of S, but is no longer maximal.

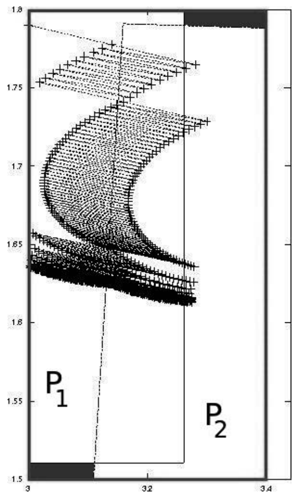


Figure 3.1. *Maximal controlled invariant subset of $S = [3.0, 3.4] \times [1.5, 1.8]$, composed of two polyhedra P_1 (mode 1) and P_2 (mode 2), with a controlled trajectory starting at $x_0 = (3.01, 1.79)$. For a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip*

3.2. Approximate bisimulation (indirect approach)

The direct application of algorithm 3.1 at the continuous state level works well on simple examples, as explained in section 3.1. However, there is no guarantee of termination of the procedure because the state space is infinite. An interesting alternative approach is the "indirect approach": it consists of making an *abstraction* of the system into a finite discrete system. Algorithm 3.1 then always terminates, and allows us to synthesize a maximally safe abstract controller, from which a controller can be derived at the real level, at the price of a certain approximation. We now explain such a method using the notion of *approximate bisimulation*.

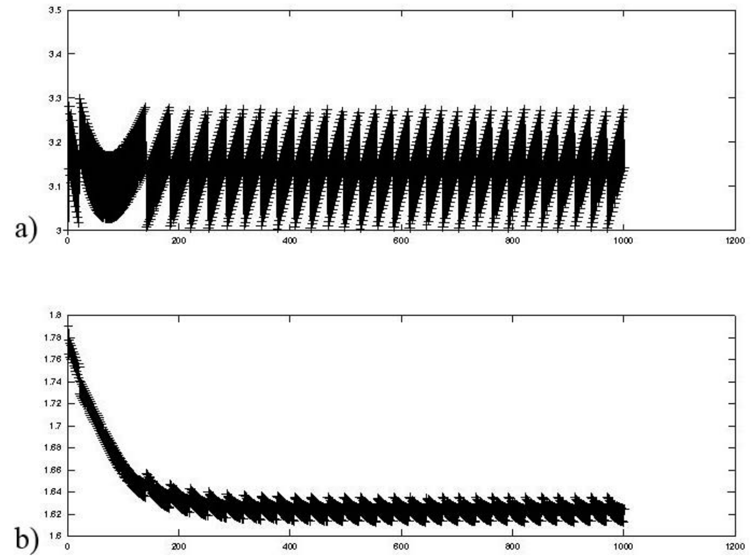


Figure 3.2. *Discrete-time trajectory starting from point* $x_0 = (3.01, 1.79)$*, using the control found by the direct method. a): evolution of* v_c *in time; b): evolution of* i_l

[GIR 10b] propose a method for abstracting a switched system under the form of a discrete model, that is equivalent to the original model, under certain Lyapunov-based stability conditions. They use a Euclidean norm $\|.\|$, and define the approximation of the set of states $\mathbb{R}^n$ as follows:

$$[\mathbb{R}^n]_\eta = \{x \in \mathbb{R}^n \mid x_i = k_i \frac{2\eta}{\sqrt{n}},\ k_i \in \mathbb{Z}, i = 1, \ldots, n\},$$

where $\eta \in \mathbb{R}^+$ is a state-space discretization parameter. It is then easy to see that: $\forall x \in \mathbb{R}^n\ \exists q \in [\mathbb{R}^n]_\eta : \|x - q\| < \eta$. The transition relation $\rightarrow^u_\tau$ of Σ is then approximated as follows: let $q \in [\mathbb{R}^n]_\eta$ and $q_e = \mathbf{x}(\tau, q, u)$ such that $q \rightarrow^u_\tau q_e$ in the real system, let $q' \in [\mathbb{R}^n]_\eta$ with $\|q_e - q'\| < \eta$. Then, we have $q \rightarrow^u_{\tau,\eta} q'$ for the approximated transition relation. Formally, the transition relation of the abstract system Σ_η is defined as follows.

DEFINITION 3.1.– *Given a switched system* $\Sigma\colon(\tau, U, \mathcal{F})$ *and its trajectory* $\mathbf{x}\colon\mathbb{R}^+ \to \mathbb{R}^n$*, the system* Σ_η *is the transition system* $(Q, \to^u_{\tau,\eta})$ *defined by:*

– the set of states $Q = [\mathbb{R}^n]_\eta$.

– the transition relation given by:
$q \to^u_{\tau,\eta} q'$ *iff* $\|\mathbf{x}(\tau, q, u) - q'\| \leq \eta$.

This relation is depicted in Figure 3.3 (see [GIR 10a]). The notion of "approximate bisimilarity" between systems Σ and Σ_η is defined as follows.

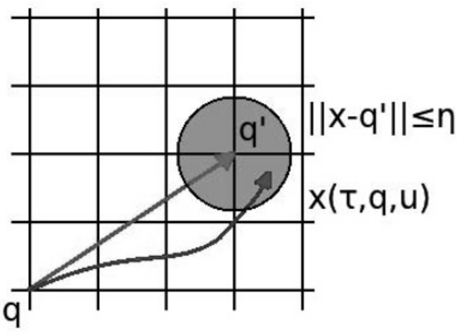


Figure 3.3. *Abstract transition relation (from [GIR 10a])*

DEFINITION 3.2.– *Systems* Σ *and* Σ_η *are* ε-bisimilar *(or bisimilar with precision* ε*) if:*

1) *for all* $x \in \mathbb{R}^n$ *and* $q, q' \in [\mathbb{R}^n]_\eta$*:* $(\|x - q\| \leq \varepsilon \wedge q \to^u_{\tau,\eta} q') \Rightarrow \|x' - q'\| \leq \varepsilon$ *for some* $x' = \mathbf{x}(\tau, x, u)$ *(i.e. for some* $x' : x \to^u_\tau x'$*);*

2) *for all* $x, x' \in \mathbb{R}^n$ *and* $q \in [\mathbb{R}^n]_\eta$*:* $(\|x - q\| \leq \varepsilon \wedge x \to^u_\tau x') \Rightarrow \|x' - q'\| \leq \varepsilon$ *for some* $q' \in [\mathbb{R}^n]_\eta$ *with* $\|\mathbf{x}(\tau, q, u) - q'\| \leq \eta$ *(i.e. for some* $q' : q \to^u_{\tau,\eta} q'$*).*

Consider a switched system $\Sigma = (\tau, U, F)$, a desired precision ε and a time sampling value τ. Under certain Lyapunov-based stabilization conditions, it is shown in [GIR 10b] that there exists a space sampling value η such that the transition systems of Σ and Σ_η are approximately bisimilar with precision ε. In the context of $\mathcal{S}^2$-systems, the conditions of Lyapunov-based stabilization can be simplified as follows.

THEOREM 3.1.– Consider a switched system $\Sigma = (\tau, U, F)$, a desired precision ε and a time sampling value τ. If all the modes of U are contractive, there exists a space sampling value η such that the transition systems of Σ and Σ_η are approximately bisimilar with precision ε.

PROOF.– Because of the contractivity of the modes of Σ, we have, for all $u \in U$, $\|\mathbf{x}(\tau, x, u) - \mathbf{x}(\tau, y, u)\| \leq \beta \|x - y\|$ for some $0 \leq \beta < 1$. The proof of ε-bisimularity is based on the fact that we can choose η so that $\beta\varepsilon + \eta \leq \varepsilon$ is true (which is possible because $\beta < 1$). We have indeed:

1) $(\|x - q\| \leq \varepsilon \wedge q \to^u_{\tau,\eta} q') \Rightarrow \|x' - q'\| = \|\mathbf{x}(\tau, x, u) - q'\| \leq \|\mathbf{x}(\tau, x, u) - \mathbf{x}(\tau, q, u)\| + \|\mathbf{x}(\tau, q, u) - q'\| \leq \beta\varepsilon + \eta \leq \varepsilon$, with $x' = \mathbf{x}(\tau, x, u)$ (i.e., $x' : x \to^u_\tau x'$ for some $u \in U$);

2) $(\|x - q\| \leq \varepsilon \wedge x \to^u_\tau x') \Rightarrow \|x' - q'\| = \|\mathbf{x}(\tau, x, u) - q'\| \leq \|\mathbf{x}(\tau, x, u) - \mathbf{x}(\tau, q, u)\| + \|\mathbf{x}(\tau, q, u) - q'\|, \leq \beta\|x - q\| + \eta \leq \beta\varepsilon + \eta \leq \varepsilon$ with q' such that $q \to^u_{\tau,\eta} q'$.

Note that theorem 3.1 for any norm $\|\cdot\|$. For implementing the method, it may be convenient to use the infinity norm (defined by $\|x\| = \max_{i=1,\cdots,n} |x_i|$, for all points $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$) rather than the Euclidean norm in order to reduce the overlapping of two adjacent bowls of radius η and the non-determinism of relation $\to^u_{\tau,\eta}$. Accordingly, the definition of $[\mathbb{R}^n]_\eta$ should be:

$$[\mathbb{R}^n]_\eta = \{x \in \mathbb{R}^n \mid x_i = 2k_i\eta \text{ for some } k_i \in \mathbb{Z} \text{ and } i = 1, 2, \ldots, n\}.$$

Let us now explain how to apply theorem 3.1 in order to synthesize a safety controller. Consider a bounded subset S of $\mathbb{R}^n$. The set $S_\eta = [\mathbb{R}^n]_\eta \cap S$ is *finite*. So, if algorithm 3.1 runs with $X^0 = S_\eta$ as an input, it terminates and outputs a maximal controlled invariant subset, say S^*_η, of S_η. For each point $q \in S_\eta^*$, there exists a point $q' \in S_\eta^*$ such that $q \rightarrow_{\tau,\eta}^{u} q'$ for some $u \in U$. Using the relation $\bigcup_{u \in U} \rightarrow_{\tau,\eta}^{u}$ restricted to S_η^*, we can define a finite state automaton $\mathcal{A}_\eta$ on S_η^*. This automaton $\mathcal{A}_\eta$ can be seen as a maximally safe controller of S_η. From such a controller, it is then possible, using the bisimilarity stated in theorem 3.1, to derive a controller for the real model Σ, which keeps the switched system Σ in $\mathcal{B}(S^*, \varepsilon)$ (see [TAB 08]).

An alternative approach consists of observing that, for all points q of S_η^*, there exists a quasi-cyclic sequence of transitions of $\mathcal{A}_\eta$ starting at q of the form $\pi \cdot \sigma^*$, where π and σ are finite sequences of modes. (This is because an infinite path in a finite graph should go through the same vertex twice.) For all $q \in S_\eta^*$, we can compute statically such a quasi-cyclic sequence starting at q. Let us denote it by $\varphi(q)$. This induces a safety controller for the real system Σ as follows: given a state $x \in S^*$, find a state $q \in S_\eta^*$ such that $\|x - q\| \leq \eta$, then apply the sequence $\varphi(q)$ to x. It follows from theorem 3.1 that, under such a control, the state of Σ always stays in $\mathcal{B}(S^*, \varepsilon)$.

Note that the correctness of the synthesis of a safety controller for S at the continuous state level relies on theorem 3.1. Actually, we can relax the assumption of contractivity of the modes of Σ made in this theorem, and just assume the *local contractivity* of modes in S. We illustrate the method on the Boost DC–DC converter.

EXAMPLE 3.2.– The method is applied to the Boost converter with the same safe set as in example 3.1: $S = [3, 3.4] \times [1.5, 1.8]$ in plane (i_l, v_c), for the desired precision, we take $\varepsilon = 3.0$. It can be seen that the system is locally contractive in S with a contraction factor $\beta = 0.99202$. For the discretization parameter, we take $\eta = 1/40$ (which satisfies $\eta < \varepsilon(1 - \beta)$). See Figure 3.4 for one of the connected components of the graph of the automaton $\mathcal{A}_\eta$. Each cycle in the subgraph corresponds to a periodic switching rule of the converter which ensures that the electric variables lie inside the predefined S up to ε. For example, we consider the cycle passing through vertices numbered: $159, 243, 173, 257, 187, 271, 201,$ $285,$ $215, 299, 229, 159$. This corresponds to the application of the cyclic sequence of modes: $(1{\cdot}2{\cdot}1{\cdot}2{\cdot}1{\cdot}2{\cdot}1{\cdot}2{\cdot}1{\cdot}2{\cdot}2)^*$. A controlled trajectory starting at point $x_0 = (3.0, 1.79)$ is given in Figure 3.5. The box S is delimited by the dashed line. We can see that the system largely exceeds the limits of S (but stays inside the ε-approximation $\mathcal{B}(S,\varepsilon\,)$).

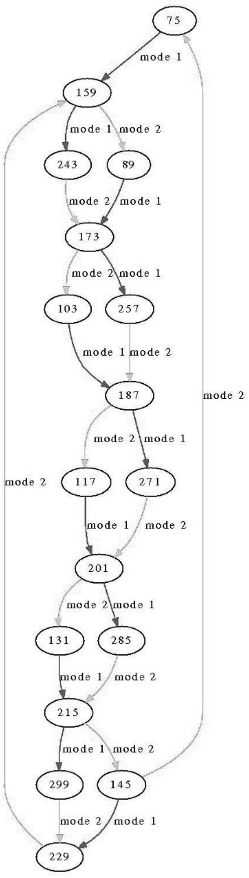


Figure 3.4. *Graph of an abstract safety controller of the Boost DC–DC converter, with* $\eta = \frac{1}{40}$ *and* $S = [3, 3.4] \times [1.5, 1.8]$. *For a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip*

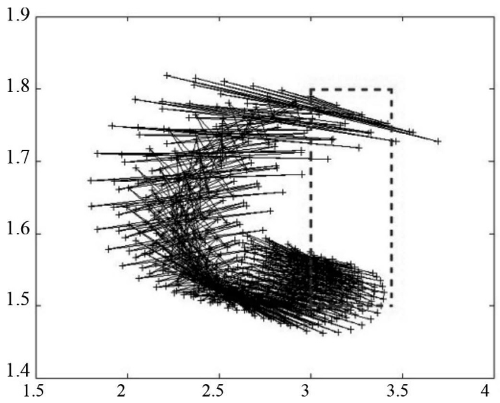


Figure 3.5. *Trajectory in plane* (i_l, v_c) *starting at* $x_0 = (3.0, 1.79)$ *controlled by switching rule* $(1 \cdot 2 \cdot 1 \cdot 2 \cdot 1 \cdot 2 \cdot 1 \cdot 2 \cdot 1 \cdot 2 \cdot 2)^*$ *found by the indirect method (precision* $\varepsilon = 3$*); the dashed box corresponds to* $S = [3, 3.4] \times [1.5, 1.8]$. *For a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip*

3.3. Application to a three-cell Boost DC–DC converter

We now apply the direct and indirect methods for synthesizing safety controllers for a bigger example: a Boost DC–DC converter with three cells. This is a real-life prototype built by the SATIE Electronics Laboratory (ENS Cachan) for the automotive industry. See Figure 3.6 for a picture of the system.

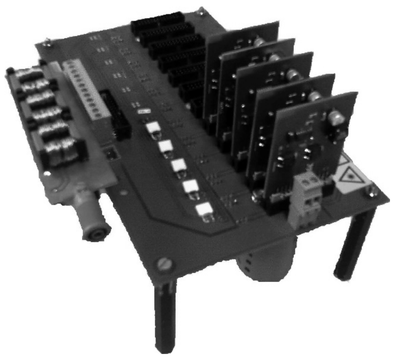


Figure 3.6. *Three-cell converter built by SATIE Electronics Laboratory*

3.3.1. *Model*

The Boost DC–DC converter with three cells relies on the same principle as the converter with one cell. An advantage of this system is its robustness: even if one switching cell is damaged, the system is still controllable with the restricted set of modes that remain available. This system is naturally more complex: there are four continuous variables of interest (instead of two), and $2^3 = 8$ modes (instead of two). Each mode is a triple $(\sigma_1\sigma_2\sigma_3)$, where σ_i indicates whether cell i is open ($\sigma_i = 0$) or closed ($\sigma_i = 1$). The electrical scheme is presented in Figure 3.7. An example of pattern is presented in Figure 3.8. The pattern is of the form $((100) \cdot (000) \cdot (010) \cdot (000) \cdot (001) \cdot (000))$ (or $(2 \cdot 1 \cdot 3 \cdot 1 \cdot 5 \cdot 1)$ under a decimal-like form), and corresponds to $(1 \cdot 0 \cdot 0 \cdot 0 \cdot 0 \cdot 0)$ for σ_1, $(0 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 0)$ for σ_2 and $(0 \cdot 0 \cdot 0 \cdot 0 \cdot 1 \cdot 0)$ for σ_3,

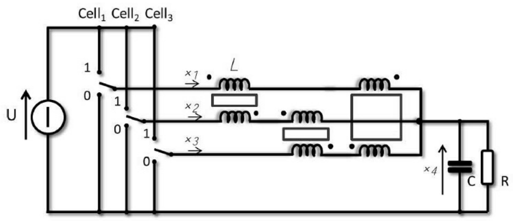


Figure 3.7. *Electrical scheme of the DC–DC converter with three cells*

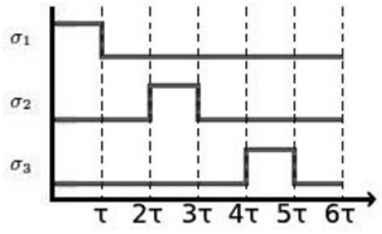


Figure 3.8. *Switching rule for the three-cell Boost DC–DC converter on one period of length* 6τ*,* $\sigma_1 = (1 \cdot 0^5)$*,* $\sigma_2 = (0^2 \cdot 1 \cdot 0^3)$ *and* $\sigma_3 = (0^4 \cdot 1 \cdot 0)$*, and the corresponding control pattern is* $(2 \cdot 1 \cdot 3 \cdot 1 \cdot 5 \cdot 1)$*). For a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip*

The system satisfies the following equations:

$$U\begin{pmatrix}\sigma_1\\\sigma_2\\\sigma_3\\0\end{pmatrix}+\begin{pmatrix}-2r & 0 & 0 & -1\\0 & -2r & 0 & -1\\0 & 0 & -2r & -1\\1 & 1 & 1 & -1/R\end{pmatrix}\begin{pmatrix}x_1\\x_2\\x_3\\x_4\end{pmatrix}=\begin{pmatrix}2L & -M & -M & 0\\-M & 2L & -M & 0\\-M & -M & 2L & 0\\0 & 0 & 0 & C\end{pmatrix}\begin{pmatrix}\dot{x}_1\\\dot{x}_2\\\dot{x}_3\\\dot{x}_4\end{pmatrix}.$$

This can be rewritten to fit our framework as:

$$\dot{x}=M_{LC}^{-1}M_Sx+b_\sigma$$

with

$$M_{LC}=\begin{pmatrix}2L & -M & -M & 0\\-M & 2L & -M & 0\\-M & -M & 2L & 0\\0 & 0 & 0 & C\end{pmatrix},\ M_S=\begin{pmatrix}-2r & 0 & 0 & -1\\0 & -2r & 0 & -1\\0 & 0 & -2r & -1\\1 & 1 & 1 & -1/R\end{pmatrix},$$

$$b_\sigma=UM_{LC}^{-1}\begin{pmatrix}\sigma_1\\\sigma_2\\\sigma_3\\0\end{pmatrix},$$

where U is the input voltage. We take in the following values per unit system: $U=100$, $L=10\times10^{-3}$, $M=9.9\times10^{-3}$, $r=100\times10^{-3}$, $R=1$, $C=300\times10^{-6}$ and $\tau=1/60,000$.

3.3.2. *Direct method*

For $S=[4,7]\times[4,7]\times[4,7]\times[15,17]$ and $\tau=1/60{,}000$, we can synthesize the maximal controlled invariant subset $S'\subset S$, using algorithm 3.1. A trajectory of the system starting at $x_0=(5,5,5,16)\in S'$ is presented in Figure 3.9. The figure shows that all the trajectories lie inside S.

3.3.3. *Indirect method*

For a safety region, we consider $S=[5.3,5.9]\times[5.3,5.9]\times[5.3,5.9]\times[15.5,16.5]$. It can be seen that the system is locally contractive in S with a contraction factor $\beta=0.99202$. For the state-space discretization, we take $\eta = 1/5$. This corresponds to a precision $\varepsilon = \eta/(1-\beta) \approx 21.6$. The abstract safety controller $\mathcal{A}_\eta$ associated with box S_η corresponds to a graph with several hundreds of vertices. A small part of the graph is given in Figure 3.10. For example, there is a cycle passing through vertices numbered: $290, 311,$ $332, 353,$ $332, 311,$ 290. This corresponds to the application of the cyclic sequence of modes: $(4 \cdot 4 \cdot 4 \cdot 1 \cdot 2 \cdot 1)^*$. For this control, the trajectory starting at point $x_0 = (5.4, 5.4, 5.4, 16)$ is given in Figure 3.11. We can see that the system does not stay inside the initial box S. However, we can check that the system stays within the ε-over-approximation $\mathcal{B}(S,\varepsilon)$ of S with $\varepsilon = 21.6$. The value of ε is too gross to give an interesting guarantee of safety. A finer precision ε would require a much smaller η and accordingly $\mathcal{A}_\eta$ with an overwhelming number of vertices. This tends to indicate that the indirect method (at least applied without further refinement) leads to prohibitively expensive computations for this example.

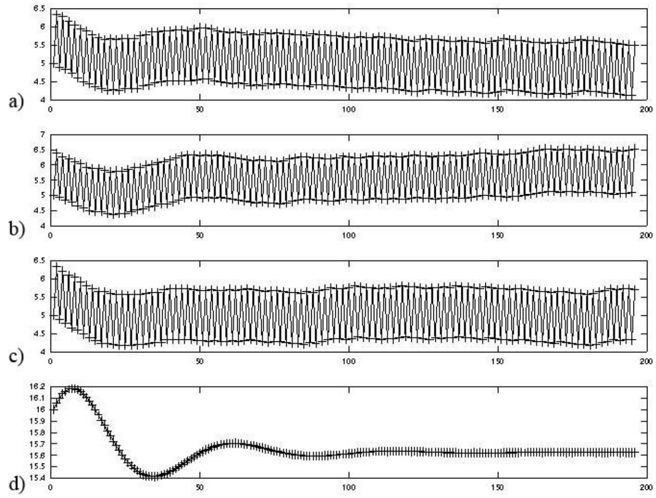


Figure 3.9. *Discrete-time trajectory of three-cell converter starting at* $x_0 = (5, 5, 5, 16)$ *in* $S = [4, 7] \times [4, 7] \times [4, 7] \times [15, 17]$ *using the direct control method (from a)–d):* x_1, x_2, x_3, x_4 *as a function of time).*

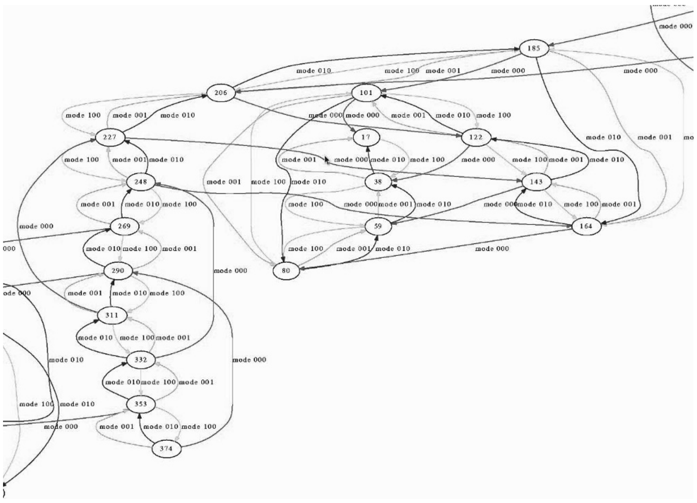


Figure 3.10. *Partial graph of an abstract safety controller of the three-cell Boost converter, with $\eta = \frac{1}{5}$. For a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip*

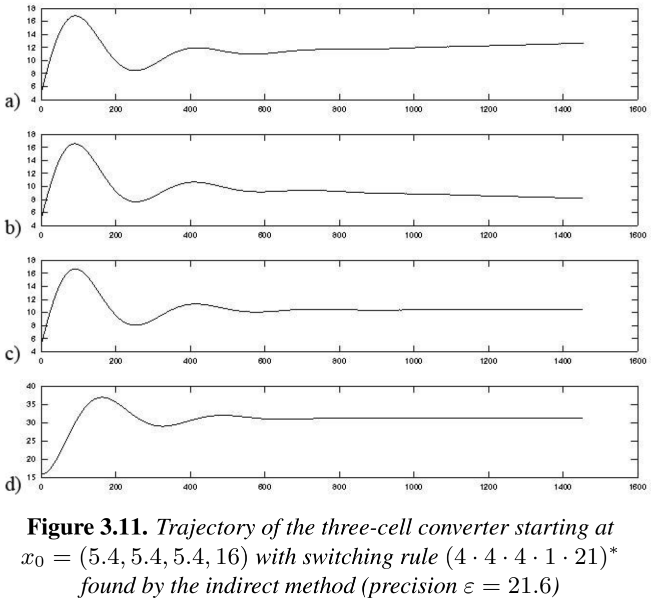


Figure 3.11. *Trajectory of the three-cell converter starting at $x_0 = (5.4, 5.4, 5.4, 16)$ with switching rule $(4 \cdot 4 \cdot 4 \cdot 1 \cdot 21)^*$ found by the indirect method (precision $\varepsilon = 21.6$)*

3.4. Notes

As explained above, the computation of maximal controlled invariant sets, found in the literature, relies essentially on a *backward* approach, computing iteratively the predecessor's reachable sets. This is in keeping with the seminal work of [RAM 89] for finite discrete systems. Unfortunately, as pointed out in [MIT 07], backward reachability constructs are more likely to suffer from *numerical stability* problems in systems displaying significant *contraction*, while contraction is generally a desirable and rewarding property of dynamic systems. In Chapter 4, we propose a *forward-oriented* approach that avoids these problems.

The *indirect* approach of controller synthesis, based on finite-state approximate models, originates from [RAI 98]. In the indirect approach, the switching rule can be computed *off-line,* while the switching rule has to be computed *online* in the direct approach. The compared merits of the direct versus indirect approaches for the synthesis of switching controllers for linear systems are more generally discussed in [ASA 00].

The notion of approximate bisimulation originates from [GIR 05b]. See also [TAB 05] and [TAB 09] for an exposition of several classes of hybrid systems admitting abstract models, along with the relationships between them.

4

Stability Controllers

In this chapter, we are interested in the problem of practical stabilization: given a region R, find a switching rule that makes the system converge to a region located inside R. Such a switching rule corresponds to a *stability controller*. It is interesting to confine the trajectories in a region that is as small as possible. The problem is closely related to the problem of finding a controlled invariant subset of R that is as small as possible, ideally *minimal*. We present a direct forward-oriented method that *decomposes* a given state region R, and induces a state-dependent control that makes the trajectories of the system converge to finite sets of points that, under certain conditions, correspond to *limit cycles*. The method can also be used for synthesizing *safety controllers* in order to prove safety properties.

After explaining our aim (section 4.1), we first give some formal preliminaries (section 4.2). We then present the decomposition method (section 4.3): the basic procedure is discussed in section 4.3.1 and its enhancement for proving safety properties is discussed in section 4.3.2. We then apply the method to the synthesis of finite controlled invariants and limit cycles that attract the trajectories of the controlled system (section 4.4). Some information on the implementation of the decomposition procedure is discussed in section 4.5.

4.1. Motivation

Let us consider Figure 3.1. The set $S = [3.0, 3.4] \times [1.5, 1.8]$ is divided into a maximal controlled subset $S^* = P_1 \cup P_2$, made of up two polyhedra P_1 and P_2, and an "uncontrollable" part colored in black made of up two parts: a lower left part, say Q_1, and an upper right part, say Q_2, of S. Each polyhedron P_1, P_2, Q_1 and Q_2 contains one (and only one) corner of S. The corner of S belonging to P_1 (respectively, P_2) is controllable: if we apply mode 1 (respectively 2) to this corner, we find a point belonging to $P_1 \cup P_2 \subset S$. In contrast, the corners of S belonging to Q_1 and Q_2 are not controllable: the application of either mode 1 or mode 2 maps these corners in points located outside S. Now, let us ask the question: does there exist a k*-pattern*, that is a sequence of modes of length (at most) k, mapping the corners to points located *inside* S? If such patterns exist, we can say that the corners are "k-controllable". In this example, we can see that it is the case for $k = 5$. Besides, if we divide S into four sub-boxes of equal size, say $V_1, \dots, V_4$, each containing a corner, say $C_1, \dots, C_4$, of S, we can see that the pattern, say π_i, that maps C_i inside S also maps the whole sub-box V_i inside S ($1 \leq i \leq 4$). This suggests a procedure of *decomposition* that splits S by bisection into sub-boxes, and looks for patterns that map the sub-boxes inside S. If this succeeds, we say that S is "k-controllable" or is a "controlled k-invariant set". The decomposition induces a *state-dependent control* strategy that makes any point of S return to S after at most k steps. In the following, we formalize these ideas.

4.2. Preliminaries

DEFINITION 4.1.– *Given a set $R \subset \mathbb{R}^n$ and a set $\{(V_i, \pi_i)\}_{i \in I}$, where I is a finite set of indices, V_i is a subset of $\mathbb{R}^n$ (for all $i \in I$) and π_i is a k-pattern (for all $i \in I$), we say that $\Delta = \{(V_i, \pi_i)\}_{i \in I}$ is a* k-invariant decomposition of R *if:*

– *$R = \bigcup_{i \in I} V_i$;*

– *V_i is R-invariant via π_i (i.e. $Post_{\pi_i}(V_i) \subset R$), for all $i \in I$.*

In the remaining chapter, we will suppose that R is a *box* (i.e. a rectangular region of $\mathbb{R}^n$). Such a set R will be referred to as a *global (control) box*. The subsets V_is (with $i \in I$) will be boxes that are included into R. They will be referred to as *local (control) boxes*.

Given a box $R \subset \mathbb{R}^n$ and a set Δ of the form $\{(V_i, \pi_i)\}_{i \in I}$ with $\bigcup_{i \in I} V_i = R$, we define $Post_\Delta$ as follows:

$$Post_\Delta(X) = \bigcup_{i \in I} Post_{\pi_i}(X \cap V_i), \quad \text{for all } X \subset R.$$

It is easy to show:

PROPOSITION 4.1.– Given a box $R \subset \mathbb{R}^n$ and a set $\Delta : \{(V_i, \pi_i)\}_{i \in I}$ where the π_is ($i \in I$) are k-patterns, and with $\bigcup_{i \in I} V_i = R$, we have:

– Δ is a k-invariant decomposition of R iff $Post_\Delta(R) \subset R$.

– The image of a (compact) convex set by $Post_\Delta$ is a finite set of (compact) convex sets.

NOTE.– For the sake of simplicity, we will use $Post_\Delta(x)$ instead of $Post_\Delta(\{x\})$, when x is a point of $\mathbb{R}^n$.

EXAMPLE 4.1.– (BOOST DC–DC CONVERTER).– Let us consider example 2.1 (see Chapter 2). In the case of the Boost DC–DC converter, we can show that, for $R = [1.55, 2.15] \times [1.0, 1.4]$, there is a decomposition $\Delta = \{(V_i, \pi_i)\}_{i=1,\dots,4}$ with $V_1 = [1.55, 1.85] \times [1.0, 1.2]$, $V_2 = [1.85, 2.15] \times [1.0, 1.2]$, $V_3 = [1.85, 2.15] \times [1.2, 1.4]$, $V_4 = [1.55, 1.85] \times [1.2, 1.4]$ and $\pi_1 = (1 \cdot 1 \cdot 2 \cdot 2 \cdot 2)$, $\pi_2 = (2)$, $\pi_3 = (2 \cdot 1 \cdot 2)$, $\pi_4 = (1)$. We can check indeed that, for all $1 \leq i \leq 4$, $Post_{\pi_i}(V_i) \subset R$. This is shown in Figure 4.1. In section 4.3, we will explain how to generate such a decomposition.

DEFINITION 4.2.– *Given a pattern π of the form $(u_1 \cdots u_m)$, and a set X; the* unfolding of X via π, *denoted by $Unf_\pi(X)$, is the set $\bigcup_{i=0}^m X_i$ with:*

– $X_0 = X$,

– $X_{i+1} = Post_{u_{i+1}}(X_i)$, *for all* $0 \leq i \leq m - 1$.

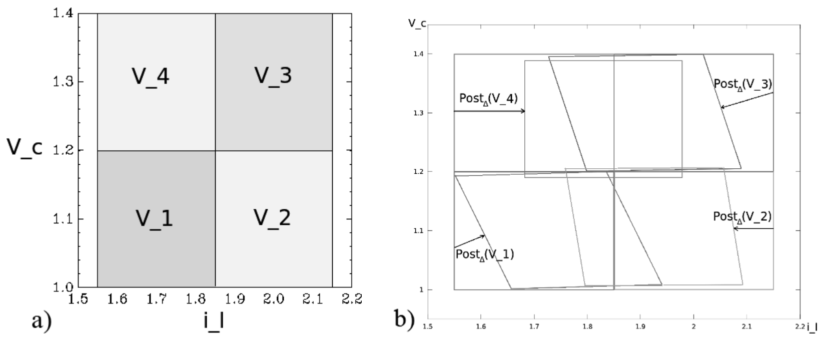


Figure 4.1. *Decomposition Δ of $R = [1.55, 2.15] \times [1.0, 1.4]$ for the Boost DC–DC converter example a), and visualization of $Post_\Delta(V_i) \subset R$, $i = 1, \ldots, 4$ b)*

DEFINITION 4.3.– *Consider a k-invariant box R of decomposition $\Delta = \{(V_i, \pi_i)\}_{i \in I}$. The Δ-*unfolding *of R, denoted by $Unf_\Delta(R)$, is the set:*

$$\bigcup_{i \in I} Unf_{\pi_i}(V_i).$$

EXAMPLE 4.2.– Figure 4.2 shows the unfolding of R for the decomposition Δ of example 4.1, where dark gray (respectively, light gray) indicates that mode 1 (respectively 2) applies.

PROPOSITION 4.2.– Suppose that a box R has a k-invariant decomposition Δ. Then, the Δ-unfolding of R is controlled invariant.

PROOF.– Let us explain how such a control can be refined in order to make R', the Δ-unfolding of R, *controlled invariant*. We extend Δ: $\{(V_i, \pi_i)\}_{i \in I}$ as follows. Each element of Δ is of the form (V,π) where π is of the form $(u^1 u^2 \cdots u^m)$. Such an element is replaced by m couples (V^1, u^1), (V^2, u^2), $\ldots$, (V^m, u^m) with $V^1 = V$, $V^2 = Post_{u^1}(V^1)$,$\ldots$, $V^m = Post_{u^{m-1}}(V^{m-1})$. The decomposition Δ becomes a decomposition Δ' of the form $\{(V_i^j, u_i^j)\}_{i,j}$ with $\bigcup_{i,j} V_i^j = R'$ and $Post_{u_i^j}(V_i^j) \subset R'$ for all i, j. Hence, for each

$x \in R'$, x belongs to some V_i^j and $Post_{u_i^j}(x) \subset R'$. This shows that R' is controlled invariant.

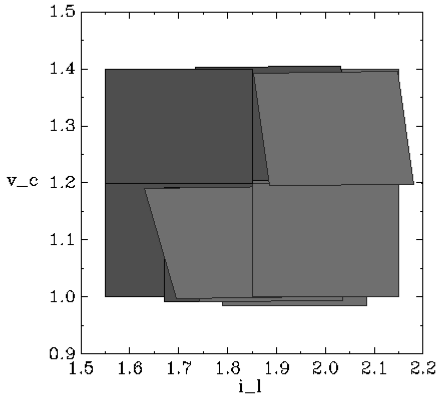


Figure 4.2. *Δ-unfolding of $R = [1.55, 2.15] \times [1.0, 1.4]$ in the Boost DC–DC converter example where dark gray (respectively, light gray) indicates that mode 1 (respectively, 2) applies*

4.2.1. *Control induced by the decomposition*

The decomposition Δ induces a state-dependent control that makes any trajectory starting from R go back to R within at most k steps: given a starting state x_0 in R, we know that $x_0 \in V_i$ for some $i \in I$ (since $R = \bigcup_{i \in I} V_i$); thus, we apply π_i to x_0, which gives a new state x_1 that belongs to R (since V_i is R-invariant via π_i); the process is repeated on x_1, and so on iteratively. Given a point $x \in R$, we will denote by $succ_\Delta(x)$ the point of R obtained by applying π_i to x when x is in V_i. Note that a non-deterministic choice has to be made when a point x belongs to more than one local box V_i. We will suppose that we have an implicit selection function that operates a non-deterministic choice in such a case (e.g. we can select the set V_i of least index containing x). When x belongs to a single local box V_i, then $succ_\Delta(x) = Post_\Delta(x)$.

A sequence of points $\{x_i\}_{i \geq 0}$, with $x_{i+1} = succ_\Delta(x_i)$ for all $i \geq 0$, is called a *discrete trajectory induced by* Δ, or more simply, a Δ-*trajectory*[1].

We will also consider the *unfolding* of a Δ-trajectory, which corresponds to consider not only the successors of points via patterns, but also all the intermediate points generated by intermediate application of the modes forming the patterns. In Figures 4.3 and 4.9, for the sake of clarity, the points of Δ-trajectories will be linked together using straight lines, and similarly for their unfoldings.

EXAMPLE 4.3.– A Δ-trajectory starting from the left upper corner of $R = [1.55, 2.15] \times [1.0, 1.4]$ for the Boost example is shown in Figure 4.3 together with its unfolding.

Using proposition 4.2, we can prove safety properties of the controlled system, by showing $Unf_\Delta(R) \subset S$, where S is known to be a set of safe positions (see section 4.3.2).

4.3. Decomposition function

4.3.1. *Basic procedure*

We suppose that we are given a global box $R \subset \mathbb{R}^n$. We now give a decomposition procedure that generates a k-invariant decomposition of R as follows.

It first calls sub-function Find_Pattern in order to get a k-pattern such that R is R-invariant. If Find_Pattern succeeds, then we are done. Otherwise, it divides R into 2^n sub-boxes $V_1, \ldots, V_{2^n}$ of equal size. If, for each V_i, Find_Pattern gets a k-pattern making it R-invariant, it is done. If, for some V_j, no such pattern exists, the procedure is recursively applied to V_j. It ends with success when a k-invariant decomposition of R is found, or failure when the maximal degree d of decomposition is reached.

1 We will sometimes denote such a trajectory under the form: $x_0 \rightarrow_{\pi_{i_1}} x_1 \rightarrow_{\pi_{i_2}} \cdots$ with $i_1, i_2, \cdots \in I$.

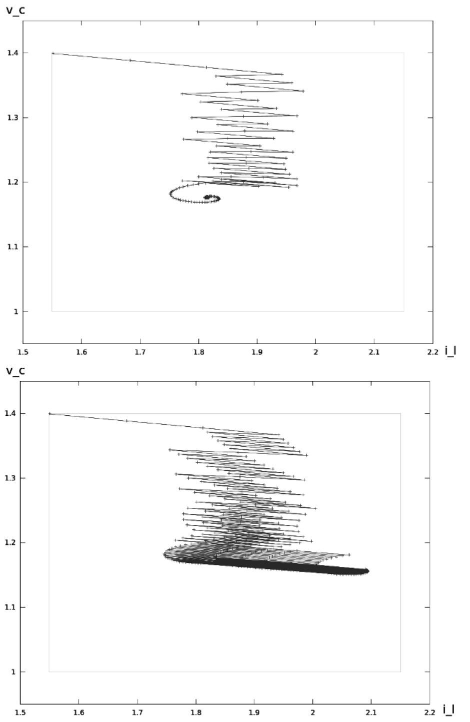


Figure 4.3. *Δ-trajectory for the a) Boost example and its b) unfolding*

REMARK 4.1.– Since the local boxes V_is are Cartesian products of closed intervals, two adjacent boxes V_i and V_j share a common facet.

The algorithmic form of the procedure is given in algorithms 4.1 and 4.2. (For the sake of simplicity, we consider the case of dimension $n = 2$, but the extension to $n > 2$ is straightforward.) The main function Decomposition (W,R,D,K) is called with R as input value for W, d for input value for D and k as input value for K; it returns either $\langle\{(V_i, \pi_i)\}_i, True\rangle$ with $\bigcup_i V_i = W$ and $\bigcup_i Post_{\pi_i}(V_i) \subset R$ or $\langle\_, False\rangle$. Function Find_Pattern(W,R,K) looks for a K-pattern for which W is R-invariant (i.e. $Post_\pi(W) \subset R$): it selects all the K-patterns by non-decreasing length order until either it finds such a pattern π (output: $\langle\pi, True\rangle$) or none exists (output: $\langle\_, False\rangle$).

REMARK 4.2.– Since R is a box, the inclusion test $Post_\pi(W) \subset R$ in function Find_Pattern(W,R,K) is implemented under the equivalent form $\square(Post_\pi(W)) \subset R$, which can be done in quadratic time (see section 2.3 and Appendix 5)).

The correctness of the procedure is stated as follows.

THEOREM 4.1.– If Decomposition(R,R,d,k) returns $\langle \Delta, True \rangle$, then Δ is a k-invariant decomposition of R. (Hence, $Unf_\Delta(R)$ is controlled invariant.)

As a whole, the complexity of function Find_Pattern is $O(n^3 N^k)$ since there are N^k patterns of length k, and the complexity of computation of successor states and inclusion test using zonotopes can be done in $O(kn^3)$. This procedure is called function Decomposition at most $2^{n \cdot d}$ times that corresponds to the number of sub-boxes in the case of a maximal decomposition of length d. The worst complexity of the procedure is thus in $O(2^{n \cdot d} N^k)$. Unsurprisingly, it suffers from the *curse of dimensionality* regarding not only the state dimension n, but also the depth of decomposition d and the length of patterns k.

The examples treated with a simple implementation of the procedure scales up to seven continuous variables (see section 4.5).

Note that there are boxes R so that Decomposition(R,R,d,k) does not succeed for any k and d. More generally, there are boxes R that are never k-invariant, for any k. In Appendix 1, we give a sufficient condition on the position of R for ensuring its k-invariance for some integer k.

4.3.2. *Enhancement for safety*

Let us now explain how to extend the decomposition procedure in order to show additionally that we have $Unf_\Delta(R) \subset S$, where S is a given safety set containing R. This is done by adding an extra input argument to Decomposition and Find_Pattern procedures corresponding to S. We then replace line 1 of Decomposition

(W,R,D,K,S) by Find_Pattern(W,R,K,S), and line 6 of function Find_Pattern by:

If $Post_\pi(W) \subset R$ ***And*** $Unf_\pi(W) \subset S$ **then**

Algorithm 4.1: *Decomposition(W,R,D,K)*

```
Input: A box $W$, a box $R$, a degree $D$ of decomposition and a
       length $K$ of pattern
Output: $\langle\{(V_i,\pi_i)\}_i, True\rangle$ with $\bigcup_i V_i = W$ and
        $\bigcup_i Post_{\pi_i}(V_i) \subset R$, or $\langle\_, False\rangle$
1  $(\pi, b) := Find\_Pattern(W, R, K)$
2  if $b = True$ then
3      return $\langle\{(W,\pi)\}, True\rangle$
4  else
5      if $D = 0$ then
6          return $\langle\_, False\rangle$
7      else
8          Divide equally $W$ into $(W_1, W_2, W_3, W_4)$   /* (case
               $n = 2$)                                    */
9          $(\Delta_1, b_1)$ := Decomposition($W_1$,$R$,$D-1$,$K$)
10         $(\Delta_2, b_2)$ := Decomposition($W_2$,$R$,$D-1$,$K$)
11         $(\Delta_3, b_3)$ := Decomposition($W_3$,$R$,$D-1$,$K$)
12         $(\Delta_4, b_4)$ := Decomposition($W_4$,$R$,$D-1$,$K$)
13         return $(\Delta_1 \cap \Delta_2 \cap \Delta_3 \cap \Delta_4, b_1 \wedge b_2 \wedge b_3 \wedge b_4)$
```

In other words, if π is a pattern of the form $(u_1 \cdots u_m)$ with $u_1, ..., u_m \in U$, we check additionally $W_i \subset S$ for all $1 \leq i \leq m$, where the W_is are the intermediate sets defined by $W_1 = Post_{u_1}(W)$, $\ldots$, $W_m = Post_{u_m}(W_{m-1})$. We have:

THEOREM 4.2.– If the function Decomposition(R,R,d,k,S) returns $\langle\Delta, True\rangle$, then $Unf_\Delta(R)$ is controlled invariant. Furthermore, we have $Unf_\Delta(R) \subset S$.

Algorithm 4.2: *Find_Pattern(W,R,K)*

```
Input: A box $W$, a box $R$ and a length $K$ of pattern
Output: $\langle \pi, True \rangle$ with $Post_\pi(W) \subset R$, or $\langle \_, False \rangle$ when no
        pattern maps $W$ into $R$
1 for $i = 1 \dots K$ do
2     $\Pi$ := set of patterns of length $i$
3     while $\Pi$ is non-empty do
4         Select $\pi$ in $\Pi$
5         $\Pi := \Pi \setminus \{\pi\}$
6         if $Post_\pi(W) \subset R$ then
7             return $\langle \pi, True \rangle$
8 return $\langle \_, False \rangle$
```

Hence, the system under the control inferred by the procedure (when it succeeds) is guaranteed to be safe.

The enhanced procedure has been implemented (see section 4.5 for details). We illustrate its application on the Boost DC–DC converter of example 2.1. Other examples are given in Appendix 2.

EXAMPLE 4.4.– For R, we now consider the box $[1.75, 1.95] \times [1.14, 1.26]$, which corresponds to a medium value 1.85 for i_l with ± 0.1 for variability and medium value 1.20 for v_c with ± 0.06 for variability. For the safety region, we take $S = [1.7, 2.0] \times [1.10, 1.30]$, which corresponds to an additional variability of ± 0.05 for i_l and ± 0.04 for v_c. The application of algorithm 4.1 to R and S, with $k = 10$ and $d = 4$ succeeds, yields a k-invariant decomposition Δ of the form $\{(V_j, \pi_j)\}_{j=1,\dots,16}$ of R^2 satisfying $Unf_\Delta(R) \subset S$. The k-invariant decomposition Δ of R is

2 The associated patterns are: $\pi_1 = (1122122122)$, $\pi_2 = (12121222)$, $\pi_3 = (12122122)$, $\pi_4 = (122)$, $\pi_5 = (2)$, $\pi_6 = (12)$, $\pi_7 = (12)$, $\pi_8 = (1)$, $\pi_9 = (1)$, $\pi_{10} = (1)$, $\pi_{11} = (12)$, $\pi_{12} = (12)$, $\pi_{13} = (2)$, $\pi_{14} = (2)$, $\pi_{15} = (12)$ and $\pi_{16} = (221)$.

shown in Figure 4.4, and the Δ-unfolding is shown in Figure 5.4(b): dark gray (respectively light gray) indicates that mode 1 (respectively 2) should be applied. The unfolded Δ-trajectory of the system starting at point $(1.75, 1.26)$ is shown in Figure 4.5.

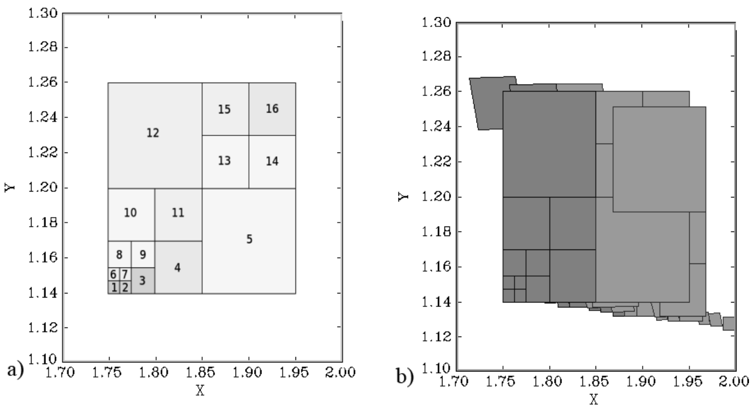


Figure 4.4. *a) Decomposition Δ for Boost converter of $R = [1.75, 1.95] \times [1.14, 1.26]$; b) Δ-unfolding where dark gray (respectively, light gray) indicates mode 1 (respectively, 2), with enclosing box $S = [1.7, 2.0] \times [1.1, 1.3]$*

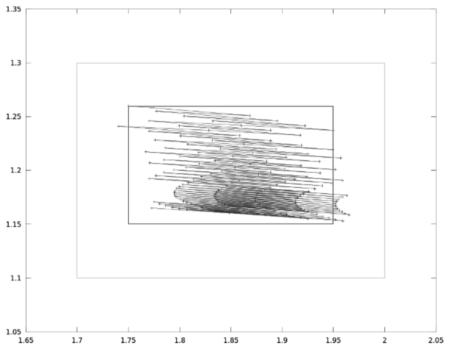


Figure 4.5. *Unfolded Δ-trajectory of the Boost converter starting at $(1.75, 1.26)$ (inner box $R = [1.75, 1.95] \times [1.14, 1.26]$ and outer box $S = [1.7, 2.0] \times [1.1, 1.3]$)*

4.4. Limit cycles

In Figure 4.3, we see that the (unfolding of the) Δ-trajectory seems to converge to a cycle. We now formally state that under certain assumptions, this is actually the case. We suppose that we are given a global box R and a decomposition $\Delta = \{(V_i, \pi_i)\}_{i \in I}$ produced by the decomposition algorithm of section 4.3. We denote the union of the set of borders of V_i ($i \in I$) by $\partial\Delta$. (Recall that two adjacent boxes V_i and V_j share a common border; see remark 4.1.)

We show how to produce *attractors* of R, using an iteration of $Post_\Delta$. Furthermore, under certain assumptions that are often encountered in practice, these attractors are made up of *finite* subsets of points corresponding to *limit cycles* that attract all the Δ-trajectories starting in R.

The idea is the following: since $Post_\Delta(R) \subset R$, we have $Post_\Delta^{i+1}(R) \subset Post_\Delta^i(R)$ for all $i \geq 0$, and the limit set $R_\Delta^* = \bigcap_{i \geq 0} Post_\Delta^i(R)$ is well defined and non-empty.

LEMMA 4.1.– Consider a k-invariant decomposition $\Delta = \{(V_i, \pi_i)\}_{i \in I}$ of R. The sequence $\{R_\Delta^j\}_{j \geq 0}$ defined by:

– $R_\Delta^0 = R$;

– $R_\Delta^{j+1} = Post_\Delta(R_\Delta^j)$

is a decreasing nested sequence and the set $R_\Delta^* = \bigcap_{j \geq 0} R_\Delta^j$ is well defined. Furthermore, R_Δ^* is an *attractor set* of R, that is:

1) $Post_\Delta(R_\Delta^*) = R_\Delta^*$ (*invariance* property);

2) $\forall x \in R,\ d(Post_\Delta^j(x), R_\Delta^*) \to 0$ as j tends to ∞[3] (*attraction* property).

3 $d(y, Z)$ denotes the smallest distance between a point $y \in \mathbb{R}^n$ and any point of $Z \subset \mathbb{R}^n$.

We now make the following assumptions:

H1): All the modes are locally contractive in R.

H2): There exists $N \in \mathbb{N}$ such that $Post^N_\Delta(R) \cap \partial\Delta = \emptyset$.

These assumptions will be discussed in section 4.4.1. We show that under these assumptions, R^*_Δ is a *finite* set of points composed of disjoint "cycles". Furthermore, each Δ-trajectory starting from a point of R converges to one of these cycles. This is formally stated as follows.

DEFINITION 4.4.– *A* cycle *is a finite set of points of R of the form $\{y_0, y_1, \dots, y_{m-1}\}$ with $y_0 \rightarrow_{\pi_{i_1}} y_1 \rightarrow_{\pi_{i_2}} \cdots \rightarrow_{\pi_{i_m}} y_m = y_0$ for some patterns $\pi_{i_1}, \dots, \pi_{i_m}$ of Δ.*

THEOREM 4.3.– Under assumptions H1 and H2, we have:

1) R^*_Δ is a finite union of (disjoint) cycles.

2) The Δ-unfolding of each cycle of R^*_Δ is a controlled invariant finite set.

3) Each Δ-trajectory $\{x_0, x_1, \dots\}$ *converges to a cycle* of the form $\{y_0, y_1, \dots, y_{m-1}\}$ in the following sense:

$$\exists M \in \mathbb{N}\ \forall \ell = 0, \dots, m-1\ \lim_{i \to \infty} x_{M+i\cdot m+\ell} = y_\ell.$$

The proof of theorem 4.3 is given in Appendix 3.

Note that, although the Δ-unfolding of each cycle is finite, it is not ensured to be a *minimal* controlled invariant: it is *a priori* possible that a strict subset of the Δ-unfolding be itself controlled invariant.

4.4.1. *Discussion of the assumptions H1 and H2*

Under assumptions H1 and H2, we have stated that each trajectory converges to a finite cyclic set of points.

The first assumption is classical in order to ensure convergence results, and was also used in Chapter 3. Actually, even if some of the modes are not contractive, we may observe the convergence to finite cyclic sets of points because of the contractivity of the patterns involved by the control. This is the case in the helicopter motion example (see Figure 7.5). However, if none of the modes are (locally) contractive, then there is no limit cycle and R^*_Δ is infinite (e.g. see Appendix 4).

Assumption H2 is justified as follows. When $Post_\Delta$ is applied to R for the first time, the local boxes are transformed into $|I|$ convex sets. If such a set, say W, crosses a border of $\partial\Delta$ and partly belongs to, say, two local boxes V_1 and V_2, it will be split into two sets $Post_{\pi_1}(W \cap V_1)$ and $Post_{\pi_2}(W \cap V_2)$ at the next application of $Post_\Delta$. The number of convex sets generated at each application of $Post_\Delta$ thus increases repeatedly until no image crosses a border, which happens at step N by assumption H2. The images generated by further application of $Post_\Delta$ will then never cross $\partial\Delta$: these images will either disappear or shrink toward single points by assumption H1. In the absence of assumption (H2), the number of connected sets of $Post^k_\Delta(R)$ can increase indefinitely. Thus, assumption H2 seems to be a necessary condition for the finiteness of R^*_Δ.

4.4.2. *Illustrative examples*

We now illustrate the convergence of $Post^k_\Delta$ to a cyclic set of points as k tends to infinity, on the Boost and two-tank example.

EXAMPLE 4.5.– (BOOST DC–DC CONVERTER).– We have already seen that the modes of the Boost converter are locally contractive (see example 3.2), hence H1 is satisfied. Likewise, H2 is satisfied: for $N = 100$, $Post^N_\Delta(R)$ is entirely contained into the local box V_1 of the decomposition Δ (see Figure 4.6 showing the iterated images $Post^k_\Delta$ for $k = 0, 20, 40, 60, 80, 100$). The limit set R^*_Δ is here composed of a unique limit cycle that is made of a single point $y_0 \in V_1$. We have $y_0 \to_{\pi_1} y_1 = y_0$, with $\pi_1 = (1 \cdot 1 \cdot 2 \cdot 2 \cdot 2)$. The Δ-unfolding of this limit cycle is thus made up of 5 points corresponding to the composing modes of π_1 (see Figure 4.7).

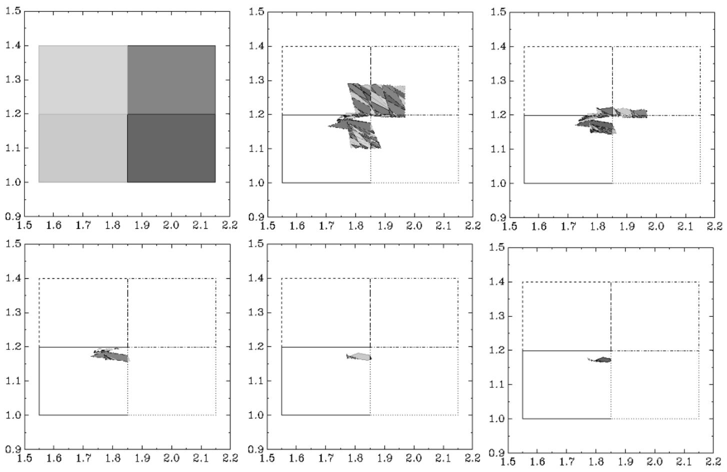


Figure 4.6. *Visualization of $Post^k_{\Delta}$ for $k = 0, 20, 40, 60, 80, 100$*

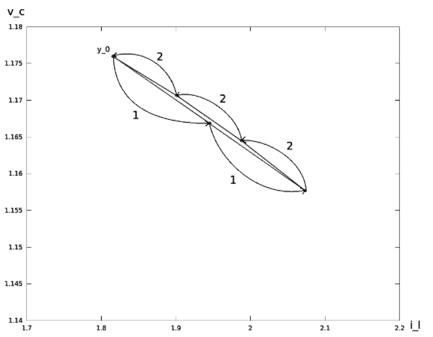


Figure 4.7. *Δ-unfolding of the limit cycle $\{y_0\}$ for the Boost example*

EXAMPLE 4.6.– (TWO-TANK SYSTEM).– The two-tank system example is taken from [HIS 01]. The system consists of two tanks and two valves. The first valve adds to the inflow of tanks 1 and the second valve is a drain valve for tank 2. There is also a constant outflow from tank 2 caused by a pump. The system is linearized at a desired operating point. The objective is to keep the water level in both tanks within limits using a discrete open/close switching strategy for the valves. Let the water level of tanks 1 and 2 be given by x_1 and x_2, respectively. The behavior of x_1 is given by $\dot{x}_1 = -x_1 - 2$ when the tank 1 valve is closed, and $\dot{x}_1 = -x_1 + 3$ when it is closed. Likewise, x_2 is driven by $\dot{x}_2 = x_1$ when the tank 2 valve is closed and $\dot{x}_2 = x_1 - x_2 - 5$ when it is closed. Using $R = [-1.5, 2.5] \times [-0.5, 1.5]$ as a control box, we obtain the decomposition shown in Figure 4.8. With $V_1 = [-1.5, 0.5] \times [-0.5, 0.5]$ associated with pattern $\pi_1 = (2 \cdot 3 \cdot 3)$, $V_2 = [0.5, 2.5] \times [-0.5, 0.5]$ to $\pi_2 = (2), V_3 = [0.5, 2.5] \times [0.5, 1.5]$ to $\pi_3 = (1 \cdot 4)$ and $V_4 = [-1.5, 0.5] \times [0.5, 1.5]$ to $\pi_4 = 3$. Figure 4.9 shows a discrete trajectory of the two-tank system and its Δ-unfolding.

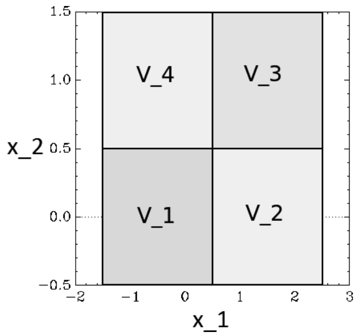


Figure 4.8. *Decomposition for the two-tank problem*

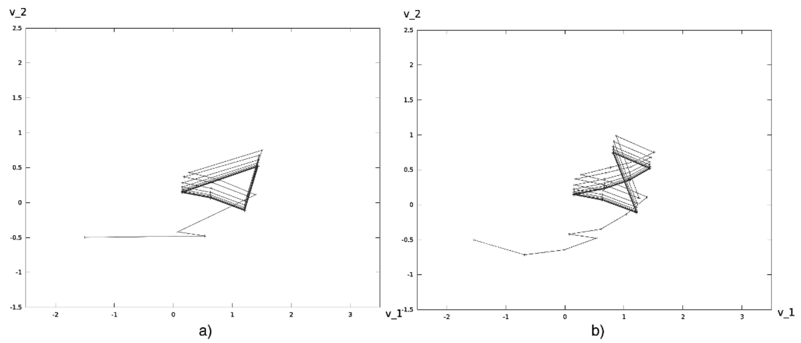


Figure 4.9. Δ*-trajectory starting from the bottom left corner of R a), and its* Δ*-unfolding b)*

We can check that all the modes of the two-tank system are contractive (the eigenvalues of all the associated matrix have a negative real part). Hence, assumption H1 is satisfied. Likewise, H2 is satisfied: for $N = 10$, $Post^N_\Delta(R)$ does not intersect the borders of Δ (see Figure 4.10 showing the iterated images $Post^k_\Delta(R)$, for $k = 0, 5, 10, 15, 20, 25$).

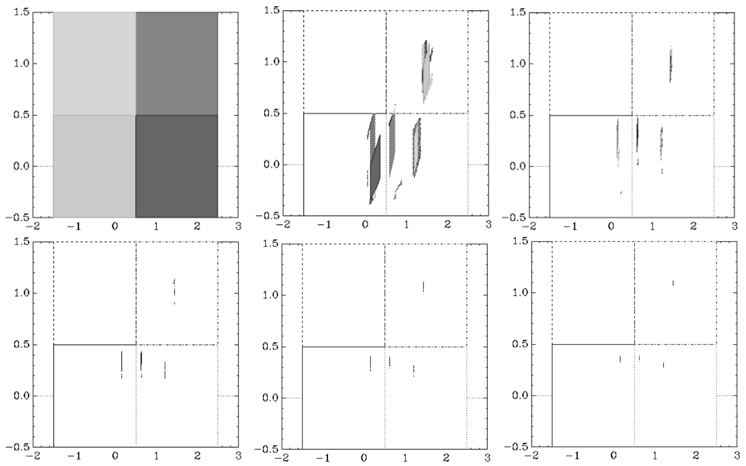


Figure 4.10. *Visualization of* $Post^k_\Delta$ *for* $k = 0, 5, 10, 15, 20, 25$

Here, the limit set R^*_Δ is composed of a unique limit cycle of the form $\{y_0, y_1, y_2, y_3\}$ with $y_0 \rightarrow_{\pi_2} y_1 \rightarrow_{\pi_2} y_2 \rightarrow_{\pi_1} y_3 \rightarrow_{\pi_3} y_4 = y_0$ (with $\pi_2 = (1), \pi_1 = (2{\cdot}2{\cdot}3), \pi_3 = (1{\cdot}4)$). This limit cycle is shown in Figure 4.11, and its Δ-unfolding (corresponding to 7 points generated by the composing modes of $\pi_2\pi_2\pi_1\pi_3$) is shown in Figure 4.11b).

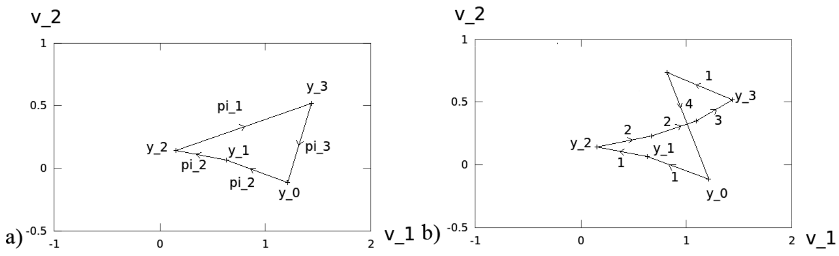


Figure 4.11. *Limit cycle for the a) two-tank example and its b) Δ-unfolding*

4.5. Implementation

The implementation of the method is made up of two basic procedures: a function Decomposition described in section 4.3.1, written in Octave [OCT 13], that outputs Δ, and a procedure called Iteration that constructs R^i_Δ for $i \geq 0$, written in Ocaml [OCA 13]. The function Decomposition is implemented using zonotopes. The code is given in Appendix 8.

We cannot implement the procedure Iteration using zonotopes because it involves the intersection operator that does not preserve the structure of zonotopes. It is thus implemented using the more general structure of polyhedra using the PPL library [PPL 13]. The Iteration procedure receives Δ from module Decomposition and outputs the successive iterations of $Post_\Delta$. The sequence of post sets can also be visualized as an animation (see Figure 4.6).

The examples of decomposition given in this book have been performed using the code in Octave given in Appendix 5, except the multilevel examples (see Chapter 5). The multilevel examples have been performed using a code written in PLECS [PLE 13], which is better-suited for these examples, because in PLECS we can enter the electrical circuits under a schematic form and obtain automatically the associated systems of differential equations. The examples have run on a machine equipped with an Intel Core2 CPU X6800 at 2.93 GHz with 2 GB of RAM memory. Some of the experiments are listed in the following table.

Example	Running time	No. of patterns	N	k	d	n	Contrac.	Cycle
Boost (exercise 4.4)	150 s	12,113	2	10	4	2	Loc.	Yes
Helicopter (exercise A2.1)	$\approx$ 2 h	$\approx$ 1.5 million	9	6	4	2	No	Yes
Heating (exercise A2.2)	1 s	134	2	2	4	2	Glob.	Yes
Two-tank (exercise 4.6)	4 s	1,423	4	3	1	2	Glob.	Yes
5-level (section 5.2.1)	3 min	-	16	8	1	3	Yes	Yes
7-level (section 5.2.2)	35 min	-	64	32	1	5	Yes	Yes
9-level (section 5.2.2)	$\approx$ 5 h	-	256	128	1	7	Yes	Yes

The first column indicates the name of the example together with its reference in the book. The second column indicates the running time to obtain a decomposition, and the third column indicates the numbers of patterns generated to obtain this decomposition[4]. The subsequent columns labeled by N, k, d and n indicate the number of modes, the input parameter of maximal pattern length, the input parameter of decomposition depth and the space dimension, respectively. Finally, the column "contrac." indicates if the example is locally contractive (loc.), globally contractive ("glob") or not contractive, and the column "cycle" indicates if the procedure Iteration generates a limit cycle.

4.6. Notes

The method of proving controlled invariance presented here, based on a decomposition of the state space, is original. This method presents some similarities to the *box invariance* method of Abate *et al.* [ABA 09] that exhibit rectangular invariant subregion of affine hybrid systems containing an equilibrium point. The process of state decomposition by dichotomy (or "bisection") has been used in [JAU 01] for the purpose of *set inversion* and applied to a robust control and stability analysis.

As pointed out above, the limit cycle generated is not necessarily a minimal invariant. Actually, it is difficult to define an appropriate notion of size for a minimal invariant set. In the framework of discrete linear time-invariant (LTI) with quantized input, Picasso and Bicchi [PIC 08] have used hypercubes in order to provide a lower bound for the minimal feasible size of an invariant set.

The presence of limit cycles in switched systems has often been observed in the context of power electronics (see [PAT 09] for example). Various methods are generally used for proving their existence and stability: Lyapunov techniques (see [RUB 00] [BUI 05] for example); Poincaré map technique (see [GON 03, HIS 01] for example); sensibility functions [FLI 06] or describing functions [SAN 93].

4 This figure is not available for the multilevel converters because they have been implemented using PLECS, rather than our standard code in Octave.

5 Application to Multilevel Converters

Power converters play an important role in the field of renewable energy: they are used to connect renewable sources to power grids, and optimize the efficiency of solar panels and wind generators. Switched control has gained much attention recently in the field of high-order converters, due to its property of being easily implemented. In some topologies, there is, however, a dramatic increase in the number of switches, which entails an increasing number of degrees of freedom, and complicates the controller design (see [CER 09]). There is therefore a wide variety of applications for formal methods in order to produce correct-by-design control methods.

In this chapter, we consider the design of control policies for power converters with five and seven levels. The objective is to design a switching signal forcing the output voltage to be a “quantized” sinusoidal signal while ensuring that the voltages across the capacitors in the power converter remain within predefined safety ranges. The staircase waveform is achieved by controlling four (respectively, six) switches, which are used to divide the incoming voltage into two paths and produce different levels of incoming voltage with the help of three (respectively, five capacitors). We adapt the decomposition procedure explained in Chapter 4, in order to synthesize a controller that guarantees that the electrical state parameters will always stay within a predefined safe zone of variations. The synthesized controllers have been validated on real hardware on prototypes built by the SATIE Electronics Laboratory.

In section 5.1, we explain the principle and architecture of multilevel converters. In section 5.2, we apply the decomposition method to the control of five-level and seven-level converters, and give numerical simulations. In section 5.3, we present physical experimentations done with a prototype built by the SATIE Laboratory.

5.1. Multilevel converters

The general function of a multilevel power converter is to synthesize a desired voltage from several levels of DC voltage. For this reason, multilevel power converters can easily provide the high power required by large electric drive systems. Schematically, a multilevel converter is made up of capacitors and switching cells (as well as opposite switching cells that are in complementary positions). According to the positions of the cells (the high-side switch conducting position is indicated by 1 and the low-side switch conducting position by 0), we are able to fraction the load voltage. By controlling the global position of the switches during a simple fixed time-stepping procedure, it is then possible to generate a staircase voltage with levels that approximate a triangular or sinusoidal waveform (see Figure 5.1 for $\ell = 5$). The problem that arises is to select the appropriate switching control strategy among a number of combinations of switch positions, which increase exponentially with the number of levels (and pairs of switches). A crucial additional difficulty comes from the fact that, in order to be admissible, the control of the switching cells must guarantee that the voltages across the cell-capacitors are constrained within a certain range defined by the device blocking voltage rating. The control must then guarantee a *safety property*, called "capacitor voltage balancing": the voltage of each individual capacitor should stay inside a limited predefined interval.

5.2. Application of the decomposition procedure

We apply a simplified version of the decomposition procedure given in section 4.3.1 in the context of this case study. The procedure can be simplified here because, as explained in the following only a restricted number of patterns allow us to produce a staircase output signal of ℓ levels. These admissible patterns correspond to paths in a predefined graph. They have all the same length $k = 2(\ell - 1)$. Besides, the procedure succeeds by simple bisection of the input control box R into $2^{\ell-1}$ sub-boxes (which corresponds to value $D = 2$ for input depth parameter of algorithm 4.1).

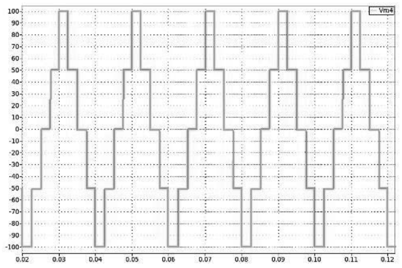


Figure 5.1. *Staircase output voltage waveform for a five-level converter*

5.2.1. *Five-level converter*

There are different possible topologies for multilevel power converters: neutral-point clamped, cascaded H-bridge, flying capacitor, etc. We focus here on the flying capacitor topology [MEY 92]. The electrical scheme of a five-level converter is given in Figure 5.2. There are four pairs of switching cells $S_1, \ldots, S_4$ and three capacitors $C_1, \ldots, C_3$. The state of the system is $x(t) = [v_1(t)\ v_2(t)\ v_3(t)\ i(t)]^T$ where $v_i(t)$ is the voltage across C_i $(1 \leq i \leq 3)$ and $i(t)$ is the current flowing in the circuit. The duration of a cycle is $T = 8\tau$. The *mode* of the system is characterized by the value (0 or 1) of the switching cells, that is by the value of vector $S = [S_1\ S_2\ S_3\ S_4]^T$[1]. There are thus $2^4 = 16$ modes. A mode S induces an output voltage of value $-v_{low} + (\sum_{i=1}^{4} S_i)v_{high}/2$, where v_{low} and v_{high} are the input voltages of low level and high level, respectively. The system thus outputs five different levels of voltage, which go from $-v_{low}$ up to $+v_{high}$ with

1 Besides, we have: $S_5 = \neg S_1$, $S_6 = \neg S_2$, $S_7 = \neg S_3$ and $S_8 = \neg S_4$.

steps at $-1, -\frac{1}{2}, 0, \frac{1}{2}, 1$. The ideal value v_i^* of the voltage across capacitor C_i $(1 \leq i \leq 3)$ depends on the values of v_{low} and v_{high}. Here, we use: $v_{low} = v_{high} = 100\,\mathrm{V}$, and $v_1^* = 150\,\mathrm{V}$, $v_2^* = 100\,\mathrm{V}$, $v_3^* = 50\,\mathrm{V}$. The five-level converter can be seen as a switched system. Given a mode S, the associated dynamics is of the form $\dot{x}(t) = A_S x(t) + b_S$ with:

$$A_S = \begin{pmatrix} -\frac{1}{R_1C_1} & 0 & 0 & \frac{S_1-S_2}{C_1} \\ 0 & -\frac{1}{R_2C_2} & 0 & \frac{S_2-S_3}{C_2} \\ 0 & 0 & -\frac{1}{R_3C_3} & \frac{S_3-S_4}{C_3} \\ \frac{S_2-S_1}{L_{Load}} & \frac{S_3-S_2}{L_{Load}} & \frac{S_4-S_3}{L_{Load}} & -\frac{R_{Load}}{L_{Load}} \end{pmatrix} \text{ and } b_S = \begin{pmatrix} 0 \\ 0 \\ 0 \\ S_1 \frac{v_{high}+v_{low}}{L_{Load}} \end{pmatrix}.$$

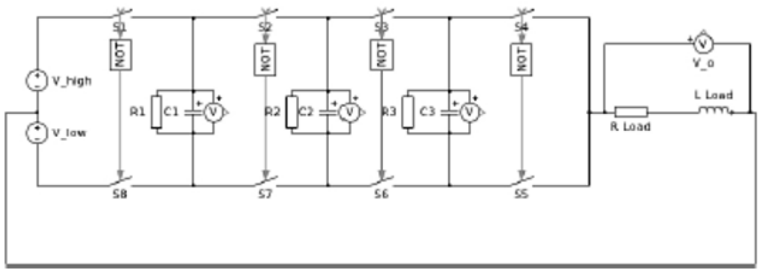


Figure 5.2. *Electrical scheme of a five-level converter*

By controlling the modes at each sampling time, we can synthesize a five-level staircase function. Not all the transitions between modes are admissible: we are allowed to switch only one (pair of) cell(s) at a time. The graph of admissible transitions during a cycle is shown in Figure 5.3. The nodes of the graph are labeled by the modes. Each path represents a possible sequence of control for one cycle, leading from voltage $-v_{low}$ (state 0000) to voltage $+v_{high}$ (state 1111) through voltages $-\frac{1}{2}.v_{low}$, 0, $\frac{1}{2}.v_{high}$ then back to voltage $-v_{low}$ (state 0000) through voltages $\frac{1}{2}.v_{high}$, 0, $\frac{1}{2}.v_{low}$. There are thus 576 possible sequences of control for generating a five-level staircase signal on one cycle. These sequences of control correspond to patterns of length 8, denoted by $\pi_1, \ldots, \pi_{576}$. The control problem is now to find a strategy for deciding, at each beginning of cycle, which π_i $(1 \leq i \leq 576)$ to apply in order to maintain all the capacitor voltages within a predefined limited zone.

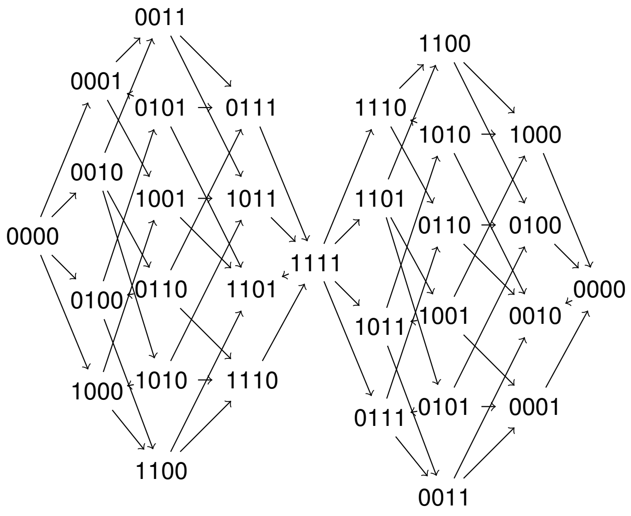


Figure 5.3. *Transition graph corresponding to a cycle of five-level staircase signal*

We will use the numerical values: $R_{Load} = 50\Omega$, $C_1 = C_2 = C_3 = 0.0012$ F, $L_{Load} = 0.2$ H, $R_1 = R_2 = R_3 = 20{,}000\Omega$, $T = 8\tau = 0.02$ s (which correspond to a frequency of 50 Hz). The five-level inverter outputs ideally a staircase waveform with an amplitude of 200 V, centered around 0 V. We consider that a variation of ± 5 V is admissible as it represents a variation of 10% on the least charged capacitor C_3. It is interesting to note that the beginning of each cycle the value of i is null. This suggests to look for a state-dependent control that depends only on the capacitor voltages v_1, v_2, v_2, and not on the value of i. We will thus focus on the voltage dimensions of the control box R and disregard its intensity dimension. For R, we take $R = [145, 155] \times [95, 105] \times [45, 55]$, which corresponds to a product of intervals centered around the ideal values with a variation of ± 5 V (i.e. 10% of the least charged capacitor C_3). For S, we take $R + \varepsilon$ with $\varepsilon = 1\,\text{V}$, which means that we have an additional tolerance of $\pm 1\,\text{V}$ for the fluctuations occurring between two beginnings of cycle. The decomposition procedure is thus adapted as follows. We decompose R into eight subsets V_i of equal size:

– $V_1 = [145, 150] \times [95, 100] \times [45, 50]$;

– $V_2 = [145, 150] \times [95, 100] \times [50, 55]$;

– $V_3 = [145, 150] \times [100, 105] \times [45, 50]$;

– $V_4 = [145, 150] \times [100, 105] \times [50, 55]$;

– $V_5 = [150, 155] \times [95, 100] \times [45, 50]$;

– $V_6 = [150, 155] \times [95, 100] \times [50, 55]$;

– $V_7 = [150, 155] \times [100, 105] \times [45, 50]$;

– $V_8 = [150, 155] \times [100, 105] \times [50, 55]$.

Using a standard random generate-and-test program, we find patterns π_j $(1 \leq j \leq 8)$, which, applied to points of V_j, generate points that are all contained in S. The patterns π_js correspond to the following paths of the transition graph:

– π_1: $(0000 \rightarrow 0001 \rightarrow 0101 \rightarrow 1101 \rightarrow 1111 \rightarrow 1101 \rightarrow 0101 \rightarrow 0001 \rightarrow 0000)$;

– π_2: $(0000 \rightarrow 0100 \rightarrow 0101 \rightarrow 1101 \rightarrow 1111 \rightarrow 1101 \rightarrow 0101 \rightarrow 0100 \rightarrow 0000)$;

– π_3: $(0000 \rightarrow 0001 \rightarrow 0011 \rightarrow 1011 \rightarrow 1111 \rightarrow 1011 \rightarrow 0011 \rightarrow 0001 \rightarrow 0000)$;

– π_4: $(0000 \rightarrow 0010 \rightarrow 0011 \rightarrow 1011 \rightarrow 1111 \rightarrow 1011 \rightarrow 0011 \rightarrow 0010 \rightarrow 0000)$;

– π_5: $(0000 \rightarrow 1000 \rightarrow 1010 \rightarrow 1110 \rightarrow 1111 \rightarrow 1110 \rightarrow 1010 \rightarrow 1000 \rightarrow 0000)$;

– π_6: $(0000 \rightarrow 1000 \rightarrow 1100 \rightarrow 1101 \rightarrow 1111 \rightarrow 1101 \rightarrow 1100 \rightarrow 1000 \rightarrow 0000)$;

– π_7: $(0000 \rightarrow 0100 \rightarrow 0110 \rightarrow 0111 \rightarrow 1111 \rightarrow 0111 \rightarrow 0110 \rightarrow 0100 \rightarrow 0000)$;

– π_8: $(0000 \rightarrow 1000 \rightarrow 1010 \rightarrow 1011 \rightarrow 1111 \rightarrow 1011 \rightarrow 1010 \rightarrow 1000 \rightarrow 0000)$.

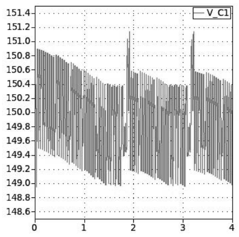
We present in Figures 5.4 and 5.5 a numerical simulation of this controller on the system starting from the point $v_1(0) = 150\,\mathrm{V}, v_2(0) = 100\,\mathrm{V}, v_3(0) = 50\,\mathrm{V}$ and $i(0) = -3\,\mathrm{A}$. This simulation has been performed using tool PLECS [PLE 13]. We can see on the simulation that the system state always stays inside S.

5.2.2. *Seven-level converter*

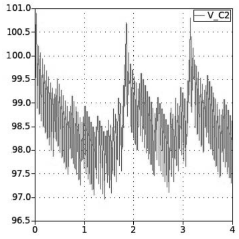
The flying capacitor architecture is generic. We now consider the case of an ℓ-level converter with $\ell = 7$. There are now six pairs of switching cells $S_1, \dots, S_6$ and five capacitors $C_1, \dots, C_5$. The state of the system is $x(t) = [v_1(t)\ v_2(t)\ v_3(t)\ v_4(t)\ v_5(t)\ i(t)]^T$ where $v_i(t)$ is the voltage across C_i $(1 \leq i \leq 5)$ and $i(t)$ is the current flowing in the circuit. The generated waveform now goes from $-v_{low}$ up to $+v_{high}$ with steps at $-v_{low} + iv_{high}/3$ for $i = 0, \dots, 6$, and the duration of a cycle is $T = 12\tau$. There are now $518,400$ possible sequences of control (patterns) for generating an ℓ-level staircase signal on one cycle. We used the following values for the system constants: output at $50\,\mathrm{Hz}$[2], capacitances of $0.1\,\mathrm{F}$, resistor values 50Ω, inductor values $0.137\,\mathrm{H}$, $v_{low} = v_{high} = 300\,\mathrm{V}$. Ideally, the output is thus a staircase waveform with an amplitude of 600 V, centered around 0 V, and the ideal values v_i^* of the capacitor voltages of the capacitor C_i are given by: $v_1^* = 500\,\mathrm{V}$, $v_2^* = 400\,\mathrm{V}$, $v_3^* = 300\,\mathrm{V}$, $v_4^* = 200\,\mathrm{V}$, $v_5^* = 100\,\mathrm{V}$. For R, we take $R = [495, 505] \times [395, 405] \times [295, 305] \times [195, 205] \times [95, 105]$, which corresponds to a product of intervals centered around the ideal values with a variation of $\pm 5V$ (i.e. 5% of the least charged capacitor C_5). For S, we take $R + \varepsilon$ with $\varepsilon = 1\,\mathrm{V}$, which means that we have an additional tolerance of $\pm 1\,\mathrm{V}$ for the fluctuations occurring

2 Corresponds to $T = 12\tau = 0.02\,\mathrm{s}$

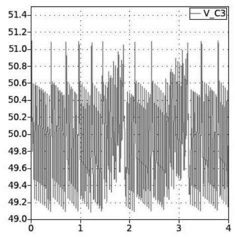
between two beginnings of cycle. The decomposition of R into subzones $\{V_i\}_{i \in I}$ and the corresponding set of patterns $\{\pi_i\}_{i \in I}$ are given in [FEL 12b]. We present in Figures 5.6 and 5.7 a numerical simulation of the controlled system starting from the point $v_1(0) = 500\,\text{V}$, $v_2(0) = 400\,\text{V}$, $v_3(0) = 300\,\text{V}$, $v_4(0) = 200\,\text{V}$, $v_5(0) = 100\,\text{V}$ and $i(0) = -2.5\,\text{A}$. We can check on the simulation that the system state always stays inside S.



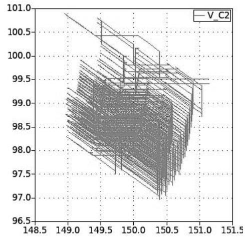
a) Voltage $v_1 = f(t)$



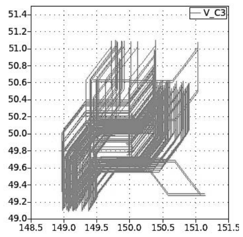
b) Voltage $v_2 = f(t)$



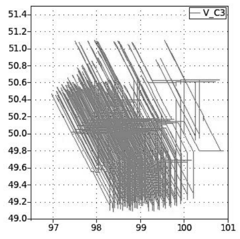
c) Voltage $v_3 = f(t)$



d) Plane $v_2 = f(v_1)$

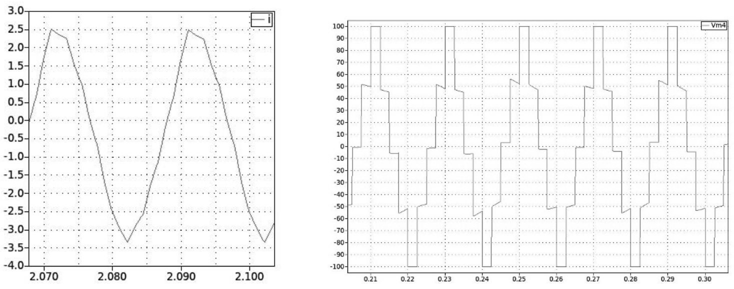


e) Plane $v_3 = f(v_1)$



f) Plane $v_3 = f(v_2)$

Figure 5.4. *Capacitor voltages*



a) Current i b) Output voltage v_o

Figure 5.5. *Current and output voltage*

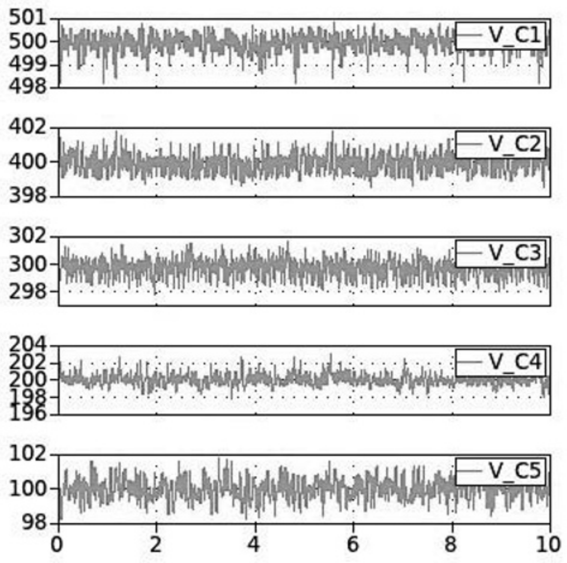
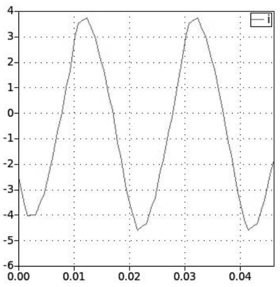


Figure 5.6. *Capacitor voltages*

We have also performed experiments with an ℓ-level converter for $\ell = 9$. We have been able to obtain a decomposition after 5 h of running time, but we are clearly at the limit of the existing implementation.


a) Current i

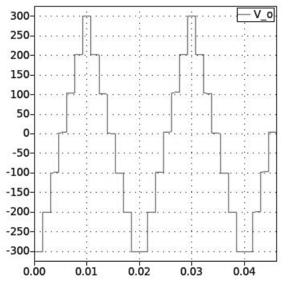

b) Current v_o

Figure 5.7. *Current and output voltage*

5.3. Physical experimentations

A prototype of the five-level flying capacitor has been built by the SATIE Laboratory in order to test our control strategy on an actual system; see Figure 5.8 for an image of the system. Our control strategy was applied to the system via Simulink and a dSpace® interface. The results are shown in Figure 5.9 for the output voltage and the capacitor charges. In Figure 5.10, we show the same results but with a larger scale on the capacitor voltage to see the fluctuations around the reference values. As we can see, the experimental results are very close to those obtained by simulation with PLECS [PLE 13]. In Figure 5.11, we show the output voltage together with the current (after appropriate resizing) flowing through the load. During the experimentations, we have successfully tested the robustness of the controller in presence of the following perturbations:

1) The ideal voltage source as input is no longer ideal but its values fluctuate around the reference value.

2) The system does not start from the reference valuations for the capacitor voltages and the input voltage, but the input voltage increases gradually until reaching the desired value while the capacitor is initially discharged.

3) We apply the same pattern during two consecutive cycles (instead of updating the pattern at the end of the first cycle).

4) We use a time-varying period T of cycle (instead of a constant period), and check the preservation of the capacitor voltages balance. The result of this experiment is shown in Figure 5.12.

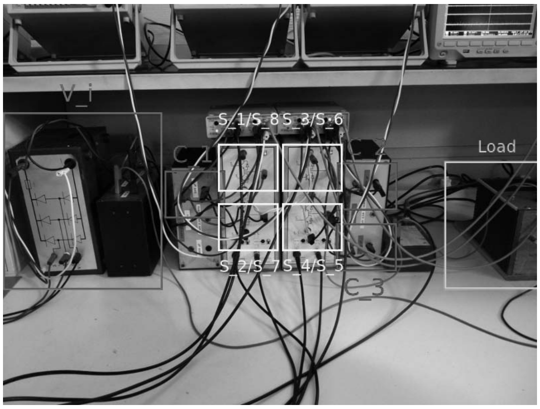


Figure 5.8. *Prototype built by SATIE (for a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip)*

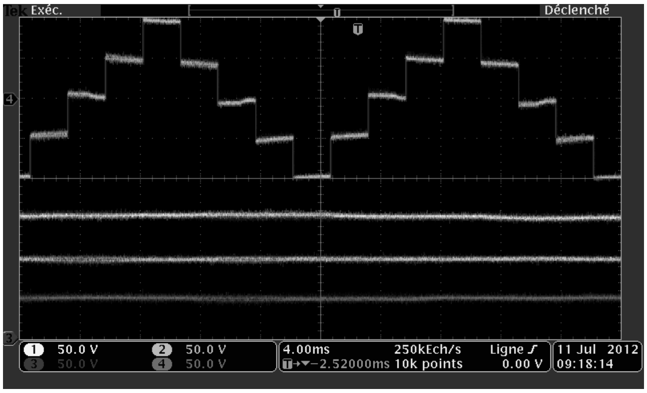


Figure 5.9. *Output voltage (above) and capacitor voltages (1 unit for 50 V) (below) (for a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip)*

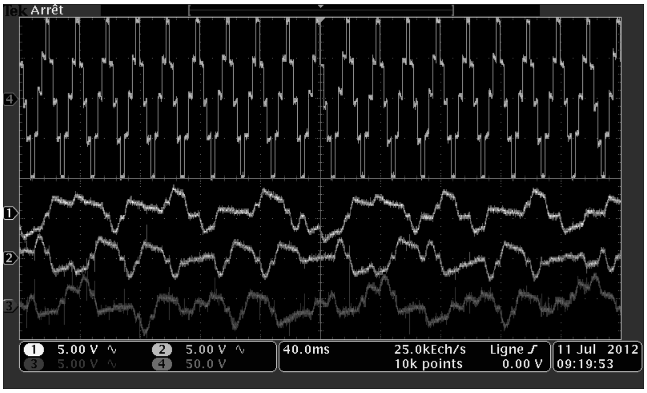


Figure 5.10. *Zoom of output voltage (above) and capacitors voltages (1 unit for 5 V) (below) (for a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip)*

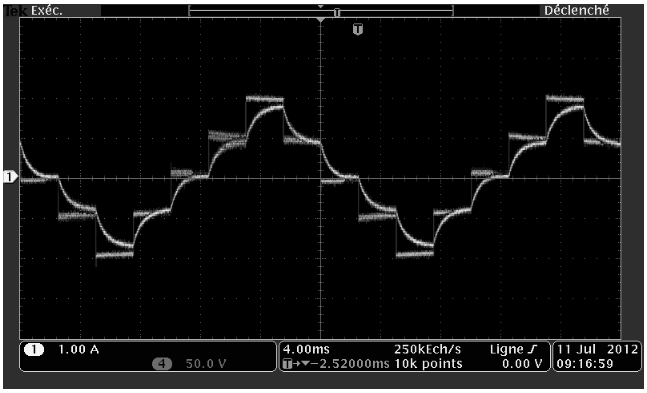


Figure 5.11. *Output voltage and current (after appropriate resizing) in the circuit (for a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip)*

Although these preliminary tests of robustness are promising, they need to be further validated, in particular in the presence of significant variations of resistor loads. Besides, although the principle of the method is general, it also suffers from an exponential increase of complexity when the level ℓ grows: the method reaches the limit for $\ell = 9$, which corresponds to a dimension $n = 7$ of the state space.

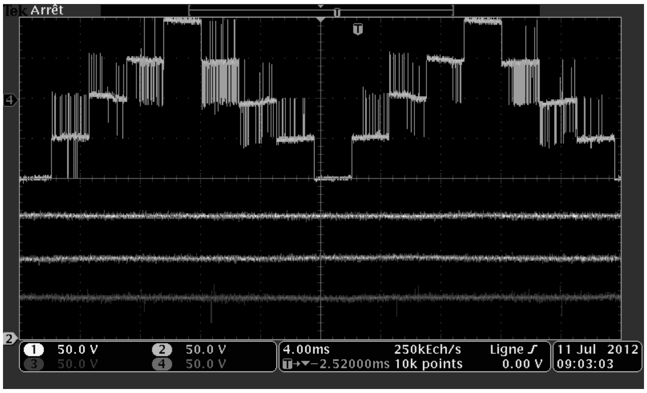


Figure 5.12. *Output voltage (above) and capacitor voltages (below) in the presence of time-varying period T (for a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip)*

5.4. Notes

We have synthesized a state-dependent control that involves only the subset of the state vector related to the voltage information, but ignores the intensity component. This is interesting because for practical applications, a current sensor is not always desired (see [DU 09]). More generally, control should use, as far as possible, only a subset of the state vector because measuring all signals in high-order converters becomes prohibitively expensive.

The method can be easily refined in order to generate sinusoidal-like output signals rather than the triangular-like output signals, as done here: it suffices to adjust the switching instants within the period T of the cycle, instead of using uniformly τ.

6

Other Issues: Reachability, Sensitivity, Robustness and Nonlinearity

In this chapter, we indicate several ways of using different facets of the procedure of invariant generation by decomposition explained in Chapter 4.

We show how an iteration of the decomposition procedure can be used to synthesize *reachability* controllers (section 6.1). We suggest to use the *sensitivity* of the limit cycles in order to infer the values of physical parameters of the system using an inverse approach (section 6.2). We explain how to extend the decomposition procedure in order to synthesize *robust safety* controllers in the presence of disturbance (section 6.3). We indicate how to extend the method for nonlinear systems in section 6.4.

6.1. Reachability control

The *reachability control* problem consists of steering the system from an initial region, say R, to a target region typically containing a reference point, say O. We have seen in Chapter 4 how the iteration of the decomposition procedure may drive the trajectories to a stability subregion of R. If such a region contains O, then the goal is achieved. Actually, if we start from a vast region of attraction R, it is possible to interleave the process of generating the successor sets with a process of updating the decompositions. We first apply the decomposition procedure to R: if the procedure is successful, we get a decomposition Δ with $Post_{\Delta}(R) \subset R$. If $Post_{\Delta}(R)$ still contains the reference point 0, we reapply the decomposition procedure to the bounding box $R' = \Box(Post_{\Delta}(R))$, of $Post_{\Delta}(R)$. If the procedure succeeds, yielding a new decomposition Δ', we can compute the successor set $Post_{\Delta'}(R') \subset R'$. Now, if $Post_{\Delta'}(R')$ still contains O, we can reiterate the process to $R'' = \Box(Post_{\Delta'}(R'))$, and so on iteratively. Thus, we produce nested boxes $R, R', R'', \ldots$ and associated decompositions $\Delta, \Delta', \Delta'', \ldots$. The procedure terminates either when the sequence stabilizes ($R^{i+1} \approx R^{i}$), or when the decomposition procedure fails, or when R^{i+1} does not contain O. The control induced by the decompositions $\Delta, \Delta', \ldots$ can be used to construct a *reachability controller* that steers the system from R to a region R^{i} containing O. This is illustrated in example 6.1.

EXAMPLE 6.1.– We first illustrate the process of iterative decomposition on the Boost converter example (see example 2.1). We start from a "large" region $R = [0, 10] \times [0, 4]$. We take $O = (1.8, 1.2)$ as a reference point. The iteration finds eight nested decompositions (see Figure 6.1). A trajectory controlled by such nested decompositions is given in Figure 6.2. The process is also illustrated in the helicopter motion example (see example 2.3), starting from $R = [-10, 10] \times [-10, 10]$, with the origin $(0, 0)$ as a reference point. The iteration finds nine nested decompositions (see Figure 6.3). A controlled trajectory is given in Figure 6.4.

Actually, since the procedure constructs the successive decompositions in a "blind" manner, without *a priori* consideration for the position of O, it is unable to drive the system to a *close* neighborhood of O. This shortcoming seems unavoidable when using a *forward* approach. More precise reachability controllers should implement a *backward* approach, starting from O and using dynamic programming techniques as done for example in [BER 00] and [LYG 99], but this may require an expensive gridding of the state space.

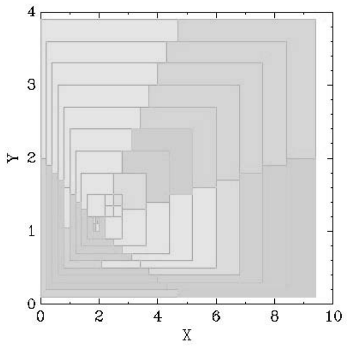


Figure 6.1. *Nested decompositions for Boost DC–DC converter found by starting from* $R = [0, 10] \times [0, 4]$

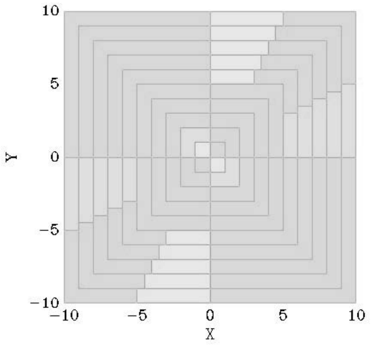


Figure 6.2. *Nested decompositions for helicopter motion found by starting from* $R = [-10, 10] \times [-10, 10]$

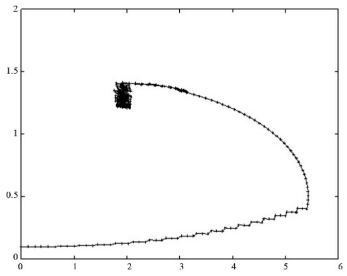


Figure 6.3. *Controlled trajectory of Boost DC–DC converter starting from* $(0, 0.01)$

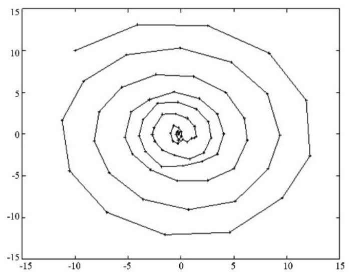


Figure 6.4. *Controlled trajectory of helicopter starting from* $(-10, 0)$

6.2. Sensitivity

We have shown in Chapter 4 that, under the control induced by decomposition of a given region R, trajectories converge to stable *limit cycles* inside R. In the following example, we point out here the sensitivity of limit cycles to parameter variations by showing the evolution of limit cycles in the presence of small perturbations of system parameters. As indicated, for example, in [HIS 01], this suggests that limit cycles are good candidates for reliable estimation of physical parameters of the system, using an appropriate *inverse approach* (see [TAR 05]).

EXAMPLE 6.2.– We take the Boost DC–DC example with the same region $R = [1.55, 2.15] \times [1.0, 1.4]$ as considered in example 4.1. The application of algorithm 4.1 with $k = 5$ and $d = 1$ yields a decomposition Δ of R (see Figure 4.1 of Chapter 4). As shown in Figure 6.5, the trajectories starting from the four corners of R, under the control strategy induced by Δ, converge to the same limit cycle. A remarkable feature is that, even in the presence of (small) *variations* of parameters of the system, the same decomposition Δ ensures the k-invariance of R. In our example, the decomposition Δ, originally found for $r_0 = 1$, is still k-invariant when r_0 varies from 0.965 to 1.005. It follows that the state-dependent control found for $r_0 = 1$ still ensures the convergence to limit cycles in R, for slight variations of r_0. Nevertheless, as shown in Figure 6.6, the form, length and position of the limit cycles are very sensitive to the actual value of r_0: for

$r_0 = 0.965$, the limit cycle corresponds to the pattern $(\pi_3\pi_1^3\pi_3\pi_1^2\pi_3\pi_1^3\pi_3\pi_1^3\pi_3\pi_1^3)$ (with $\pi_1 = (1 \cdot 1 \cdot 2 \cdot 2 \cdot 2)$, $\pi_3 = (2 \cdot 1 \cdot 2)$); for $r_0 = 0.975$, the pattern is $(\pi_3\pi_1^5)$, while for $r_0 = 1$ and $r_0 = 1.005$, it is just (π_1).

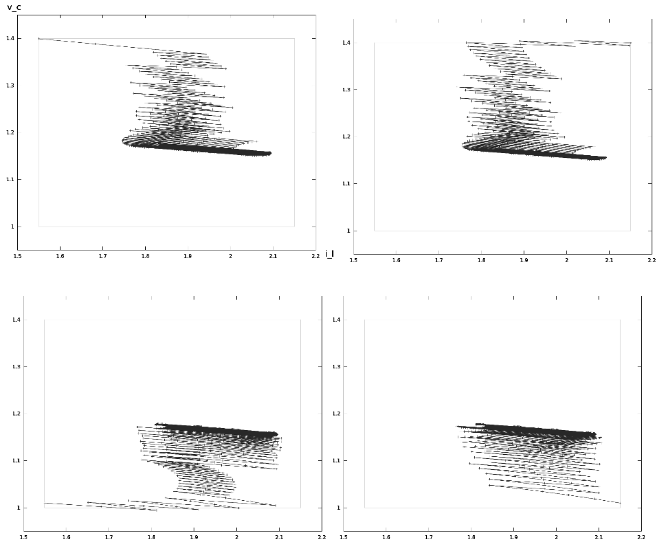


Figure 6.5. *Runs starting from the four corners of R, following the control strategy induced by the decomposition, and converging to the same limit cycle*

6.3. Robust safety control

As explained in section 2.3, zonotopes allow us to extend the computation of successor sets in order to account for small perturbations of the system dynamics. The dynamics of the system are now of the form $\dot{x} = A_u x + b_u + \varepsilon$ where ε represents a disturbance vector belonging to a given box Λ centered at the origin, and relations $Post_u$ and $Post_\pi$ are extended, as explained in section 2.3. The decomposition procedure and its enhancement for safety (section 4.3.2) are then simply adapted by replacing the inclusion test $Post_\pi(W) \subset R$ by $Post_\pi(W, \Lambda) \subset R$. We now give two examples of application of the decomposition procedure (enhanced for safety) in the presence of disturbance.

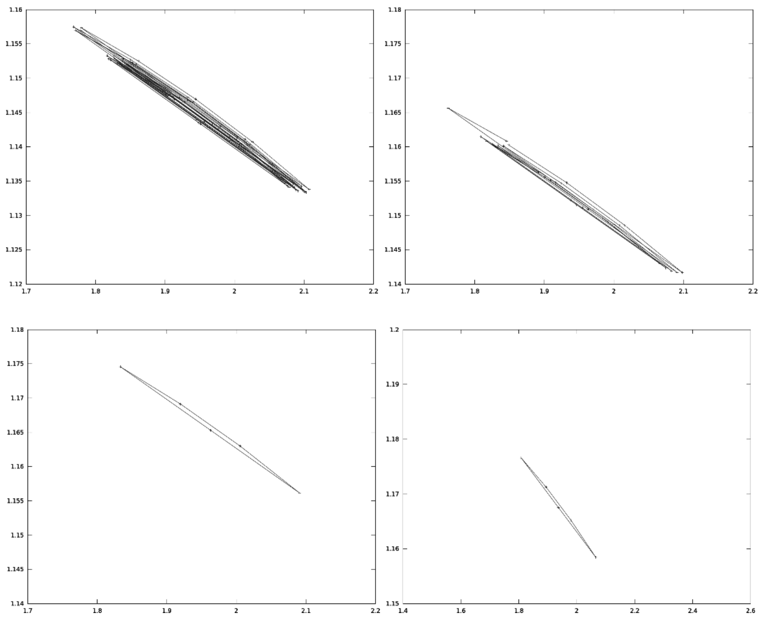


Figure 6.6. *Limit cycles for* $r_0 = 0.965$ *(pattern* $(\pi_3\pi_1^3\pi_3\pi_1^2\pi_3\pi_1^3\pi_3\pi_1^3\pi_3\pi_1^3)$*) on the upper left, for* $r_0 = 0.975$ *(pattern* $(\pi_3\pi_1^5)$*) on the upper right, for* $r_0 = 1$ *(pattern* π_1*) on the lower left, and for* $r_0 = 1.005$ *(pattern* π_1*) on the lower right*

EXAMPLE 6.3.– (BOOST CONVERTER).– We consider the dynamics of the Boost DC–DC converter (see examples 4.1 and 4.4) in the presence of disturbances ε belonging to $\Lambda = [0, 0] \times [\frac{-0.064}{x_l}, \frac{0.064}{x_l}]$. These disturbances correspond to noise on the input voltage, and represent up to 8% of the value of the input voltage. With such disturbances, we are *not* able to find a control preserving the safety zone of example 4.4 (viz. $[1.7, 2.0] \times [1.10, 1.30]$). We thus consider a larger (i.e. more tolerant) safety zone defined by $S' = [1.65, 2.05] \times [1.10, 1.30]$. The extended decomposition procedure then succeeds for $k = 13$ and $d = 5$: it generates a k-invariant decomposition Δ' of R satisfying $Unf_{\Delta'}(R) \subset S'$. The decomposition Δ' is shown in Figure 6.7[1]. A trajectory starting at $(1.75, 1.26)$, submitted to perturbation, is shown in Figure 6.8.

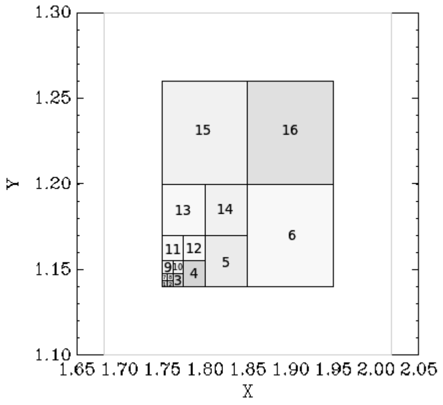


Figure 6.7. *k-invariant decomposition for Boost converter with disturbances*

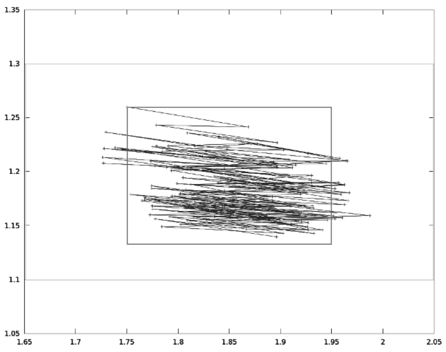


Figure 6.8. *Unfolded Δ-trajectory of the Boost converter with disturbances in plane (i_l, v_c), starting at $(1.75, 1.26)$ (inner box: $R = [1.75, 1.95] \times [1.14, 1.26]$, outer box: $S' = [1.65, 2.05] \times [1.1, 1.3]$)*

1 The corresponding patterns are: $\pi_1 = (1122121222)$, $\pi_2 = (1122122121222)$, $\pi_3 = (21121222)$, $\pi_4 = (12121222)$, $\pi_5 = (122)$, $\pi_6 = (2)$, $\pi_7 = (12)$, $\pi_8 = (12)$, $\pi_9 = (12)$, $\pi_{10} = (12)$, $\pi_{11} = (1)$, $\pi_{12} = (1)$, $\pi_{13} = (1)$, $\pi_{14} = (12)$, $\pi_{15} = (12)$ and $\pi_{16} = (21221)$.

EXAMPLE 6.4.– (HELICOPTER MOTION).– As done in [DIN 11], we will now solve the control problem with bounded disturbances to take into account a potential real-life environment. We consider the same regions $R = [-0.3, 0.3] \times [-0.5, 0.5]$ and $S = [-0.4, 0.4] \times [-0.7, 0.7]$ as in example A2.1, and add the disturbance $\varepsilon \in \Lambda = [-0.02, 0.02] \times [-0.1, 0.1]$. The extended decomposition procedure succeeds, and generates a k-invariant decomposition Δ' with $Unf_{\Delta'}(R) \subset S$. The decomposition Δ' is shown in Figure 6.9. A trajectory starting at point $(-0.3, 0.5)$, submitted to disturbance, is shown in Figure 6.10.

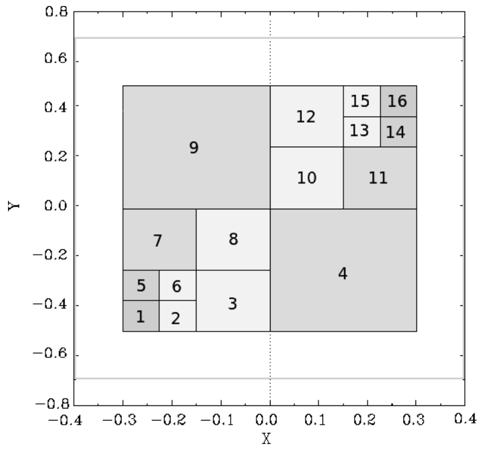


Figure 6.9. *k-invariant decomposition for helicopter motion with disturbance*

6.4. Nonlinearity

The basic decomposition procedure, explained in section 4.3.1, is quite general, and does not suppose that the successor operator $Post_\pi$ is linear or affine. However, in the case where $Post_\pi$ is an affine function, the computation can be done in an *exact* manner. We now discuss how to modify the decomposition procedure in case of non-affine dynamics. The process is inspired by what has been done for the handling disturbance (section 6.3). Following [ALT 08], we compute (an overapproximation of) the successor sets using *local* *linearizations* of the system, and enlargement of the linear images by addition of *error* intervals. We will consider here a system governed by a *unique* equation of the form $x(t + \tau) = f(x(t))$ where f is a polynomial. The set U is thus reduced to a single element ($U = \{1\}$). A pattern π associated with a subregion V is now just an *integer* indicating the number of times f should be applied when the state is in V. We put $f(x)$ under the form $f(x) = f_{lin}(x) + Q(x)$, where $f_{lin}(x)$ is a polynomial of order 1, and Q is a sum of monomials of order greater than or equal to 2. As in section 2.3, we can compute a local overapproximation of f by considering the polynomial subpart $Q(x)$ as a perturbation. We have:

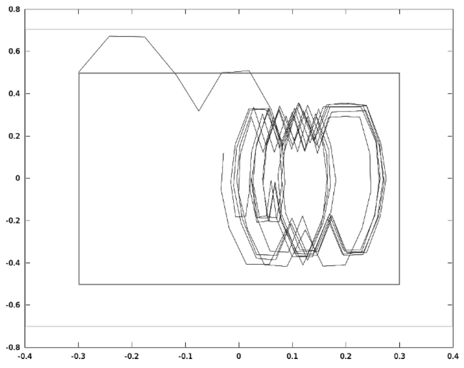


Figure 6.10. *Unfolded Δ-trajectory of helicopter motion with disturbance in plane $(x, \dot{x})$, starting at $(-0.3, 0.5)$ (inner box: R, outer box: S)*

LEMMA 6.1.– Consider a function f defined by: $f(x) = f_{lin}(x) + Q(x)$, where $f_{lin}(x)$ is the first-order polynomial of the form $d + Cx$, and $Q(x)$ is the sum of polynomials of order greater than or equal to 2. Given a zonotope $Z :< c, G >$, we have:

$$f(Z) \subset f_{lin}(Z) + Z_{\Lambda}$$

with:

– $f_{lin}(Z) =< f(c), CG >$

$$– Z_\Lambda =< 0, \begin{pmatrix} \varepsilon_1(Z) & 0 & \dots & 0 \\ 0 & \varepsilon_2(Z) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \varepsilon_n(Z) \end{pmatrix} >$$

with $(1 \leq i \leq n)$: $\varepsilon_i(Z) = max_{x \in Z}(|Q_i(x) - Q_i(c)|)$.

Using a repeated application of the lemma, we can compute an overapproximation of $f^\pi(Z)$ of the form $f^\pi_{lin}(Z) + Z^\pi_\Lambda$. This lemma allows to modify the decomposition procedure as follows: the inclusion test $Post_\pi(W) \subset R$, which corresponds here to a test of the form $f^\pi(W) \subset R$, is replaced by test $f^\pi_{lin}(W) + W^\pi_\Lambda \subset R$. What remains to do is to find an upper bound of $\varepsilon_i(W)$, that is an upper bound of $(|Q_i(x) - Q_i(c)|)$ over W, for each $1 \leq i \leq n$. We now explain two standard examples (taken from [AMI 12]) to compute such upper bounds. The decomposition procedure is then iterated in order to construct an attractor, which is an overapproximation of R^*_Δ.

EXAMPLE 6.5.– (VAN DER POL OSCILLATOR).–The dynamics of the Van der Pol oscillator are the following:

$$x(t+\tau) = \begin{pmatrix} 1 & \tau \\ -\tau & 1+\tau \end{pmatrix} x(t) + \begin{pmatrix} 0 \\ x_1(t)^2 x_2(t)\tau \end{pmatrix}.$$

When linearized to a point $c \in \mathbb{R}^2$, this gives:

$$x(\tau) = \begin{pmatrix} 1 & \tau \\ -\tau & 1+\tau \end{pmatrix} x(t) + \begin{pmatrix} 0 \\ -c_1^2 c_2 \tau \end{pmatrix}.$$

Thus, we have $f_{lin}(Z) = \begin{pmatrix} 1 & \tau \\ -\tau & 1+\tau \end{pmatrix} Z + \begin{pmatrix} 0 \\ -c_1^2 c_2 \tau \end{pmatrix} = \begin{pmatrix} 1 & \tau \\ -\tau & 1+\tau(1-c_1^2) \end{pmatrix} x(t)$. It is easy to see that for a box $V \subset \mathbb{R}^2$, we are making an error of at most 0 on the x-axis and $|(c_1^2 - (c_1 + G_{1,2} + G_{2,2})^2|\tau$ on the y-axis. Thus, we need to enlarge any image of a zonotope $Z = < c, G >$ by 0 on the x-axis and $\tau|(C_1^2 - (C_1 + G_{1,2} + G_{2,2})^2|$ on the y-axis, that is:

$$Z_\Lambda = < 0, \begin{pmatrix} 0 & 0 \\ 0 & |(C_1^2 - (C_1 + G_{1,2} + G_{2,2})^2|\tau \end{pmatrix} > .$$

The decomposition procedure is applied to $R = [-3, 3] \times [-3, 3]$ and $\tau = 0.01$ (with parameters $k = 30$, $d = 7$). At boxes located around the center of R, the length of patterns is 1 while in the lower left and upper right edges, the length is up to 30. The result of the decomposition is shown in the left part of Figure 6.11 and (an overapproximation of) the attractor set R_Δ^* in the right part.

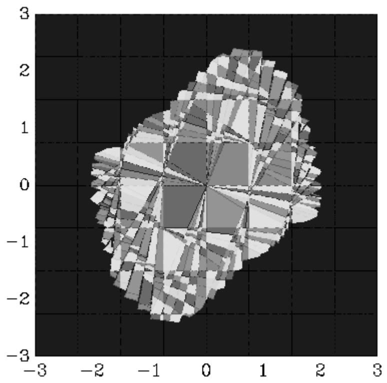


Figure 6.11. *Decomposition for the Van der Pol oscillator (left); R_Δ^j for $j = 30$ (right) (for a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip)*

EXAMPLE 6.6.– (FITZHUGH–NAGUMO NEURON).–The dynamics of the FitzHugh–Nagumo neuron are the following:

$$x(\tau) = \begin{pmatrix} 1+\tau & -\tau \\ 0.08\tau & -0.0064\tau + 1 \end{pmatrix} x(t) + \begin{pmatrix} -x_1(t)^3\tau/3 + 0.875\tau \\ 0.056\tau \end{pmatrix}$$

When linearized to a point $c \in \mathbb{R}^2$, this gives:

$$x(\tau) = \begin{pmatrix} 1+\tau & -\tau \\ 0.08\tau & -0.0064\tau + 1 \end{pmatrix} x(t) + \begin{pmatrix} -c_1^3\tau/3 + 0.875\tau \\ 0.056\tau \end{pmatrix}.$$

It is easy to see that for a box $V \subset \mathbb{R}^2$, we are making an error of at most $max_{x \in V}(\frac{|x_1^3 - c_1^3|}{3})\tau$ on the x-axis and 0 on the y-axis. Thus, we need to enlarge any image of a zonotope Z by $max_{x \in Z}(\frac{|x_1^3 - c_1^3|}{3})\tau$ on the x-axis and 0 on the y-axis, that is:

$$Z_\Lambda =< 0, \begin{pmatrix} max_{x \in Z}(\frac{|x_1^3 - c_1^3|}{3})\tau & 0 \\ 0 & 0 \end{pmatrix} > .$$

The decomposition procedure is applied to $R = [-2.5, 2.5] \times [-0.5, 2.5]$ and $\tau = 0.1$ (with parameters $k = 30$, $d = 7$). For boxes located around the center of R, the length of patterns is 1 while in the lower left and upper right corners, the length is up to 22. The result of the decomposition is shown in the left part of Figure 6.12 and (an overapproximation of) the attractor set R^*_Δ in the right part.

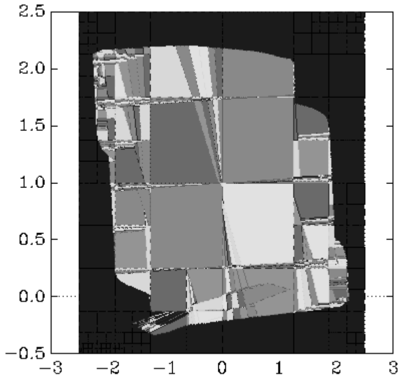


Figure 6.12. *Decomposition for the FitzHugh–Nagumo neuron (left); R^j_Δ for $j = 30$ (right) (for a color version of this figure, see www.iste.co.uk/fribourg/switchingsystems.zip)*

We have thus discussed how to construct attractors of polynomial dynamic systems by overapproximating the subpolynomial subpart of order greater than 1. So far, the overapproximation is done in an *ad hoc* fashion for each specific example. It would be interesting to use a general method of overapproximation based, for example, on the notion of Lagrange remainder (see [ALT 08]).

6.5. Notes

The reachability approach, discussed in section 6.1, does not take into account the question of time optimality. The *time-optimal control* problem consists of steering, in minimal time, the state of the system to a desired target while keeping the system safe along the way. It is classically solved using dynamic programming and a backward procedure (see [BER 00]). A solution for $\mathcal{S}^2$-systems, based on approximate bisimulation, is given in [GIR 10a].

The idea of using trajectory sensitivities in order to solve inverse problems, discussed in section 6.2, comes from [HIS 01], where it is explained how to use the measurements of disturbance effects to improve the estimates of parameters of power systems.

Conclusions and Perspectives

Switched systems are now commonly used in industrial domains such as power electronics or the automotive industry. These systems are easily programmable and allow for flexible design of the controlled components. However, with the growing number of switches, the task of control synthesis becomes challenging. Traditional methods based on extensive testing face difficulties to prove correctness of the control designed at hand. This opens the path to the synthesis of *correct-by-design* control software using formal methods.. We have focused in this book mainly on two issues of correctness for controllers: *safety* and *stability*. We have seen that the safety problem is closely related to the synthesis of a *maximal controlled invariant*, while the stability problem is related to the synthesis of a *minimal controlled invariant*.

The synthesis of a maximal controlled invariant is classically done by a fixed point iteration that is applied to a *backward* manner for computing successively the predecessor reachable sets. The procedure terminates when the state space is finite. Thus, the most common approach to provide correct-by-design synthesis techniques is *indirect*: it first converts the infinite state model into a finite state abstraction, then the controller synthesized at the abstract level is carried through the original infinite state model at the price of a certain approximation. However, in order to keep an acceptable precision, the method imposes a gridding of the state space that entails an explosion of the number of states at the abstract level. An other approach is to work *directly* at the infinite state level. As already mentioned, the backward iteration procedure is thus not guaranteed to terminate, and we have to *under-approximate* the symbolic reachable states generated in order to overcome this problem. However, here again, if we want to keep an acceptable precision, the number of states generated often becomes intractable, especially in the case of higher dimensional systems.

We have therefore proposed an alternative approach which is *forward oriented* and works directly on the infinite state level. In order to make the approach tractable, we have made a certain number of realistic assumptions: we have considered that the mode dynamics were affine, that the switching frequency was constant, and have ignored possible violations of the safety requirements between two switching instants. These assumptions are realistic in the context of power electronics. The method interleaves the decomposition of the operating space into subregions with the computation of patterns mapping these subregions inside the operating space. The method can be used for synthesizing safety controllers, as exemplified in the case studies of multilevel converters. Under further assumptions (essentially, local stability), this method is able to stabilize the system around identified limit cycles. We have finally sketched out some possible extensions of the decomposition procedure in order to address issues such as reachability control, robust control and nonlinear dynamics.

An important topic of current research is to synthesize controllers that satisfy multiple objectives *simultaneously*. When safety is among the objectives, a typical method is to construct the *maximally permissive* safety controller, then to refine it in order to complete secondary objectives. As usual, in practice, things are often more difficult, and techniques of synthesis and verification may have to be combined in an *ad hoc* manner. For example, in the oil pump system studied in [CAS 09], the problem is to design an operating cycle that meets safety requirements (for maintaining the oil level in a given interval), robustness (for taking into account imprecisions on measures of volume or time) and performance (for minimizing the oil accumulated during each cycle). Such a goal is achieved using a clever combination of analysis tools (UPPAAL-TIGA [BEH 07] for synthesis, PHAVER [FRE 08] for verification and SIMULINK [TEA 08] for simulation). In the domain of hybrid systems, many tools of control synthesis indeed share many common features with tools of verification: they rely on the same techniques of compact representation of states, and efficient construction of reachable sets. The tool d/dt [ASA 02] can thus be used to solve safety verification problems as well as safety switching controller synthesis. Other tools such as SPACEEX [FRE 11] and IMITATOR [AND 13] can be similarly used both for verification and synthesis problems of hybrid systems.

The methods and associated tools described in this book have now reached a level of maturity that allows them to tackle control problems with state space of dimensions up to 7 and reachable state sets made up of millions of elementary structures (typically, boxes or zonotopes, polyhedra). A major challenge is of course to design methods that scale with higher dimensional systems and to face the well-known *curse of dimensionality* (see [RUN 13] for example). This problem is currently addressed via the design of new structures for representing the union of convex sets, such as “support functions” [LEG 09] or “Brunowsky normal forms” [RUN 13].

In addition, as pointed out in the introduction, practical considerations specific to each case study often allow us to make realistic assumptions that considerably simplify the treatment of the problem. This was illustrated in the book by the multilevel converter case study where, due to physical considerations, the number of admissible sequences of modes during an electrical cycle was only a small fraction of the total number of possible combinations. The code given in Appendix 1 should allow the interested reader to reproduce himself/herself other examples of control synthesis given in the book and get initiated to the programming of switching systems. Hopefully, this will help the student to better grasp the interest and richness of formal methods and encourage the engineer to produce correct-by-design control software for interesting industrial applications.

Appendix 1

Sufficient Condition of Decomposition

Given a zone R, an invariant decomposition does not always exist. We give hereafter some geometrical conditions on the position of R that guarantee the decomposability of R when the system is contractive. For the sake of simplicity, we suppose that the switched system has only $|U| = 2$ modes, and the state-space dimension is $n = 2$, but the reasoning extends to larger values of $|U|$ and n. We assume that matrix A_u associated with mode u ($u = 1, 2$) is invertible and its eigenvalues have negative real parts. This implies that both modes are *contractive*. Let $e_u = -A_u^{-1}b_u$ be the unique attractive equilibrium point associated with mode u ($u = 1, 2$). Let us define the "pure" switching rule $\mathcal{S}_u$ ($u = 1, 2$) that applies repeatedly mode u to any point $x \in \mathbb{R}^2$. Let $\mathcal{C}_1$ (respectively, $\mathcal{C}_2$) be the τ-sampled trajectory issued from e_1 (respectively, e_2) under $\mathcal{S}_2$ (respectively, $\mathcal{S}_1$) (i.e. $\mathcal{C}_1 = Post_2^*(e_1)$ and $\mathcal{C}_2 = Post_1^*(e_2)$). Since each mode is contractive, and e_u is the unique equilibrium point associated with mode u, any trajectory under control $\mathcal{S}_u$ ends at the equilibrium point e_u ($u = 1, 2$), whatever the starting point of $\mathbb{R}^2$. In particular, $\mathcal{C}_1$ ends at e_2 and $\mathcal{C}_2$ at e_1, as shown in Figure A1.1 for the Boost converter (see examples 2.1 and 4.1).

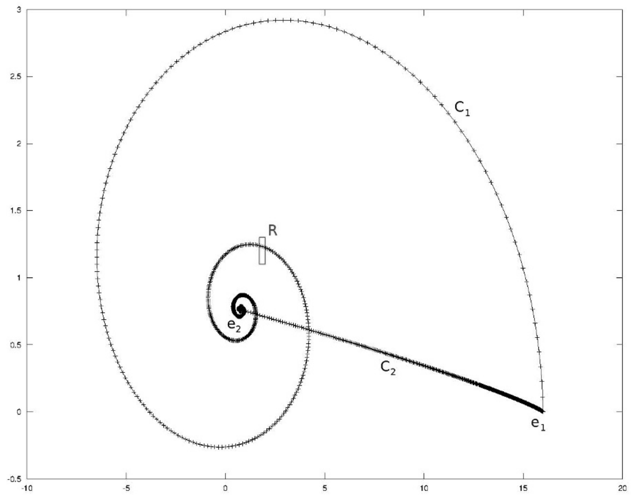


Figure A1.1. *Trajectories $\mathcal{C}_1$, and $\mathcal{C}_2$ and zone $R = [1.7, 2] \times [1.1, 1.2]$ for the DC-DC converter example*

THEOREM A1.1.– Let Σ be a sampled switched affine system as defined above. Suppose that the reference point O is in $\mathcal{C}_1 \cup \mathcal{C}_2$. If $R \subset \mathbb{R}^2$ is a box whose interior contains O, then there exists a positive integer k such that R is k-invariant.

PROOF.– Suppose $O \in \mathcal{C}_2$. (The case $O \in \mathcal{C}_1$ is symmetrical.) Consider a box R of interior $\dot{R}$ with $O \in \dot{R}$. There exists $\Delta_O > 0$ such that $\mathcal{B}(O, \Delta_O) \subset R$. Since $O \in \mathcal{C}_2$, we have:

a) $e_2 \rightarrow_{\pi_1} O$ for some pattern $\pi_1 \in (1)^*$.

Furthermore, for all $x \in R$, we have:

b) $x \rightarrow_{\pi_2} x_1$ for some $x_1 \in \mathcal{B}(e_2, \Delta_O)$ and some pattern $\pi_2 \in (2)^*$, because e_2 is an attractive equilibrium point;

c) $x_1 \rightarrow_{\pi_1} x_2$ for some $x_2 \in \mathcal{B}(O, \Delta_O)$, because of (a) and because mode 1 is contractive. This is shown in Figure A1.2. It follows from (b) and (c) that, for all $x \in R$: $x \rightarrow_{\pi_x} x_2$ for some $x_2 \in \mathcal{B}(O, \Delta_O)$ and some pattern $\pi_x \in (2^*1^*)$. Hence, we have $\mathcal{B}(x_2, \Delta_x) \subset R$ for some

$\Delta_x > 0$. Since, for any π_x, $Post_{\pi_x}$ is continuous, $Pre_{\pi_x}(\mathcal{B}(x_2, \Delta_x))$ is an open subset of $\mathbb{R}^2$ containing x. Since R is a compact of $\mathbb{R}^2$, from the set $C = \{Pre_{\pi_x}(\mathcal{B}(Post_{\pi_x}(x), \Delta_x))\}_{x \in R}$, we can extract a subset $C' = \{Pre_{\pi_{x_i}}(\mathcal{B}(Post_{\pi_{x_i}}(x_i), \Delta_{x_i}))\}_{i \in I}$ by Heine–Borel's theorem, for some finite set of indices I, such that C' covers R and $\mathcal{B}(x_i, \Delta_{x_i})$ is R-invariant via π_i. This means that $C' \cap R$ is a k-invariant decomposition of R of the form $\{(V_i, \pi_i)\}_{i=1,\ldots,m}$, where m is the cardinal of I, k the maximum length of $\pi_1, \ldots, \pi_m$ and $V_i = \mathcal{B}(x_i, \Delta_{x_i}) \cap R$ is such that $\bigcup_{i=1}^{m} V_i = R$ and V_i is R-invariant via π_i $(1 \leq i \leq m)$.

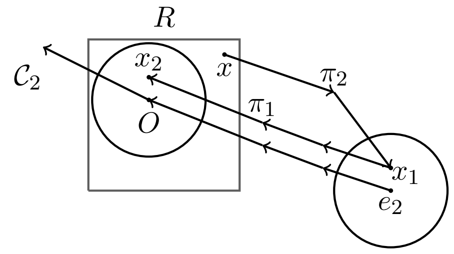


Figure A1.2. *Illustration of the proof*

The theorem gives an interesting locality condition on O (location on one of the "pure" trajectories linking the equilibrium points), for ensuring the existence of k-invariant boxes R. This justifies *a posteriori* the existence of a decomposition for the zone R of the DC–DC converter example 4.1 since it overlaps trajectory $\mathcal{C}_1$. Note also that R can be arbitrarily small as far as it intersects $\mathcal{C}_1$ (or $\mathcal{C}_2$).

Appendix 2

Applications of the Enhanced Decomposition Procedure

We now apply the decomposition procedure (enhanced for safety properties) for several examples. For each example, we are given a global control box R and a safety set S that contains it. We generate a k-invariant decomposition Δ of R satisfying $\mathit{Unf}_{\Delta}(R) \subset S$, thus proving that $\mathit{Unf}_{\Delta}(R)$ is a controlled invariant, and that the system is safe.

EXAMPLE A2.1.– Let us consider the helicopter motion example 2.3 of Chapter 2. We take $R = [-0.3, 0.3] \times [-0.5, 0.5]$ in plane (x_1, x_2). This corresponds to an equilibrium zone centered at the state $(0, 0)$ of the ground vehicle, and a variability of ± 0.3 for position and ± 0.5 for velocity. We take $S = [-0.4, 0.4] \times [-0.7, 0.7]$ for the safety region, which corresponds to an additional variability of ± 0.1 for position and velocity. (This is the same safety zone S as in [DIN 11].)

The application of algorithm 4.1 to R and S with $k = 6$ and $d = 4$ is successful, yielding a k-invariant decomposition Δ of R of the form $\{(V_i, \pi_i)\}_{i=1,\dots,10}$[1] satisfying $\mathit{Unf}_{\Delta}(R) \subset S$. The decomposition Δ is

1 The associated patterns are: $\pi_1 = (-10 \cdot -10 \cdot -10 \cdot -10 \cdot -10 \cdot 0 \cdot 10)$, $\pi_2 = (-10)$, $\pi_3 = (0)$, $\pi_4 = (-10 \cdot -10 \cdot -10 \cdot 10)$, $\pi_5 = (-10)$, $\pi_6 = (0)$, $\pi_7 = (-10)$, $\pi_8 = (10 \cdot 10 \cdot 0 \cdot 0)$, $\pi_9 = (0)$ and $\pi_{10} = (10 \cdot 10 \cdot 10 \cdot 10 \cdot 0 \cdot 10 \cdot -10)$.

shown in Figure A2.1. The unfolding of R is shown in Figure A2.2. The unfolding is divided into regions of three different colors corresponding to the different control modes: dark-gray color (respectively, medium gray and light gray) indicates that mode 10 (respectively. -10 and 0) should be applied. The surrounding box is $S = [-0.4, 0.4] \times [-0.6, 0.6]$.

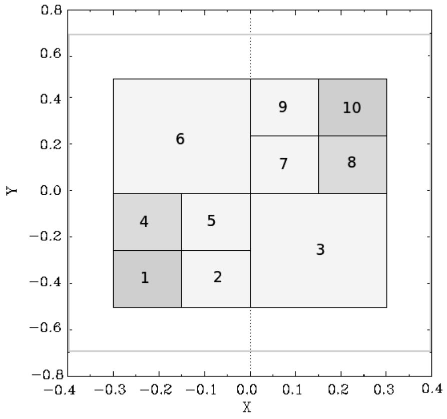


Figure A2.1. *k-invariant decomposition for helicopter motion*

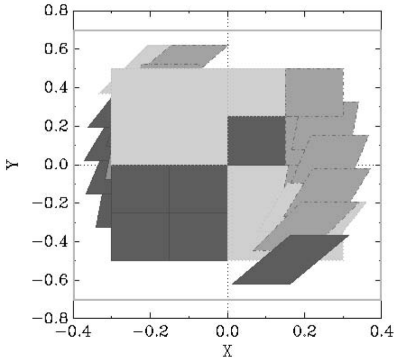


Figure A2.2. *Δ-unfolding for helicopter motion where dark gray (respectively, medium gray and light gray) indicates mode 10 (respectively, -10 and 0). (The enclosing box is S)*

The unfolded Δ-trajectory, in plane $(x, \dot{x})$, of the system starting at point $(-0.3, 0.5)$, is shown in Figure A2.3.

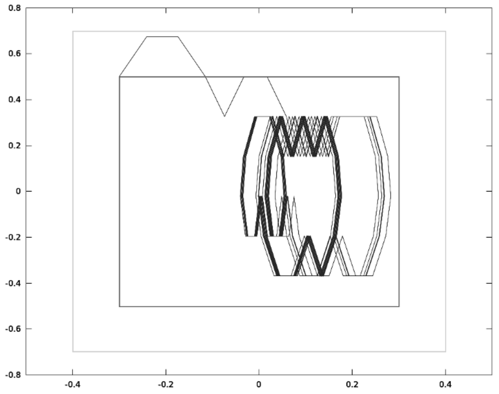


Figure A2.3. *Unfolded Δ-trajectory of helicopter motion in plane $(x, \dot{x})$ starting at $(-0.3, 0.5)$ (inner box: R, outer box: S)*

EXAMPLE A2.2.– (TWO-ROOM BUILDING HEATING).– Let us consider the example of the two-room building heating problem (see example 2.2 of Chapter 2). For the safety zone, we take $S = [20, 22] \times [20, 22]$, as in [GIR 12].

The application of algorithm 4.1 to R and S with $k = 4$ and $d = 2$ succeeds, yielding a k-invariant decomposition Δ of the form $\{(V_j, \pi_j)\}_{j=1,\ldots,10}$ of R^2 satisfying $Unf_\Delta(R) \subset S$. The decomposition Δ is shown in Figure A2.4. The unfolding of R is shown in Figure A2.5. The unfolding is divided into regions of two colors corresponding to the different control modes: the dark-gray color (respectively, light gray) indicates that control 0 (respectively, 1) should be applied. The outer box is the safety zone S.

2 The associated patterns are: $\pi_1 = (1010), \pi_2 = (1), \pi_3 = (10), \pi_4 = (0), \pi_5 = (0), \pi_6 = (1), \pi_7 = (0), \pi_8 = (0), \pi_9 = (0)$ and $\pi_{10} = (10)$.

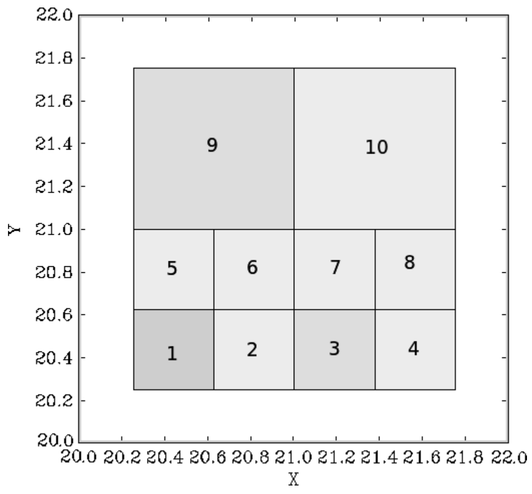


Figure A2.4. *k-invariant decomposition for heating system*

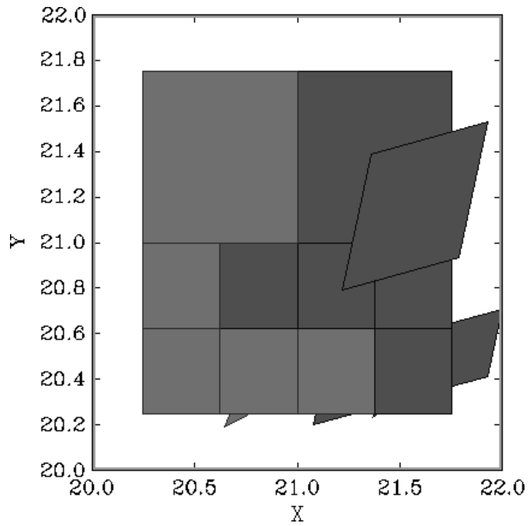


Figure A2.5. *Δ-unfolding for heating system where dark gray (respectively, light gray) indicates mode 0 (respectively, 1). (Outer box: $S = [20, 22] \times [20, 22]$)*

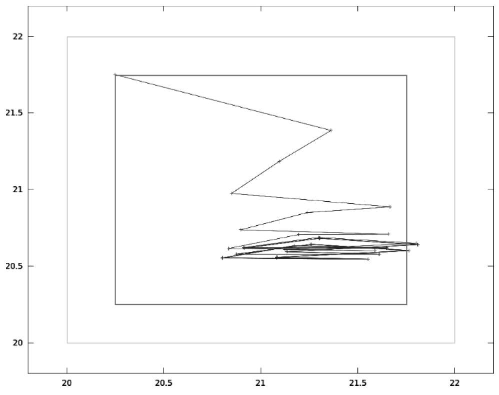


Figure A2.6. *Unfolded Δ-trajectory of heating system in plane (T_1, T_2), starting at $(20.25, 21.75)$ (inner box $R = [20.25, 21.75] \times [20.25, 21.75]$, outer box $S = [20, 22] \times [20, 22]$)*

The controlled system has been simulated using Octave [OCT 13]. A simulation is shown in Figure A2.6 for starting temperature point $(20.25, 21.75)$.

Appendix 3

Proof of Theorem 4.3

Consider a k-invariant decomposition $\Delta = \{(V_i, \pi_i)\}_{i \in I}$ of R, and the sequence $\{R^j_\Delta\}_{j \geq 0}$ and its limit R^*_Δ defined in Chapter 4.

Let us consider the (compact) convex sets W^k_σ where $k \in \mathbb{N}$ and $\sigma \in I^k$, defined as follows:

– $W^0_\epsilon = R$ where ϵ denotes the empty sequence.

– $W^{k+1}_{(i \cdot \sigma)} = Post_{\pi_i}(W^k_\sigma \cap V_i)$ with $i \in I$ and $\sigma \in I^k$.

It is easy to show that, for all $k \in \mathbb{N}$ and all $\sigma \in I^k$, W^k_σ is a compact convex set such that:

1) $Post^k_\Delta(R) = \bigcup_{\sigma \in I^k} W^k_\sigma$.

It follows from assumption (H2) that, for all sequence $\sigma \in I^N$, $W^N_\sigma \cap \partial\Delta = \emptyset$, therefore $W^N_\sigma \subset \overset{\circ}{V_i}$ for some $i \in I$, and $Post_\Delta(W^N_\sigma) = Post_{\pi_i}(W^N_\sigma)$. Since $Post^{N+1}_\Delta(R) \subset Post^N_\Delta(R)$, $Post_\Delta(W^N_\sigma) = Post_{\pi_i}(W^N_\sigma)$ is a compact convex set included into $Post^N_\Delta(R)$, and is therefore included in one of the convex components of $Post^N_\Delta(R)$. We have:

2) $Post_\Delta(W^N_\sigma) \subset W^N_{\sigma'}$ for some $\sigma' \in I^N$.

Now, for all Δ-trajectory $\{x_0, x_1, \dots\}$, we have:

3) $\forall k \geq N \; \exists \sigma \in I^N : \; x_k \in W_\sigma^N$ because, for all $k \geq N$, $x_k \in Post_\Delta^k(R) \subset Post_\Delta^N(R) = \bigcup_{\sigma \in I^N} W_\sigma^N$. By rewriting the elements of $\{W_\sigma^N\}_{\sigma \in I^N}$ under the form $\{W_1, \dots, W_M\}$, and denoting the set $\{1, \dots, M\}$ by J, we recapitulate these results as follows.

PROPOSITION A3.1.– There exist M compact convex sets $W_1, \dots, W_M$ with $\bigcup_{j=1}^M W_j \cap \partial\Delta = \emptyset$, such that:

1′) $Post_\Delta^N(R) = \bigcup_{j=1}^M W_j$.

2′) $\forall j \in J \; \exists j' \in J : \; Post_\Delta(W_j) \subset W_{j'}$.

3′) for all Δ-trajectory $\{x_0, x_1, \dots\}$, $\forall i \geq N \; \exists j \in J : \; x_i \in W_j$.

The element j' associated with j in $(2')$ will be denoted by $s(j)$. (If there is more than one such j', an arbitrary one is selected.)

Using part $(2')$ of proposition A3.1, we can define a directed graph where J is the set of vertices, and there is an oriented edge from $j \in J$ to $j' \in J$ iff $j' = s(j)$ (i.e. $Post_\Delta(W_j) \subset W_{j'}$). The strongly connected components of this graph are cyclic (because each vertex of the graph has a unique outgoing edge).

In the following, we will denote a cyclic subgraph by $\mathcal{C} = (j_0, \dots, j_{m-1})$ with $j_{\ell+1} = s(j_\ell)$ for $0 \leq \ell \leq m-1$, using the convention: $j_m = j_0$. For every element j of $\mathcal{C}$, we have $s^m(j) = j$, that is $Post_\Delta^m(W_j) \subset W_j$. More generally, $Post_\Delta^{(i+1)\cdot m}(W_j) \subset Post_\Delta^{i\cdot m}(W_j)$. We can define a decreasing sequence of non-empty compact convex sets $Post_\Delta^{i\cdot m} W_j$ for all $i \geq 0$. The limit set $\bigcap_{i \geq 0} Post_\Delta^{i\cdot m}(W_j)$ is a non-empty compact set. Furthermore, since by (H1) all the modes are contractive, it is easy to see that this limit set is reduced to a point that will be denoted by z_j. Every cycle of indices $\mathcal{C} = (j_0, \dots, j_{m-1})$ thus corresponds to a cycle of points $Z_\mathcal{C} = (z_{j_0}, \dots, z_{j_{m-1}})$. Furthermore, it is easy to show that, for all $0 \leq \ell \leq m-1$, we have $succ_\Delta(z_{j_\ell}) = z_{j_{\ell+1}}$. Let us denote the set of vertices of all the cyclic subgraphs by J'. An illustration of the form of such a graph is given in Figure A3.1.

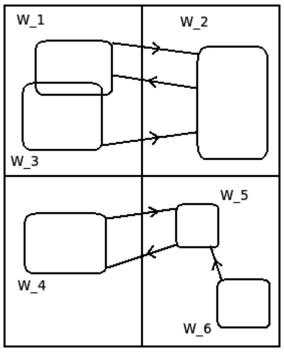


Figure A3.1. *Illustration of the graph of W_js with $J = \{1,2,3,4,5,6\}$ and $J' = \{1,2,4,5\}$*

These results are recapitulated in the following.

PROPOSITION A3.2.– For all cycle $\mathcal{C} = (j_0, \dots, j_{m-1})$ of J', we have: $\forall j \in \mathcal{C} : \;\; Post^m_\Delta(W_j) \subset W_j$.

Furthermore, for all $j \in \mathcal{C}$, the set $\bigcap_{i \geq 0} Post^{i \cdot m}_\Delta(W_j)$ is well defined and equal to a point denoted by z_j. We have, for all $\ell = 0, \dots, m-1$: $succ_\Delta(z_{j_\ell}) = z_{j_{\ell+1}}$.

Consider now the vertices of the graph that are $J \setminus J'$. Each of them is the destination of a finite number of acyclic paths. This means that after a finite number of iterations of $Post_\Delta$, no point will belong to W_j for $j \in J \setminus J'$[1]. Formally: $\exists M \geq N \; \forall j \in J \setminus J' \; Post^M_\Delta(R) \cap W_j = \emptyset$. Furthermore, $\forall k \geq M \; Post^k_\Delta(R) \subset \bigcup_{j \in J'} W_j$. It follows that, for all Δ-trajectory $\{x_0, x_1, \dots\}$, x_M belongs to W_j for some $j \in J'$. Let $\mathcal{C} = (j_0, \dots, j_{m-1})$ be the cycle that contains j. *Modulo* a circular permutation of $\mathcal{C}$, we can suppose $j = j_0$. Then, for all $0 \leq \ell \leq m-1$,

1 More formally, no point will belong to a W_j for $j \in J \setminus J'$ if it does not belong to a $W_{j'}$ for $j' \in J'$

$x_{M+\ell}$ belongs to W_{j_ℓ}. More generally, $x_{M+i\cdot m+\ell} \in Post^{i\cdot m}(W_{j_\ell})$. It follows: $\lim_{i\to\infty} x_{M+i\cdot m+\ell} = \bigcap_{i\geq 0} Post_\Delta^{i\cdot m}(W_{j_\ell}) = z_{j_\ell}$. We have:

THEOREM A3.1.– Every Δ-trajectory $\{x_0, x_1, \dots\}$ converges to a limit cycle in the following sense: for all initial points $x_0 \in R$, there exists a cycle $\mathcal{C} = (j_0, \dots, j_{m-1})$ of J', a cycle of points $Z_\mathcal{C} = (z_{j_0}, \dots, z_{j_{m-1}})$ and an integer $M \in \mathbb{N}$ such that, for all $\ell = 0, \dots, m-1$:

– $\forall i \geq 0 : \; x_{M+i\cdot m+\ell} \in Post_\Delta^{i\cdot m}(W_{j_\ell}) \subset W_{j_\ell}$.

– $\lim_{i\to\infty} x_{M+i\cdot m+\ell} = z_{j_\ell}$.

The expression $Post_\Delta^m(z_{j_\ell}) = z_{j_\ell}$ of theorem 4.3 gives a practical method to compute z_{j_ℓ} ($\ell = 0, \dots, m-1$). Indeed, we have that $z_{j_\ell} \in V_i$ for some $i \in I$. Let us denote such an i by $\phi(j_\ell)$, and let $\pi = (\pi_{\phi(j_0)} \cdots \pi_{\phi(j_{m-1})})$. Since $Post_\Delta^m(z_{j_0}) = Post_\pi(z_{j_0}) = C_\pi \cdot z_{j_0} + d_\pi$ for some matrix C_π and vector d_π, we can compute z_{j_0} as a solution of the equation $z_{j_0} = C_\pi \cdot z_{j_0} + d_\pi$. Similar equations hold for z_{j_ℓ} with $\ell = 1, \dots, m-1$. Furthermore, we have:

PROPOSITION A3.3.– $R_\Delta^* = \{z_j \mid j \in J'\}$.

PROOF.– We know that there exists $K > 0$ such that, for all $k \geq K$, $Post_\Delta^k(R) = \bigcup_{j\in J'} Post_\Delta^k(W_j)$. We also know $\bigcap_{k\geq 0} Post_\Delta^k(W_j) = \{z_j\}$. It follows $R_\Delta^* = \bigcap_{k\geq 0} Post_\Delta^k(R) = \bigcup_{j\in J'} \bigcap_{k\geq 0} Post_\Delta^k(W_j) = \{z_j \mid j \in J'\}$.

The elements of $\{z_j \mid j \in J'\}$ are grouped together into cyclic sets of points. For all cyclic set of points $Z_\mathcal{C}$, we have $Post_\Delta(Z_\mathcal{C}) = Z_\mathcal{C}$. It follows:

PROPOSITION A3.4.– For all cyclic set of points $Z_\mathcal{C}$, the Δ-unfolding of $Z_\mathcal{C}$ is a controlled invariant set.

From theorem A3.1 and propositions A3.3 and A3.4 follows theorem 4.3 of Chapter 4.

Appendix 4

Example with $|R^*_\Delta| = \infty$

For this example, we use modes associated with repulsive homothetic transformation. There are four modes such that, close to each corner of the global box $R = [-1, 1] \times [-1, 1]$, there exists a fixed point for one of the mode. We take for the dynamics of the modes: $A_1 = A_2 = A_3 = A_4 = \begin{pmatrix} 1.5 & 0 \\ 0 & 1.5 \end{pmatrix}$, $b_1 = \begin{pmatrix} 0.6 \\ 0.6 \end{pmatrix}$, $b_2 = \begin{pmatrix} -0.6 \\ 0.6 \end{pmatrix}$, $b_3 = \begin{pmatrix} -0.6 \\ -0.6 \end{pmatrix}$ and $b_4 = \begin{pmatrix} 0.6 \\ -0.6 \end{pmatrix}$. The Δ-decomposition is shown in Figure A4.1. We have $V_1 = [-1, 0] \times [-1, 0]$ associated with pattern $\pi_1 = (1)$, $V_2 = [0, 1] \times [-1, 0]$ associated with pattern $\pi_2 = (2)$, $V_3 = [0, 1] \times [0, 1]$ associated with pattern $\pi_3 = (3)$ and $V_4 = [-1, 0] \times [0, 1]$ associated with pattern $\pi_4 = (4)$. A Δ-trajectory is shown in Figure A4.2. We can see a chaotic behavior and an absence of convergence toward a limit cycle.

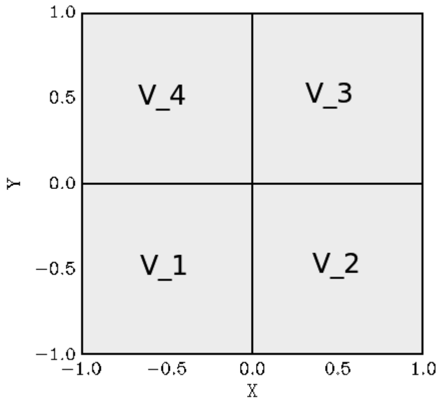


Figure A4.1. *Decomposition* Δ *of* R *for the non-contractive example*

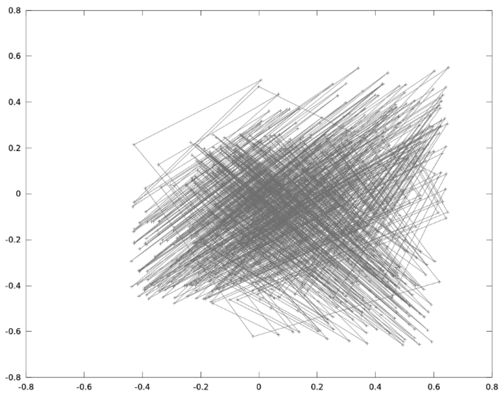


Figure A4.2. Δ*-trajectory for the non-contractive example*

Appendix 5

Code

All the codes are written in Octave [OCT 13].

We first recall some basic facts about the construction of the bounding box of a zonotope and the test of inclusion of a zonotope into a box (see section 2.3).

Let $Z =< c, G >$ be a zonotope where G is an $n \times p$ matrix. A box is a zonotope $Z =< c, D >$ where D is an $n \times n$ diagonal matrix. The bounding box of a zonotope $< c, G >$ is a zonotope, denoted by $\square(Z)$, of the form $< c, G' >$ where G' is an $n \times n$ diagonal matrix of the form:

$$\begin{pmatrix} \sum_{i=1}^{p} |G_{1,i}| & 0 & ... & 0 \\ 0 & \sum_{i=1}^{p} |G_{2,i}| & ... & 0 \\ . & . & . & . \\ 0 & 0 & ... & \sum_{i=1}^{p} |G_{n,i}| \end{pmatrix}.$$

The inclusion test of a zonotope Z into a box Z' is equivalent to the inclusion test $\square(Z) \subset Z'$. Given an $n \times n$ matrix G, and an n-vector g, let: $diag(G) = [G_{1,1}\ G_{2,2}\ \cdots\ G_{n,n}]^T$ and $abs(g) = [g_1\ g_2\ \cdots\ g_n]^T$.

PROPOSITION A5.1.– Let $Z =< c, D >$ and $Z' =< c', D' >$ be two boxes.

$$Z \subset Z' \iff \forall i \in \{1, \ldots, n\}\ abs(c - c')_i$$
$$+abs(diag(D))_i <= abs(diag(D'))_i.$$

A5.1. Functions on zonotopes

The function in Figure A5.1 decomposes a zonotope into its center C and its generators G (under the form of a matrix).

```
function C,G =zonotope_decompo(zonotope)
        % C is the center
        C = zonotope(:,1);
        % G is the generator matrix
        G = zonotope(:,2:length(zonotope(1,:)));
endfunction
```

Figure A5.1. *Function that decomposes a zonotope into its center and its generators*

The function in Figure A5.2 computes the bounding box of a given zonotope. The smallest bounding box $\square(Z)$ of a zonotope $Z =< c, G >$ is:

$$< c, \begin{pmatrix} \sum_{i=1}^{m} |G_{1,i}| & 0 & 0 & \dots & 0 \\ 0 & \sum_{i=1}^{m} |G_{2,i}| & 0 & \dots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & \dots & 0 & \sum_{i=1}^{m} |G_{n-1,i}| & 0 \\ 0 & 0 & \dots & 0 & \sum_{i=1}^{m} |G_{n,i}| \end{pmatrix} > .$$

```
function squared_zonotope = square_zonotope(zonotope)
        [C,G] = zonotope_decompo(zonotope);
        n=length(G(:,1));
        m = length(G(1,:));
        G_squared = zeros(n,n);
        for i=1:n
                for j=1:m
                        G_squared(i,i)=G_squared(i,i) + abs(G(i,j));
                endfor
        endfor
        squared_zonotope=[C, G_squared];
endfunction
```

Figure A5.2. *Function that computes the bounding box of a zonotope*

The function in Figure A5.3 tests the inclusion of a zonotope into a box. It is easy to see that testing the inclusion of a zonotope Z into a box

B is equivalent to testing the inclusion of $\square(Z)$ into B. This operation can then be done easily by computing the minimum and maximum of $\square(Z)$ on each dimension and comparing it with B.

```
function inclusion = inclusion_zonotope(zonotope_1 ,zonotope_2)
        %Zonotope 2 is a box
        %We test the inclusion of zonotope_1 into zontope_2
        %The test of inclusion of zonotope_1 into zonotope_2
        %is equivalent to the test of inclusion of
        %square_zonotope(zonotope_1) into zonotope_2
        zonotope_1 = square_zonotope(zonotope_1);
        [C1,G]=zonotope_decompo(zonotope_1);
        [C2,D]=zonotope_decompo(zonotope_2);

        vec_G = diagonale_matrix(G);
        vec_D = diagonale_matrix(D);

        if (abs(C1-C2)+abs(vec_G) <= abs(vec_D))
                inclusion = 1;
        else inclusion = 0;
        endif
endfunction
```

Figure A5.3. *Function that tests the inclusion of a zonotope into a box*

The function $diagonale\_matrix(M)$ returns the diagonal of a matrix M.

The function in Figure A5.4 returns a subset of a zonotope. More precisely, given a zonotope Z, it will return one of the quadrant of Z.

```
function morceau = split_zonotope(zonotope ,index)
        [C,G]=zonotope_decompo(zonotope);
        morceau = [C + 1/2*G*int2bin(index ,length(C)),1/2*G];
endfunction
```

Figure A5.4. *Function that returns a subset of a zonotope*

In Figure A5.4, $int2bin(n,p)$ is a function that converts a number n into its binary form with p digits. For example, $int2bin(2,5)$ returns 00010.

The function in Figure A5.5 returns the image of a zonotope by an affine application of the form $f : X \mapsto \alpha_c X + \beta_c$.

```
function zonotope_image = affine_transfo(zonotope,alpha_c,beta_c)
        [C,G]=zonotope_decompo(zonotope);
        zonotope_image = [alpha_c*C + beta_c,alpha_c*G];
endfunction
```

Figure A5.5. *Function that computes the image of a zonotope by an affine application*

The function in Figure A5.6 returns the image of a zonotope by a pattern corresponding to a sequence of affine transformations.

```
function zonotope_image = pattern_image(zonotope,pattern)
        [C,G]=zonotope_decompo(zonotope);
        zonotope_image = zonotope;
        for i=1:length(pattern)
                zonotope_image = affine_transfo((zonotope_image),
                          alpha_c(pattern(i)), beta_c(pattern(i)));
        endfor
endfunction
```

Figure A5.6. *Function that computes the image of a zonotope by a pattern*

The function in Figure A5.7 tests whether or not a point belongs in a zonotope. This is simply done by considering a point as a zonotope with generator $G = 0$.

```
function answer = is_in_zonotope(point,zonotope)
        zonotope_point = [point,zeros(length(point),length(point))];
        answer = inclusion_zonotope(zonotope_point,zonotope);
endfunction
```

Figure A5.7. *Function that tests whether or not a point belongs in a zonotope*

A5.2. Function on patterns

The function in Figure A5.8 returns a pattern. Given a pattern, it will output the next pattern in the lexicographical order of the same length. If none exists, it will return the first pattern with a greater length.

A5.3. Functions on control

Given a zonotope that needs to be controlled, a zonotope target and a pattern, the function in Figure A5.9 returns True if the image of $zonotope\_control$ by pattern is included into $zonotope\_target$, otherwise false.

```
function next_pattern = next_pattern(pattern)
        global number_modes;
        next_pattern = pattern;
        n=length(pattern);
        i = n;
        while (i > 0 && next_pattern(i) == number_modes)
                next_pattern(i) = 1;
                i = i-1;
        endwhile
        if (i == 0)
        next_pattern = [1,next_pattern];
        else next_pattern(i)=next_pattern(i) + 1;
        endif
        return
endfunction
```

Figure A5.8. *Function that computes the next pattern to test*

```
function flag = is_zonotope_invariant(zon_control ,zon_target ,pattern)
        [C,G] = zonotope_decompo(zon_target);
        flag = inclusion_zonotope(pattern_image(zon_control ,pattern),
                                          zon_target);
endfunction
```

Figure A5.9. *Function that tests whether the image of zonotope_control by pattern is included into zonotope_target*

The function in Figure A5.10 returns True if there exists a pattern up to length $pattern\_max\_length$ that makes the image of $zonotope\_control$ in $zonotope\_target$.

The function in Figure A5.12 computes the decomposition of $zonotope\_control$ in order to make it invariant into $zonotope\_target$. In practice, we first call it $zonotope\_control = zonotope\_target$.

```
function is_contr = Find_Pattern(zon_control ,zon_target ,tile_num)
        global pattern_max_length;
        pattern = [1];
        admissible_pattern_found = 0;
        is_contr = 0;
        while ((length(pattern) < pattern_max_length)
                    && (admissible_pattern_found == 0))
                if (is_zonotope_invariant(zon_control ,zon_target ,pattern)
                     == 1)
                        admissible_pattern_found = 1;
                        is_contr = length(pattern);
                endif
                pattern = next_pattern(pattern);

        endwhile
endfunction
```

Figure A5.10. *Function $Find\_Pattern$*

```
function Decomposition(zon_control , zon_target , d)
        global max_depth;
        global pattern_max_length;
        is_contr = Find_Pattern(zon_control ,zon_target);
        dimensions = length(zon_control(: ,1));
        if (is_contr == 0)
                if (d < max_depth)
                        for i=0:2^dimensions-1
                                sub_zon = split_zonotope(zon_control ,i);
                                Decomposition(sub_zon ,zon_target ,d+1);
                        endfor
                endif
        else Graphic Output or controller
        endif
endfunction
```

Figure A5.11. *Decomposition procedure*

A5.4. Other Useful Functions

The function $int2bin(n, p)$ in Figure A5.4 is a function that converts a number n into its binary form with p digits. For example, $int2bin(2, 5)$ returns 00010.

```
function vec_binaire = int2bin(k,m)
    vec_binaire = zeros(m,1);
    for i=1:m
        vec_binaire(i,1)=0;
    endfor
    i=m;
    while(k~=0)
        if (mod(k,2)==1)
            vec_binaire(i,1)=1;
            k=(k-1)/2;
        else k=k/2;
        endif
        i--;
    endwhile

endfunction
```

Figure A5.12. *Decomposition procedure*

A5.5. Building and running an example

In this section, we discuss how to build and execute an example. We use the Boost DC–DC converter to illustrate the process.

First, we need to build a .m file that describes the dynamics of the system. A skeleton is shown in Figure A5.13 and the Boost converter file is shown in Figure A5.14. We recall that we only consider dynamics under the form $\dot{x} = A_u x + b_u$, where A_u is an $n \times n$ matrix and b_u is an n array for all modes u. Moreover, we only consider τ - sampled system. Therefore, we can define a successor function $Post_u(x_0) = C_u x + d_u$ by solving the differential equation of the dynamics for a time elapse of τ time units. This .m file simply describes how to compute C_u and d_u for all modes u.

Once this .m file is built, we need to build a shell script file that will contain the constants for the analysis used by Octave (i.e. R, $MaxDepth$ and $MaxLength$).

```
##############################
# CONSTANTS OF THE ANALYSIS #
##############################

global max_depth;
global file_to_carto;
global pattern_max_length;
global total_pattern_tried = 0;

########################
#     EXAMPLE CONSTANTS    #
########################
global example_name = "example_name";
global numberModes = n;

global A_1 = ;
...
global A_n = ;

global B_1 = ;
...
global B_n = ;

function C = alpha_c(mode)
        global A_1;
        ...
        global A_n;

        global B_1;
        ...
        global B_n;

        global tau;

        if (mode == 1)
                C = ;
        ...
        elseif  (mode == n)
                C = ;
        endif
endfunction

function D = beta_c(mode)
        global A_1;
        ...
        global A_n;

        global B_1;
        ...
        global B_n;

        global tau;

        if (mode == 1)
                D = ;
        ...
        elseif  (mode == n)
                D = ;
        endif
endfunction

# Start control synthesis
is_contr = main_control()

if (is_contr == 0)
exit(1)
else
exit(0)
endif
```

Figure A5.13. *Skeleton of the .m file*

```
##############################
# CONSTANTS OF THE ANALYSIS #
##############################

global max_depth;
global file_to_carto;
global pattern_max_length;
global total_pattern_tried = 0;

########################
#     BOOST CONSTANTS     #
########################
global example_name = "boost";
global numberModes = 2;

global x_c = 70;
global x_l = 3;
global r_c = 0.005;
global r_l= 0.05;
global r_0 = 1;
global v_s = 0.8;
global A_1 = [-r_l/x_l , 0
                  0, -1/(x_c*(r_0 + r_c))] ;
global A_2 = [-(r_l + r_0*r_c/(r_0+r_c))/x_l ,
                     -1/x_l * (r_0/(r_0+r_c)) ;
               1/x_c * (r_0/(r_0+r_c)) ,
                     -1/x_c * (1/(r_0+r_c))];
global B = [v_s/x_l ; 0] ;

function C = alpha_c(mode)
        global A_1;
        global A_2;
        global B;
        global tau;

        if (mode == 1)
                C = expm(A_1*tau);
        elseif (mode == 2)
                C = expm(A_2*tau);
        endif
endfunction

function D = beta_c(mode)
        global A_1;
        global A_2;
        global B;
        global tau;

        if (mode == 1)
                D = (expm(A_1*tau)-eye(2))*inv(A_1)*B;
        elseif (mode == 2)
                D = (expm(A_2*tau)-eye(2))*inv(A_2)*B;
        endif
endfunction

# Start control synthesis
is_contr = main_control()

if (is_contr == 0)
  exit(1)
else
  exit(0)
endif
```

Figure A5.14. *Boost converter: .m file*

A skeleton of such a file is shown in Figure A5.15, and the file for Boost is shown in Figure A5.16. “BENCH” is filled with the example name, MAXLENGTH with the maximal length admissible for patterns, MAXDEPTH with the maximal depth of decomposition, R with the box R that we wish to control, TAU with τ the time step and POST with the depth at which we want to perform the computation of $Post_{\Delta}$. HULL and BBOX are parameters to ease the computation of $Post_{\Delta}$. Every $Hull$ number of steps in the computation, the convex hull of all the polyhedra included in each part of the decomposition will be computed to reduce the number of polyhedra. Similarly, every $Bbox$ number of steps, the bounding box of those polyhedra will be computed. This greatly reduces the computation time at the cost of an overapproximation. They can be set to -1 if we want to compute the exact $Post_{\Delta}$.

```
#!/bin/sh

export BENCH="example_name"

export MAXLENGTH= l
export MAXDEPTH= d
export R= "[l_1,u_1]*[l_2,u_2]*...*[l_n,u_n]"
export TAU= tau

export POST= Post
export HULL= Hull
export BBOX= Bbox

./run.sh
```

Figure A5.15. *Skeleton of the script shell file*

Now, to run the tool on your example, simply execute the script shell file for your example.

```
#!/bin/sh

export BENCH="boost"

export MAXLENGTH=5
export MAXDEPTH=4
export R="[1.55,2.15]*[1.0,1.4]"
export TAU=0.1

export POST=10
export HULL=2
export BBOX=-1

./run.sh
```

Figure A5.16. *Script shell file of the Boost example*

For more details about the tool and how to use it, see the tool's webpage [MIN 13].

Bibliography

[ABA 09] ABATE A., TIWARI A., SASTRY S., "Box invariance in biologically-inspired dynamical systems", *Automatica*, vol. 45, no. 7, pp. 1601–1610, 2009.

[ALT 07] ALTHOFF M., STURSBERG O., BUSS M., "Reachability analysis of linear systems with uncertain parameters and inputs", *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orleans, Louisiana, USA, pp. 726–732, 12–14 December 2007.

[ALT 08] ALTHOFF M., STURSBERG O., BUSS M., "Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization", *Proceedings of the 47th IEEE Conference on Decision and Control (CDC)*, IEEE, Cancun, Mexico, pp. 4042–4048, 9–11 December 2008.

[AMI 12] AMIN BEN SASSI M., TESTYLIER R., DANG T., *et al.*, "Reachability analysis of polynomial systems using linear programming relaxations", in CHAKRAHORTY S., MUKUND M. (eds), *10th International Symposium on Automated Technology for Verification and Analysis (ATVA)*, Lecture Notes in Computer Science, Thiruvananthapuram, India, vol. 7561, Springer, pp. 137–151, October 2012.

[AND 13] ANDRÉ É., SOULAT R., *The Inverse Method*, ISTE, London, John Wiley and Sons, New York, 2013.

[ANT 02] ANTSAKLIS P., KOUTSOUKOS X., "Hybrid systems control", *Encyclopedia of Physical Sciences and Technology*, Academic Press, vol. 7, pp. 445–458, 2002.

[ASA 00] ASARIN E., BOURNEZ O., DANG T., *et al.*, "Effective synthesis of switching controllers for linear systems", *Proceedings of the IEEE*, Special Issue on Hybrid Systems, vol. 88, no. 7, pp. 1011–1025, 2000.

[ASA 02] ASARIN E., DANG T., MALER O., "The d/dt tool for verification of hybrid systems", in BRINKSMA E., LARSEN K.G. (eds), *Computer Aided Verification (CAV)*, Lecture Notes in Computer Science, Copenhagen, Denmark, vol. 2404, Springer, pp. 365–370, July 2002.

[BEC 05] BECCUTI A., PAPAFOTIOU G., MORARI M., "Optimal control of the boost dc-dc converter", *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC '05)*, pp. 4457–4462, December 2005.

[BEH 07] BEHRMANN G., COUGNARD A., DAVID A., *et al.*, "UPPAAL-Tiga: time for playing games!", in DAMM W., HERMANNS H. (eds), *Computer Aided Verification (CAV)*, Lecture Notes in Computer Science, Berlin, Germany, vol. 4590, Springer, pp. 121–125, July 2007.

[BER 00] BERTSEKAS D., *Dynamic Programming And Optimal Control*, 2nd ed., Athena Scientific, 2000.

[BLA 99] BLANCHINI F., "Set invariance in control", *Automatica*, vol. 35, pp. 1747–1767, 1999.

[BRO 83] BROCKETT R.W., "Asymptotic stability and feedback stabilization", *Differential Geometric Control Theory*, Defense Technical Information Center, vol. 27, pp. 181–191, 1983.

[BRO 00] BROCKETT R., LIBERZON D., "Quantized feedback stabilization of linear systems", *IEEE Transactions on Automatic Control*, vol. 45, no. 7, pp. 1279–1289, 2000.

[BUI 05] BUISSON J., RICHARD P.-Y., CORMERAIS H., "On the stabilisation of switching electrical power converters", *Hybrid Systems: Computation and Control (HSCC)*, Lecture Notes in Computer Science, Zurich, Switzerland, vol. 3414, Springer, pp. 184–197, March 2005.

[CAS 09] CASSEZ F., JESSEN J.J., LARSEN K.G., *et al.*, "Automatic synthesis of robust and optimal controllers–an industrial case study", in MAJUMDAR R., TABUADA P. (eds), *HSCC*, Lecture Notes in Computer Science, vol. 5469, Springer, pp. 90–104, 2009.

[CER 09] CERVANTES I., PEREZ-PINAL F., MENDOZA-TORRES A., "Hybrid control of DC-DC power converters", in HAMMONS T.J. (ed.), *Renewable Energy*, InTech, Chapter 10, 2009.

[DIN 11] DING J., LI E., HUANG H., *et al.*, “Reachability-based synthesis of feedback policies for motion planning under bounded disturbances”, *IEEE International Conference on Robotics and Automation (ICRA’11)*, Shanghai, China, pp. 2160–2165, 9–13 May 2011.

[DU 09] DU Z., TOLBERT L., OZPINECI B., *et al.*, “Fundamental frequency switching strategies of a seven-level hybrid cascaded h-bridge multilevel inverter”, *IEEE Transactions on Power Electronics*, vol. 24, no. 1, pp. 25–33, January 2009.

[FEL 12a] FELD G., FRIBOURG L., LABROUSSE D., *et al.*, Control of multilevel power converters using formal methods, Research Report no. LSV-12–14, Laboratoire Spécification et Vérification, ENS Cachan, France, p. 14, June 2012.

[FEL 12b] FELD G., FRIBOURG L., LABROUSSE D., *et al.*, Correct By Design Control Of 5-Level And 7-Level Converters, Research Report Num. Lsv-12-25, Laboratoire Spécification et Vérification, ENS Cachan, France, December 2012.

[FLI 06] FLIELLER D., RIEDINGER P., LOUIS J.-P., “Computation and stability of limit cycles in hybrid systems”, *Nonlinear Analysis*, vol. 64, no. 2, pp. 352–367, 2006.

[FRE 08] FREHSE G., “PHAVer: algorithmic verification of hybrid systems past HyTech”, *International Journal on Software Tools for Technology Transfer (STTT)*, Berlin, Heidelberg, vol. 10, Springer, no. 3, pp. 263–279, May 2008.

[FRE 11] FREHSE G., LE GUERNIC C., DONZÉ A., *et al.*, “SpaceEx: scalable verification of hybrid systems”, in GOPALAKRISHNAN G., QADEER S. (eds), *Computer Aided Verification (CAV)*, Lecture Notes in Computer Science, Snowbird, Utah, USA, vol. 6806, Springer, pp. 379–395, July 2011.

[GEY 08] GEYER T., PAPAFOTIOU G., MORARI M., “Hybrid model predictive control of the step-down DC–DC converter”, *IEEE Transactions on Control System Technology*, vol. 16, no. 6, pp. 1112–1124, January 2008.

[GIR 05a] GIRARD A., “Reachability of uncertain linear systems using zonotopes”, in MORARI M., THIELE L. (eds), *Hybrid Systems: Computation and Control (HSCC)*, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, vol. 3414, pp. 291–305, 2005.

[GIR 05b] GIRARD A., PAPPAS G.J., "Approximation metrics for discrete and continuous systems", *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 782–798, 2005.

[GIR 10a] GIRARD A., "Synthesis using approximately bisimilar abstractions: state-feedback controllers for safety specifications", *HSCC*, pp. 111–120, 2010.

[GIR 10b] GIRARD A., POLA G., TABUADA P., "Approximately bisimilar symbolic models for incrementally stable switched systems", *IEEE Transactions on Automatic Control*, vol. 55, pp. 116–126, 2010.

[GIR 12] GIRARD A., "Low-complexity switching controllers for safety using symbolic models", in HEEMELS M., DE SCHUTTER B., LAZAR M. (eds), *4th IFAC Conference on Analysis and Design of Hybrid Systems*, Eindhoven, The Netherlands, June 2012.

[GON 03] GONÇALVES J.M., "Region of stability for limit cycles of piecewise linear systems", *IEEE Conference on Decision and Control*, Maui, HI, 9–12 December 2003.

[HEN 96] HENZINGER T.A., "The theory of hybrid automata", *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS '96)*, Washington, DC, IEEE Computer Society, pp. 278–292, 1996.

[HIS 01] HISKENS I., "Stability of limit cycles in hybrid systems", *34th Annual Hawaii International Conference on System Sciences (HICSS-34)*, Maui, HI, 3–6 January 2001.

[JAU 01] JAULIN L., KIEFFER M., DIDRIT O., *et al.*, *Applied Interval Analysis*, 1st edition, Springer, September 2001.

[KÜH 98] KÜHN W., "Zonotope dynamics in numerical quality control", in HEGE H.-C. (ed.), *Mathematical Visualization*, Springer, Berlin, Heidelberg, pp. 125–134, 1998.

[LAS 61] LASALLE J., LEFSCHETZ S., *Stability by Lyapunov's Direct Method*, Academic Press, 1961.

[LEG 09] LE GUERNIC C., GIRARD A., "Reachability analysis of hybrid systems using support functions", in BOUAJJANI A., MALER O. (eds), *CAV*, Lecture Notes in Computer Science, vol. 5643, Springer, pp. 540–554, 2009.

[LIB 99] LIBERZON D., MORSE A., "Basic problems in stability and design of switched systems", *Control Systems*, vol. 19, no. 5, pp. 59–70, 1999.

[LIB 03] LIBERZON D., *Switching in Systems and Control*, Birkhausen, 2003.

[LYG 99] LYGEROS J., TOMLIN C.J., SASTRY S., "Controllers for reachability specifications for hybrid systems", *Automatica*, vol. 35, no. 3, pp. 349–370, 1999.

[MEY 92] MEYNARD T., FOCH H., "Multi-level conversion: high voltage choppers and voltage-source inverters", *23rd Annual IEEE Power Electronics Specialists Conference*, vol. 1, pp. 397–403, June–3 July 1992.

[MIN 13] MINIMATOR TEAM, "Minimator web page", available at http://www.lsv.ens-cachan.fr/~soulat/minimator, 2013.

[MIT 07] MITCHELL I.M., "Comparing forward and backward reachability as tools for safety analysis", *Proceedings of the 10th International Conference on Hybrid Systems: Computation and Control, (HSCC'07)*, Springer, Pisa, Italy, no. 16, pp. 428–443, 2007.

[OCA 13] OCAML TEAM, "OCaml web page", available at http://caml. inria.fr/ocaml/index.fr.html, 2013.

[OCT 13] OCTAVE TEAM, "Octave web page", available at http://www.gnu.org/software/octave/, 2013.

[PAT 09] PATINO GUEVARA D., Pilotage des cycles limites dans les systèmes dynamiques hybrides. Application aux alimentations électriques statiques, Phd Thesis, Nancy University, 2009.

[PAT 05] PICASSO B., BICCHI A., "Control synthesis for practical stabilization of quantized linear systems", *Rend. Sem. Mat. Univ. Pol. Torino, Control Theory and Stabil., I*, vol. 63, no. 4, pp. 397–410, 2005.

[PIC 08] PICASSO B., BICCHI A., "Rendiconti del Seminario Matematico Università e Politecnico di Torino, Control Theory and Stabilization, I", *Nonlinear Analysis: Hybrid Systems*, vol. 2, no. 3, pp. 706–720, 2008.

[PLE 13] PLECS TEAM, "PLECS web page", available at http://www. plexim.com, 2013.

[PPL 13] PPL TEAM, "PPL web page", available at http://bugseng. com/products/ppl/, 2013.

[RAI 98] RAISCH J., O'YOUNG S., "Discrete approximation and supervisory control of continuous systems", *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 569–573, 1998.

[RAM 89] RAMADGE P.J., WONHAM W.M., "The control of discrete event systems", *Proceedings of the IEEE*, vol. 77, no. 1, pp. 81–98, January 1989.

[RUB 00] RUBENSSON M., LENNARTSON B., "Stability of limit cycles in hybrid systems using discrete-time Lyapunov techniques", *Proceedings of the 39th IEEE Conference on Decision and Control*, Sydney, Australia, 12–15 December 2000.

[RUN 13] RUNGGER M., MAZO M., TABUADA P., "Specification-guided controller synthesis for linear systems and safe linear-time temporal logic", *Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control (HSCC)* , pp. 333–342, 2013.

[SAN 93] SANDERS S., "On limit cycles and the describing function method in periodically switched circuits", *IEEE Transactions on Circuits and Systems*, vol. 40, no. 9, pp. 564–572, 1993.

[SEN 03] SENESKY M., EIREA G., KOO T.-J., "Hybrid modelling and control of power electronics", *Proceedings of the 6th International Conference on Hybrid Systems: Computation and Control, (HSCC'07)*, Springer, Prague, Czech Republic, no. 16, pp. 450–465, 2003.

[SUN 05] SUN Z., GE S., *Switched Linear Systems: Control and Design*, Springer, 2005.

[TAB 05] TABUADA P., "Symbolic sub-systems and symbolic control of linear systems", *44th IEEE Conference on Decision and Control 2005 and 2005 European Control Conference (CDC-ECC '05)* , pp. 18–23, 2005.

[TAB 08] TABUADA P., "An approximate simulation approach to symbolic control", *IEEE Transactions on Automatic Control*, vol. 53, no. 6, pp. 1406–1418, 2008.

[TAB 09] TABUADA P., *Verification And Control Of Hybrid Systems: A Symbolic Approach*, 1st ed., Springer, 2009.

[TAR 05] TARANTOLA A., *Inverse Problem Theory and Methods for Model Parameter Estimation*, SIAM, 2005.

[TEA 08] TEAM S., "Simulink web page", available at http://www.mathworks.com/products/simulink/, 2008.

[TOM 00] TOMLIN C., LYGEROS J., SASTRY S., "A game-theoretic approach to controller design for hybrid systems", *Proceedings of the IEEE*, vol. 88, no. 7, pp. 949–970, 2000.

Index