Combinatorial Hybrid Systems

Ph.D. thesis

Jesper A. Larsen

Department of Electronic Systems
Section for Automation and Control
Aalborg University
Fredrik Bajers Vej 7, 9220 Aalborg East, Denmark

Combinatorial Hybrid Systems
Ph.D. thesis

ISBN 978-87-92328-11-3
February 2009

This thesis was typeset using LaTeX 2_ε in `report` document class.

Preface

This thesis is submitted as partly fulfillment of the requirements for the Doctor of Philosophy at the Section of Automation and Control, Department of Electronic Systems, Aalborg University, Denmark. The work has been carried out in the period September 2005 to February 2009 under the supervision of Professor Rafael Wisniewski and Associate Professor Roozbeh Izadi-Zamanabadi.

Aalborg University, February 2009
Jesper A. Larsen

Abstract

Combinatorial Hybrid Systems

This thesis treats the theory of hybrid systems and presents the reader with a unified framework for modelling hybrid systems; the framework accepts natural limitations of not only the system itself, but also limitations on the possible inputs to the system. The framework uses a fully discrete abstraction of the continuous dynamics of the hybrid system, however consepts from classical continuous control theory are preserved.

Previous ideas in this direction comes from the works of Tabuada and Pappas on Model Checking LTL on controllable linear systems, where it is shown that model checking, as known in computer science, can be used for checking properties of linear systems, and from van Schuppen, Collins and Habets, who have shown sufficient conditions that guarantee an affine system to cross a given hyper-plane in finite time. These two ideas will be discussed and examplified in the first part of this thesis.

Based on this, the desired properties for a new modeling framework are derived and presented in the second part of this thesis. Amongst other properties, some of the requirements to the framework are, that such consepts as vector fields, flow lines and Lyapunov functions should be preserved. From these requirements and based on the theories developed by R. Forman two similar frameworks have been developed and are presented. The first framework preserves some of the topological properties of Forman's work which results in a nice mathematical formulation, whereas the second framework is purely geometric oriented, which allows for more freedom in the mathematical formulation and thereby gets a better and more intuitive relation to the original underlying dynamical system.

In practice a discrete equivalent of the continuous dynamics is derived, by splitting the continuous state space into a polyhedra complex, and on each of its polytopes approximate the dynamics with an affine system on which the possible flows to the polytopes facets can be expressed. Thereby control is abstracted into a selection of which subset of facets of a given polytope should be considered as exit facets. Thus a purely discrete abstraction of the entire hybrid system can be found and tools from computer science can be applied directly.

In the final part ways of extending the work are discussed. Amongst others it is discussed how optimal control can be included in the work along with possible ways of refining the division of the system dynamics.

Synopsis

Kombinatoriske Hybridsystemer

Denne afhandling behandler teorien bag hybridsystemer og præsentere læseren for en forenet modelleringsramme for hybridsystemer; denne modelleringsform faciliterer naturlige begrænsninger af ikke kun systemet, men også input begrænsninger af systemet. Modelleringsformen bruger en fuldtud diskret abstraktion af hybridsystemets kontinuerte dynamik, dog på en sådan måde, at koncepter fra klassisk kontinuert reguleringsteori er bevaret.

Tidligere ideer i denne retning kommer fra Tabuada og Pappas arbejde med modeltjekkende LTL på kontrollerbare lineære systemer, hvor det er vist at modeltjekning, som det er kendt i computer science, kan anvendes til at undersøge egenskaber ved lineære systemer, og fra van Schuppen, Collins og Habets, som har vist tilstrækkelige betingelser til at garantere at et affint system vil krydse et givent hyperplan i endelig tid. Disse to ideer vil blive diskutteret og eksemplificeret i afhandlingens første del.

Baseret på dette bliver de ønskede egenskaber for en modelleringsformalisme udledt og præsenteret i den anden del af afhandlingen. Nogle af egenskaberne, som bør bevares er koncepter såsom vektor felter, flow linier og Lyapunov funktioner. På basis af disse krav og baseret på teorier udviklet af R. Forman er to lignende modelleringsformalismer blevet udviklet og præsentert. Den første modelleringsformalisme bevarer nogle af de topologiske egenskaber i Formans arbejde, hvilket resulterer i en pæn matematisk formulering, hvorimod den anden formalisme er rent geometrisk orienteret, hvilket giver mere frihed i den matematiske formulering og derved muliggør en tættere og mere intuitiv relation til det oprindelige underlæggende dynamiske system.

I praksis bliver den diskrete ækvivalent af systemet lavet ved at opdele det kontinuerte tilstandsrum i et polyhedralt kompleks, og på hver af polyhederne tilnærme systemets dynamik med et affint system, hvorpå det er muligt at udlede hvike facetter af polytopen som der kan styres til. Herved bliver kontrollen abstraheret til et valg af hvike facetter som man gerne vil forlade polytopen igennem. Herved kan man lave en ret diskret abstraktion af hele hybrid systemet og værktøjer fra computer science kan direkte anvendes til kontrolsyntese og -validering.

I den sidste del af afhandlingen bliver forskellige udvidelsesmuligheder behandlet. Bl.a. diskuteres der hvordan optimal kontrol kan blive indkorporeret samt muligheder for at rafinere opdelingen af systemet.

Acknowledgments

I would like to acknowledge my supervisor Rafael Wisniewski for his continuous support and guidance during the past three years. Rafael has been a great help in solving many theoretical problems encountered during this research. Also my initial co-supervisor Roozbeh Izadi-Zamanabadi, who left for fame and fortune in the private industri, deserves thanks for bringing a more tangable view of the things into play when things became too theoretical.

I would like to thank everyone, students as well as employees at Aalborg University and others, who have participated in our work with hybrid systems and participated in our hybrid systems group. This has lead to many interesting and fruitfull discussions through the past years.

I was a guest at Centrum Wiskunde & Informatica in Amsterdam during the autumn of 2007 and I would like to thank Professor Jan H. van Schuppen and Dr. Pieter Collins for making this stay possible and involving me in their research group and work.

Finally the greatest acknowledgment goes to my wife Vaida for her support and patience during these past three years.

Contents

Part I

Introduction

This part gives a general introduction to the background and motivation of this research. A review of the literature in the different fields examined in this thesis is given and a general methodology combining the contributions is presented.

Chapters

Chapter 1

Introduction

In the very heart of modern society lies control. One of the earliest usages of feed back control is probably the use for regulating flow in water clocks in the 16^{th} century B.C. Later on it played a key role during the industrial revolution with Watts invention of the flyball governor, which lead to a wide spread use of manufacturing machines. Such simple proportional feed back regulation systems are omnipresent today, from the mechanical thermostats in house hold radiators to fuel level control in car carburators.

Today, most of such simple controllers are still implemented purely mechanical due to costs, but as the price for micro controllers capable of handling the tasks are continuing to go down in price more and more control areas are taken over by a digital controller of some form, one example of such is the afforementioned carburator, which in all newer cars today is replaced by an electronic fuel injection system, with added sensors and a micro controller for steering the motor revolutions.

All such controllers have traditionally been single set-point controllers working in a homogenious environment. For instance, a house hold radiator thermostat is put at a certain temperature and keeps it there. Likewise the fuel injection system gets a set-point from the throttle and tries to generate a fuel consumption proportional to this. However, as the processing capabilities of embedded micro controllers and the societies demands goes up, the possibilities, and the demands for more advanced control does the same.

Some of these areas are for instance the desires to develop autonomous robots, this being either under water, on land or in the air and development of supervisory control systems/strategies for manufacturing and processing plants under possible changes in it sensor, actuator and dynamic structure.

Such requirements can formally be captured within the modelling capabilities of hybrid systems which this chapter will seek to introduce some of the basic concepts of.

First of all a clarification is needed. Since the word *hybrid* has become a much hyped buzz word during the last few years, especially in connection with *hybrid* cars and *hybrid* approaches, *hybrid* systems in this context means a connection of dynamical systems. Another, probably more correct word for the type of systems dealt with would be switched dynamical systems. However, this definition is often used when talking about purely deterministic systems, which is not the case in general.

In the remainder of this work this is what is to be understood as a hybrid system:

Definition 1. *A* **Hybrid System** *is a system composed of*

- *A finite set of dynamic systems defined on manifolds.*

- *A finite set of transfer maps, which maps a subset of one manifold to another.*

Such systems appear everywhere in almost any system. One good example of a hybrid system is a normal car with a manual transmission. Driving the car along in 1^{st} gear, and the car has one specific dynamic. Changing the gear into 2^{nd} suddenly gives another and different dynamic. This is what is known as a hybrid system with external events (the driver changing gear). Taking again the same car, but now with an automatic transmission, then there would still be a different dynamical system for each gear, however, the gearbox will be switching between them according to a set of rules, such as: if $speed > 20\frac{km}{h}$ switch to second. Such hybrid system with a controller included is sometimes referred to as a hybrid system with internal events.

Other examples of situations, where hybrid systems naturally occur:

- Manufacturing systems

- Fermentation processes

- Batch processes

- Electrical networks

- Rigid body interaction

These are just a few of the cases in which hybrid modeling techniques are used to express the dynamics of the system. Even though a given system might appear to be linear, then it is normally only valid in a small interval around a given operating point. Going beyond the validity of the models operating range various extra phenomena needs to be included, and in general hybrid systems occurs, when trying to extend linear models with natural limitations, such as:

- Saturation - i.e. power supply limitations, end point limitations, rate of change limitations

- Hysteresis - i.e. slip in gears, delays before actuation

- Faults - i.e. a sensor or actuator fails to work correctly

It is now possible to put a bit more structure on definition 1. However - the more structure which is put on the definition the more we can reason about it, but it come at the cost of describing a narrower system. An often used approach for adding structure to such systems is in the form of [Henzinger, 1996a], which describes a hybrid automaton as:

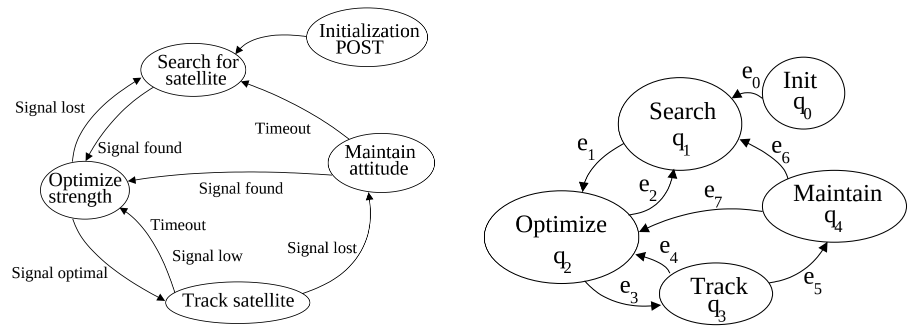
Definition 2. *A hybrid automaton H is a collection $H = (Q, X, Init, f, J, E, G, R)$, where:*
- Q: set of discrete variables, also sometimes called locations and Q is countable, i.e. $q \in Q \subset \mathbb{Z}$
- X: set of continuous variables, i.e. $x \in X \subseteq \mathbb{R}^n$, $n \in \mathbb{N}$
- $Init \subseteq Q \times X$ is a set of initial states
- f: $Q \times X \rightarrow T(X)$ is a Lipschitz vector field where T is a tangent bundle of the manifold X describing the continuous dynamics
- J: $Q \rightarrow P(X)$ invariants,assigns to each $q \in Q$ a domain
- $E \subset Q \times Q$ is a collection of discrete transitions from q to q'
- G: $E \times X \rightarrow P(X)$ assigns to $e = (q, q') \in E$ a guard, $g(e, x)$
- R: $E \times X \rightarrow P(X)$ assigns to (e, x), where $e = (q, q') \in E$ and $x \in X$, a reset relation, $r(e, x)$

An example of a system modelled as a hybrid system is given in the following and is based on the supervisory control system for a ship-mounted satellite tracking antenna.

The overall strategy for controlling the antenna is illustrated in figure 1.1(a) along with the overall control states. Initially the system will enter the upper-right state, where initialization and Power On Self Test (POST) will be conducted. If the system passes the POST it will start to search for a satellite. This searching can be conducted in a number of ways which will be discussed later. If a satellite signal is found the “Optimize strength” state will be entered. This is done to ensure maximum satellite strength reception. If, during this process the signal should be lost again the system will return back to the search state. Otherwise it will go to the “Track satellite” state, where the current attitude in the SCESF[1] frame will be maintained. With a regular interval, or when the signal strength has dropped a bit, the system will reenter the “Optimize strength” state to ensure that the received signal is optimal. Should the received signal however be lost abruptly[2] the antenna will maintain its attitude in the SCESF frame. This will be done either until the signal has been received again or a timeout is reached.

The overall system, as described in connection with figure 1.1(a), can be reformulated in a hybrid system context. There are two interesting signal levels in this context, that is the lower threshold where the signal is detectable, s_d, and the upper threshold where the signal strength is sufficiently close to the optimal, s_s. The timeout for passing under a bridge is defined to be $t_1 = 30$s and the strength optimization timeout is selected to be

[1]Ship Centered Earth Surface Fixed reference frame
[2]Passing under a bridge or similar



(a) Overall control strategy (b) The control strategy as a hybrid system

Figure 1.1: Illustration of the overall control strategy described intuitively on the left side and as a hybrid system on the right side

$t_2 = 10$s. The system can now be described as:

Cont. states	$X = \{x \mid x = \begin{bmatrix}\theta & \phi & s & t\end{bmatrix}^T, \theta \in \mathbb{R},\ \phi \in 0 \leq \phi \leq 120\,\text{deg} \subset \mathbb{R},$ $s \in \mathbb{R}, t \in \mathbb{R}^+\}$
Disc. states	$Q = \{\text{Initialize, Search, Optimize, Track, Maintain}\}$ $= \{q_0, q_1, q_2, q_3, q_4\}$
Initial state	$Init = \{(q, x) \mid q = q_0, x \in X\}$
Invariants	$J(q_0) = \{x \in X \mid \begin{bmatrix}\theta & \phi\end{bmatrix}^T \neq 0\}$ $J(q_1) = \{x \in X \mid s < s_d\}$ $J(q_2) = \{x \in X \mid s < s_s \wedge s > s_d\}$ $J(q_3) = \{x \in X \mid s > s_s \wedge t < t_2\}$ $J(q_4) = \{x \in X \mid s < s_d \wedge t < t_1\}$
Transitions	$E = \{e_0 = (q_0, q_1),\ e_1 = (q_1, q_2),\ e_2 = (q_2, q_1),\ e_3 = (q_2, q_3),$ $e_4 = (q_3, q_2), e_5 = (q_3, q_4), e_6 = (q_4, q_1), e_7 = (q_4, q_2)\}$
Guards	$G = \{g(e_2) = (s < s_d),\ g(e_3) = (s > s_s),$ $g(e_4) = (t > t_2 \vee s_d < s < s_s), g(e_5) = (s < s_d),\ g(e_6) = (t > t_1),$ $g(e_7) = (s > s_d)\}$
Resets	$R = r(e_0) = r(e_1) = r(e_2) = r(e_3) = r(e_4) = r(e_5) = r(e_6) = r(e_7)$ $= \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0\end{bmatrix} x$

Using this formulation the figure, 1.1(a), becomes the system described by figure 1.1(b). As can be seen from the continuous state definition then the antenna bore axis is described by the spherical coordinate pair, $[\theta, \phi]^T$, since this is sufficient to describe the antenna orientation. Thus the inverse kinematics used to translate between the SCESF spherical coordinates and the joint space is handled at a lower level.

As can be seen from the example, then even though the problem in question, e.g. the supervisory control challenge, can be rater difficult to formulate by words, and even impossible using classical control terminology, then it is possible to express the precise behavior of the system using a hybrid systems formalism.

1.1 Outline of the Thesis

The remaining part of this part consists of the following:

- Chapter 2 gives a brief overview of different ways of modelling deterministic hybrid systems.

- In chapter 3 an example of how hybrid control and verification of a real-world plant using the theories of Tabuada and Pappas, and implemented in UPPAAL is shown. This chapter has been presented at the 10^{th} International Conference on Hybrid Systems: Computation and Control, and is published in Lecture Notes in Computer Science nr. 4416.

- Another examle is given in chapter 4, where modelling of satellite formation flying is performed using piece wise affine hybrid systems theory. This paper was presented at the 3rd International Symposium on Formation Flying, Missions and Technologies and published in the conference proceedings by the ESA Communication Production Office.

- Finally chapter 5 gives an overview of the ideas behind the framework, which will be presented in the second part of this thesis. This paper has been accepted for presentation at the 7^{th} International Workshop on Robot Motion and Control and publication in Lecture Notes in Control and Information Sciences.

Part II of this thesis introduces the concept of a combinatorial hybrid system in the following way:

- Chapter 6 gives a short introduction to previous work in the direction of combinatorial dynamics and an introduction to the two proposed frameworks for combinatorial hybrid systems. Finally this chaper consists of a comparative analysis of the different methods presented.

- Chapter 7 introduces the first framework. This work has been presented at the 9^{th} IEEE International Conference on Computer-Aided Control Systems and published in the associated proceedings.

- This framework is further elaborated on in chapter 8, where the framework is shown applied to fault tolerant control for autonomous satellites. The content of this chapter was presented at the The F. Landis Markley Astronautics Symposium and published in the Vol. 132 of Advances in the Astronautical Sciences.

- Chapter 9 contains the second framework. This framework carries over many of the ideas of the first framework, however, it is more intuitive in its formulation and the resulting flowlines have nicer properties. This work was presented in a reduced form at the 17^{th} IFAC World Congress and is currently under review for publication in Nonlinear Analysis: Hybrid Systems, published by Elsevier.

The final part III contains the conclusions of the report along with a presentation of possible ways of extending the work to also include optimal control and ways of simplifying the combinatorial hybrid systems.

Chapter 2

Overview of Hybrid Systems

Over the years a number of different hybrid systems modelling languages have been developed for handling deterministic hybrid systems, each with their particular strengths and weaknesses. A few of such modelling frameworks are:

- Mixed Logic Dynamics
- Linear Complementory
- Max-Min-Plus-Scaling
- Piecewise Affine

which briefly will be touched upon in the following.

2.1 Mixed Logic Dynamics

Mixed Logic Dynamics is a technique for modelling systems through the physical laws of the system, logic rules and operating constraints. It is inherently capable of handling both continuous and discrete inputs and state variables, thus making it suitable for modelling not only hybrid systems, but also sequential logical systems and constrained linear systems, just to name a few. A more detailed overview of the capabilities of MLD is given in [Bemporad and Morari, 1999a].

Each component of the system is modelled as an inequality, e.g. a linear dynamical system can be modelled as

$$\begin{aligned} x(t+1) = & \quad Ax(t) + B_1u(t) + B_2\delta(t) + B_3z(t) \\ y(t) = & \quad Cx(t) + D_1u(t) + D_2\delta(t) + D_3z(t) \\ E_2\delta(t) + & \quad E_3z(t) \leq E_1u(t) + E_4x(t) + E_5 \end{aligned}$$

with $x = \begin{bmatrix} x_c \\ x_l \end{bmatrix}$ describing the combined continuous and logic state space, with $x_c \in \mathbb{R}^{n_c}$ and $x_l \in \{0,1\}^{n_l}$, y being the corresponding combined output vector, w is a disturbance vector containing both continuous disturbances (such as noice and measuring errors) along with binary disturbances (such as faults). Finally $\delta \in \{0,1\}^{r_l}$ and $z \in \mathbb{R}^{r_c}$ contains additional information used to introduce logic propositions as linear inequalities, which are expressed in the third line of the mode [Morari *et al.*, 1999; Bemporad and Morari, 1999b].

Instead of going into the details of the modelling language an example of how it works will be given instead. The example is equivalent to the one shown in [Bemporad and Morari, 1999a].

Given the piecewise linear system, which could be seen as a model of a car which, for $x \geq 0$ is going down a hill and for $x > 0$ goes up a hill:

$$\begin{aligned} \dot{x}_1 &= 0.5x + u, && \text{for } x \geq 0 \\ \dot{x}_2 &= -0.4x + u, && \text{for } x > 0, \end{aligned}$$

with state constraint $-100 \leq x \leq 100$ and input constraint $-5 \leq u \leq 5$. Introducing the binary value δ and letting it express, that $x \geq 0$, i.e. $\delta = 1 \leftrightarrow x \geq 0$, which means, that the state constraints needs to fulfil

$$\begin{aligned} -m\delta x &\leq x - m \\ -(M + \epsilon)\delta &\leq -x - \epsilon, \end{aligned}$$

where $m = \min x$ and $M = \max x$, and $\epsilon > 0$ is a small alowable tolerance.

Then the differential equations describing the two systems can be written together into one as:

$$\dot{x} = 0.9\delta x - 0.4x + u$$

Here, however, a product term appears in the form of δx. This is handled by introducing the new variable $z = \delta x$, which has the following properties:

$$\begin{aligned} &z \leq M\delta \\ &z \geq m\delta \\ &z \leq x - m(1 - \delta)) \\ &z \geq x - M(1 - \delta)) \end{aligned}$$

Thus through this transformation the combined system can be expressed as:

$$\dot{x} = 0.9z - 0.4x + u.$$

This formulation has the nice property, that it is computationally sound, i.e. it is closely related to a form in wich a computer would be able to handle it.

Typical questions, which such modells are used to answer are:

- Given a set of initial conditions, $x(0) \in X$, verify, that $\forall w \in W : x(t) \notin X_u$, where X_u is an unsafe set. Such problems are often encountered when checking a system for security, or a progam for dead-locks. This checking can be done through Mixed-Integer Feasibility Testing.

- Given a set of initial condition , $x(0) \in X$, maximize a given cost function which is linear with respect to x, w, δ, z under the constraints of the given system, which is a classical optimal control problem, in this case solvable using Mixed-Integer Programming.

2.2 Linear Complementory Systems

Linear complementary systems are systems governed by

$$\begin{aligned} \dot{x}(t) = &\quad Ax(t) + Bu(t) \\ y(t) = &\quad Cx(t) + Du(t) \\ y(t), u(t) \geq 0 \wedge y(t)^T u(t) = 0. \end{aligned}$$

By this it is possible to describe systems, which have single bounds. Such systems have been studied in [Heemels *et al.*, 2000], and the papers refered within it.

LC systems are good to describe systems, which has only one constraint, such as e.g. unlateral constrained rigid body, ideal diodes, relays Coulomb fricition.

A super class of these systems are the Extended Linear Complementory Systems, which looks like

$$\begin{aligned} \dot{x}(t) = &\quad Ax(t) + B_1u(t) + B_2d(t) \\ y(t) = &\quad Cx(t) + D_1u(t) + D_2d(t) \\ E_1x(t) + &\quad E_2u(t) + E_3d(t) \leq g_4, \end{aligned}$$

with $d(t)$ being an auxiliary variable.

2.3 Max-Min-Plus-Scaling

As a further extention to ELC systems [Schutter and van den Boom, 2001] introduces the Max-Min-Plus-Scaling expression f of the variables $x_1, \dots, x_n$ is given by the grammar

$$f := x_i | \alpha | f_k \vee f_l | f_k \wedge f_l | f_k + f_l | \beta f_k,$$

with $i \in \{1, \dots, n\}$, $\wedge$ and $\vee$ denoting maximization and minimization respectively, and where f_k and f_l are again MMPS expressions. $|$ means ”or” in this formulation. It is worth noticing, that this formulation is quite broad in its formulation capacity, in the sense that it is recursively, i.e. the expression can either be variable, constants or the maximum or the minimum or the sum of two MMPS expressions or even a scalar multiple of an expression.

Typical examples of systems modelled as MMPS systems are telecommunication and computer networks, digital circuits and manufacturing plants with one of its main advantages being that it directly includes constraints on inputs and outputs.

2.4 Piecewise Affine

Piecewise affine systems have been studied extensively during the past 30 years since it is one of the "simplest" extentions of linear systems to also be able to model non-linear systems and hybrid systems with arbitrary accuracy.

Such systems are modelled as

$$\begin{aligned} \dot{x} =& \quad A_i x(t) + B_i u(t) + a_i \\ y(t) =& \quad C_i x(t) + D_i u(t) + c_i, \end{aligned}$$

for $x(t) \in \Gamma_i$ and $u(t) \in \Lambda_i$, with all Γ_i and Λ_i being convex polytopes.

An entire system can thus be modelled by a collection of PWA systems by having $\cup\Gamma$ and $\cup\Lambda$ forming a complete cover of the state and input space respectively.

2.5 Equivalense Between Models

In [Heemels *et al.*, 2001], all the models described above are compared, and it is shown how it is possible to transform systems described within one framework into a model in another framework. The main result from the article is shown in figure 2.1.

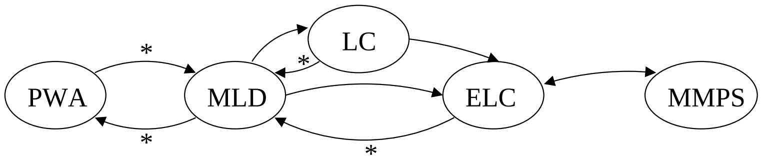


Figure 2.1: Graphical overview between the relations of the modelling frameworks. An arrow going from A to B means that A is a subset of B, furthermore, the star next to some arrows indicates, that additional conditions are required to establish the inclusion. [Heemels *et al.*, 2001]

2.6 Outro

In the following three chapters, three different approaches to modelling of hybrid systems will be introduced along with examples showing how they work and what they can be used for.

In chapter 3, it is shown how a switched dynamical system can be modelled and transformed into a Brunovsky normal form. It is further shown that having this, it is possible to specify the control objectives for the system as an LTL formulation, which was shown to be possible to validate automatically through the use of UPPAAL.

A completely different approach is shown in chapter 4, where it is shown how the randevouz problem of two satellites can be modelled as a PWA hybrid system, where the possible control actions are abstracted into a finite set of feed-back control laws guaranteeing exit through a given facet of one of the partitions. The system can then be abstrated into a finite automata on which the control objective can be formulated as a discrete game.

The final chapter of this part, i.e. chapter 5 introduces the framework of combinatorial control systems. Combinatorial control systems (CCS) carries over some of the ideas introduced in chapter 5, that is, the division of the system in question into a finite number of partitions, each with a PWA system on it. However, the systems are now considered purely combinatorial, leading to a greater abstraction of the system and towards a more computationally efficient paradigm for automatic control synthesis.

Chapter 3

Hybrid Control and Verification of a Pulsed Welding Process

Currently systems, which are desired to control, are becoming more and more complex and classical control theory objectives, such as stability or sensitivity, are often not sufficient to cover the control objectives of the systems. In this chapter it is shown how the dynamics of a pulsed welding process can be reformulated into a timed automaton hybrid setting and subsequently properties such as reachability and deadlock absence is verified by the simulation and verification tool UPPAAL.

3.1 Introduction

The lack of analytical methods for design of hybrid control systems can often result in excessive testing and validation, which is time consuming and even then might not guarantee that the system will meet the control objectives under all operating conditions. To overcome the design and implementation problems which may result from the deficient use of an analytical approach, a notation of hybrid automaton has been introduced in [Henzinger, 1996a].

Most algorithmic verification and synthesis tools for hybrid systems today are limited to systems exhibiting simple continuous dynamics, such as piecewise-affine hybrid systems[Bemporad *et al.*, 2000; Collins and van Schuppen, 2004] or timed automata[Feng, 2002; Alur and Dill, 1994; Alur and Madhusudan, 2004]. One of the main objective in [Tabuada and Pappas, 2003b] was to enlarge this class of systems to all linear controllable systems, which is continued in this paper by showing how this theory apply in practice to the Gas Metal Arc Welding (GMAW) process. By restricting the observations for the system to a finite set of partitions, enables a bisimulation of the system to be modeled using simple timed automata.

With a bisimilar model of the system built with the use of timed automata, it is possible to use a verification tools such as UPPAAL to simulate and verify different system properties. Especially questions such as reachability, liveliness and possibilities of deadlocks are new questions, which are of great interest to the designer of the supervisory system and which previously needed to be guessed by simulations or ad-hoc methods.

3.1.1 Gas Metal Arc Welding

In the GMAW process the electrode is consumable and is fed continuously at a certain rate by the pistol to the welding pool. The weld is protected from the surrounding air by a gas which is also fed by the pistol. Normally argon or argon/CO_2 is used as shielding gas. The current between the workpiece (cathode) and the welding pistol (anode) causes an arc and an electromagnetic field. The strong current makes the electrode melt and drop into the welding pool.

The GMAW process can be divided into three modes; short arc mode, spray mode and a mixed mode of the two, of which only the spray mode will be considered in this paper. In spray mode the electrode should never touch the workpiece in order to obtain the best weld quality.

The melting process can be described by two contributions, *anode heating* and *ohmic heating*. When the current rises, the temperature of the arc rises and the tip of the electrode is heated. The energy from the arc, which contributes to melting the electrode, is known as *anode heating*. The second contribution to the melting process is the *ohmic heating*, which is the heat energy resulting from the ohmic resistance in the electrode. The high current also creates a higher electromagnetic field which contributes, together with the gravitational force, to detachment of the drop.

While the tip is melting, a liquid drop of metal is formed. This drop is detached from the tip of the electrode when the surface tension on the drop, is too small to resist the gravitational- and the electromagnetic forces. Also the aerodynamic drag force from the shielding gas, contributes to the detachment of the drop. After detachment, a small liquid drop is left at the tip of the electrode and the process repeats.

A submode of the spray mode is the pulsed GMAW method, which is similar to spray mode, but in addition to the steady current between the cathode and the anode, current pulses causes the drop to detach in intervals. The advantage of using pulsed GMAW is a lower heat development in the weld pool. Furthermore the current pulses makes it possible to control the drop detachment [Thomsen, 2004].

3.1.2 Weld Quality Criteria

As described in the introduction, one of the objectives of this paper is to integrate a control structure for the GMAW process into a hybrid framework. The nature of the GMAW process makes classic control theory specifications, such as stability, inadequate. Instead control objectives focusing on obtaining the best weld quality is desirable. The quality of a weld depends on several factors, which will be discussed in the following.

Basically a high-quality weld is characterized by a good penetration, which is essential for a strong weld, as it allows a larger area of the workpiece edges to join.

A good penetration is a necessary, but not a sufficient, condition for a good weld. If the work piece becomes too hot and cools down too quickly, the material can loose some of its characterizing properties, e.g. heat-treated metals or metal alloys, such as stainless steel, can loose its characterizing properties. [Storer, 2004, ch. 5]
The facts described in the latter are related to the weld pool and are the minimum criteria, which must be fulfilled to obtain a high-quality weld and is defined as *direct weld quality influencing factors*. More indirectly an additional number of factors influences the quality of a weld. The following quality influencing factors will be referred to as *indirect quality influencing factors*. Specific for pulsed GMAW welding, the quality of the weld is influenced by the control of the drop detachment, meaning that the current pulses should ideally detach one drop per pulse to obtain the best weld possible. It is also desirable to obtain a uniform drop size, in order to achieve a homogeneous weld. An additional control objective is to keep a short arc length, since it is easier for the operator to work with. Moreover the energy input into the workpiece should be minimized.

The indirect quality influencing factors are related to the control of the electrode and the arc.

3.1.3 Delimitation of Control Tasks

As described in the latter the control can be separated into control of the weld pool, and control of the arc and electrode. As it is only hand held welding which will be the focus on this paper, the weld pool control is handled by the operator. Figure 3.1 describes the control structure [Thomsen, 2004]. The outer control loop is handled by the operator and the inner loop is handled by the welding machine. The rest of this paper will concentrate

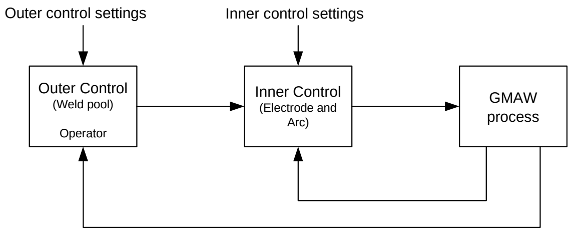
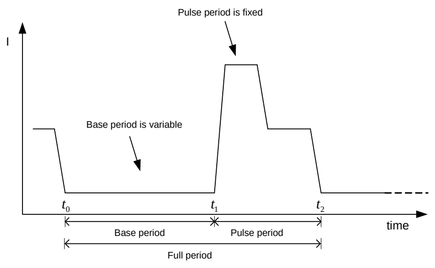
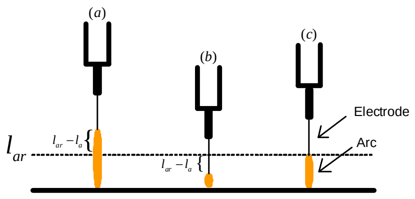


Figure 3.1: Control structure for the GMAW process.

on controlling the indirect quality influencing factors in the inner loop.



(a) Pulse by pulse method.



(b) Arc length control scenarios.

Figure 3.2: Left figure: The pulse by pulse method - the base current is fixed but the base period is variable. Right figure: The different arc length control scenarios - (a) the arc is too long (b) the arc is too short (c) the arc has the desired length l_{ar}

3.2 System Dynamics

The pulsed GMAW process is governed by the pulsing current, which is seen in figure 3.2(a). In order to control the pulsing, the base period, which is the time interval in which the electrode is melted, is variable, thus it becomes possible to control the amount of melt detached in each pulse. If the arc length between the work piece and the electrode becomes too big or too small, as shown in figure 3.2(b), then it is likewise possible to adjust the arc length, i.e. if the arc has become too small then by decreasing the base period, thus increasing the amount of electrode consumed the arc length will become longer. This is however done on the cost of a smaller drop size and is only possible within a small distance, the main part is still controlled manually.

The pulse condition can then be described as; a pulse should occur if the arc length is below the reference and the drop size is above minimum or if the arc length is longer than the reference and the drop size is bigger than the maximum, which can be written as:

$$\text{Pulse if:} \quad (l_a < l_{ar} \wedge x_{mb} \geq x_{mb\_min}) \bigvee (l_a > l_{ar} \wedge x_{mb} \geq x_{mb\_max}) \qquad (3.1)$$

Where l_a is the arc length, l_{ar} is the arc length reference and x_{mb} is the current drop size with the indices min and max providing the bound on the desired drop size. The values of the bound can be regarded as weighting parameters for the controller design.

The weld process controller can thus be depicted as in figure 3.3, where an additional mode; *Short Circuit Handling*, is shown, which will not be discussed further in this paper. In the following the overall control strategy and dynamics of the underlying process will be presented. The model used in this paper, is derived in [Thomsen, 2004].

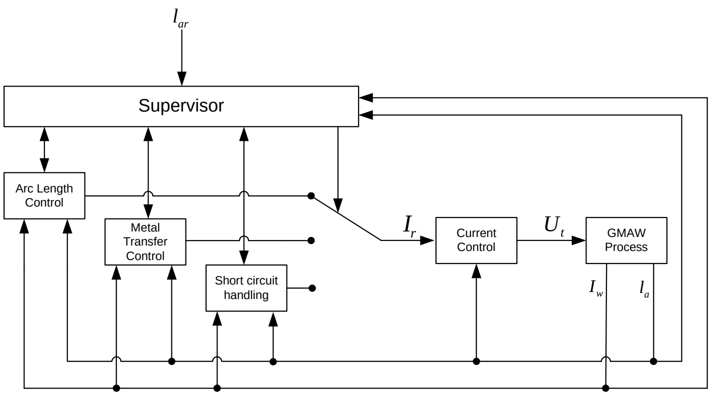


Figure 3.3: Supervisory system for the GMAW process.

Drop Dynamics

The drop dynamics, i.e. the drop growth, can be expressed as the length of melted electrode, which is a function of the welding current and the electrode length:

$$x_m = \int_{t_0}^{t_1} v_m(I, l_s)dt \tag{3.2}$$

where v_m is the velocity of melted electrode given by

$$v_m = k_1 I + k_2 I^2 l_s \tag{3.3}$$

where $l_s = 0.0115$, $k_1 = 3.6733 \cdot 10^{-4}$ and $k_2 = 6.6463 \cdot 10^{-4}$ for the considered GMAW welding application.

Arc Length Dynamics

The governing equation for the arc length dynamics can be seen in (3.4).

$$\dot{l}_a = k_1 I + k_2 I^2 \cdot (l_c - l_a) - v_e \tag{3.4}$$

where k_1 and k_2 are constants, l_c is the length from the contact tip to the workpiece, l_a is the length of the arc ($l_s = l_c - l_a$) and v_e is the velocity of the electrode.

Equation (3.5) shows the current dynamics.

$$\dot{I} = -\frac{1}{\tau_i} I + \frac{1}{\tau_i} I_r \tag{3.5}$$

where $\tau_i = 66.7\mu s$ is a constant that characterizes the dynamics. I is the welding current and I_r is the current reference.

3.3 Hybrid System Modeling

The GMAW system, as described in the previous two sections, can be formulated as the following hybrid automaton using a commonly used formalism for hybrid systems, as presented in [Henzinger, 1996a; Asarin *et al.*, 2000b], with the dynamics in each state as

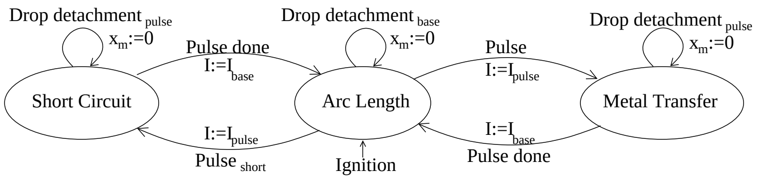


Figure 3.4: Hybrid automaton for the controlled GMAW process, divided into the three control modes: Arc length, Metal transfer and Short circuit

described in the previous section. All transitions have a label, which is used for synchronization and a reset map, which in the "Drop detachment" case is the amount of melted wire which is set to $x_m := 0$, and in the "Pulse" and "Pulse done" case it is the current, which is set to the pulse and base current respectively.

As previously mentioned, the goal of this paper is to reformulate the hybrid system into a network of timed automata in order to expand the possibilities of verifying the system properties using an automated verification tool, such as UPPAAL[Behrmann *et al.*, 2001]. This is essentially done because even though the system is exhibiting a nice and stable performance in each state, then it is possible by the right combination of switching to render the system unstable, of which a classical example can be seen in [Decarlo *et al.*, 2000].

3.3.1 Shift Register Form

In order to rewrite the dynamics of the system into shift register form, it is first put on Brunovsky normal form[Tabuada and Pappas, 2003b], for which a controllable linearized form of the system is needed.

To linearize equation (3.4) it is first rewritten into an operating point, $[\bar{I},\ \bar{l}_a]$, and deviations from this, $[\hat{I}(t),\ \hat{l}_a(t)]$, where the cubed current term is split into a varying part and an operating mode part given by I_{op}:

$$\dot{l}_a = k_1\left(\bar{I} + \hat{I}\,(t)\right) + k_2\left(I_{op}\cdot\left(\bar{I} + \hat{I}\,(t)\right)\cdot\left(l_c - \left(\bar{l}_a + \hat{l}_a\,(t)\right)\right)\right) \tag{3.6}$$

The linearization is done around the point where v_e is equal to v_m thus v_e can be omitted from this equation. After multiplying (3.6) out and neglecting the product of time varying terms, an expression of the constant terms can be found as

$$k_1\bar{I} + k_2 \cdot \left(I_{op} \cdot \left(\bar{I} \cdot \left(l_c - \bar{l}_a\right)\right)\right) \tag{3.7}$$

which, subtracted from (3.6), gives

$$\dot{l}_a = \hat{I}\left(t\right) \cdot \left(k_1 + k_2 I_{op}\bar{I} \cdot \left(l_c - \bar{l}_a\right)\right) - \hat{l}_a\left(t\right)\left(k_2 I_{op}\bar{I}\right) \tag{3.8}$$

The linearized system can now be written in state space form as

$$\begin{bmatrix} \dot{l}_a \\ \dot{I} \end{bmatrix} = \begin{bmatrix} -k_2 I_{op}\bar{I} & k_1 + k_2 I_{op}\bar{I}(l_c - l_a) \\ 0 & -\frac{1}{\tau_i} \end{bmatrix} \begin{bmatrix} l_a \\ I \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{\tau_i} \end{bmatrix} u \tag{3.9}$$

Although there are several operating points for the system only a single point will be used throughout this paper. This is done in order to simplify the presentation and is sufficient to prove the concept of the method.

The operating point used for the system is: $I_{op} = \bar{I} = 175A$, $l_c = 15mm, \bar{l}_a = 3.5mm$, which inserted into (3.9) and discretized using ZOH and a time step of $0.1s$ gives (3.10).

$$\begin{bmatrix} \dot{l}_a \\ \dot{I} \end{bmatrix} = \begin{bmatrix} 0.1306 & 0.0020 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} l_a \\ I \end{bmatrix} + \begin{bmatrix} 9.5612 \\ 1.000 \end{bmatrix} u \tag{3.10}$$

The controllability matrix for this system has full rank, which shows that the system is controllable, thus the condition for getting the system into Brunovsky normal form is satisfied.

Following the method described in [Tabuada and Pappas, 2003a] the system is transformed into the normal form shown in (3.11) through the state transformation, $x = Tz$:

$$\begin{aligned} z(t+1) &= T^{-1}ATz(t) + T^{-1}Bu(t) \Leftrightarrow \\ z(t+1) &= A_z z(t) + B_z u_z(t) \end{aligned} \tag{3.11}$$

where

$$A_z = \begin{bmatrix} 0 & 1 \\ \alpha_1 & \alpha_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0.3659 \end{bmatrix}, B_z = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, T = \begin{bmatrix} 0.392 & 0.143 \\ -2.73 & 0.001 \end{bmatrix}$$

The final step for getting the system into shift register form is to get the bottom row of the A_z matrix to be zeros, which is done through the following feedback transformation

$$u_z = u + \alpha_1 z_1 + \alpha_2 z_2 = \begin{bmatrix} F \\ 1 \end{bmatrix}^T \begin{bmatrix} z_1 \\ z_2 \\ u \end{bmatrix} \tag{3.12}$$

which gives the final system on shift register form as

$$z(t+1) \quad = \quad \tilde{A}_z z(t) + \tilde{B}_z u_z(t) \tag{3.13}$$

where

$$\tilde{A}_z = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \tilde{B}_z = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, u_z = \begin{bmatrix} F \\ 1 \end{bmatrix}^T \begin{bmatrix} z \\ u \end{bmatrix}, F = \begin{bmatrix} 0 \\ 0.366 \end{bmatrix}^T, T = \begin{bmatrix} 0.392 & 0.143 \\ -2.73 & 0 \end{bmatrix}$$

The principle of the above computations is shown in figure 3.5.

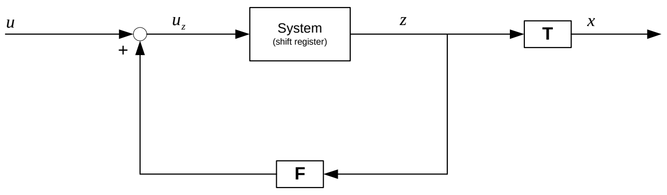


Figure 3.5: Block diagram of the shift register transformation

3.3.2 State Space Partitioning

A discrete state space $\mathbb{Z}^2$ of $\mathbb{R}^2$ is now introduced, as described in [Tabuada and Pappas, 2003b], in order to form the space in which the shift-register form system operates. The 3 domains in which the system operates is, with reference to figure 3.4, the Arc Length Control (q_1), the Metal Transfer Control (q_2) and the Short Circuit Control (q_3).

- Domains

$$\begin{aligned} Dom(q_1) &= \{(l_a, I) \in \mathbb{R}^2 | \quad 0 \le l_a \le 0.01 \quad \wedge \quad 40 \le I \le 60\} \\ Dom(q_2) &= \{(l_a, I) \in \mathbb{R}^2 | \quad 0 \le l_a \le 0.01 \quad \wedge \quad 290 \le I \le 310\} \\ Dom(q_3) &= \{(l_a, I) \in \mathbb{R}^2 | \quad 0 \le l_a \le 0.01 \quad \wedge \quad 290 \le I \le 310\} \end{aligned}$$

where l_a [m] and I [A]. The values are specified from the normal operation of a GMAW welding machine.

These domains are then transformed into shift register form by $[z_1\ z_2]^T = \mathbf{T}\,[l_a\ I]^T$ which gives the new domains

$$\begin{aligned} Dom(q_1) &= \{(z_1, z_2) \in \mathbb{R}^2 | 0 \le z_1 \le 8596 \wedge 0 \le z_2 \le 0.0643\} \\ Dom(q_2) &= \{(z_1, z_2) \in \mathbb{R}^2 | 4.18 \cdot 10^4 \le z_1 \le 4.44 \cdot 10^4 \wedge 0.307 \le z_2 \le 0.328\} \\ Dom(q_3) &= \{(z_1, z_2) \in \mathbb{R}^2 | 4.18 \cdot 10^4 \le z_1 \le 4.44 \cdot 10^4 \wedge 0.307 \le z_2 \le 0.328\} \end{aligned}$$

The relation between the new domain space and the original one can be seen from figure 3.6, where the two regions of interests are marked, one being to the left in $I \in [40 - 60]$, which is the base period, and the region to the right, $I \in [290 - 310]$, being the pulse period. As it is seen from the figure then the domains of interest are no longer square.

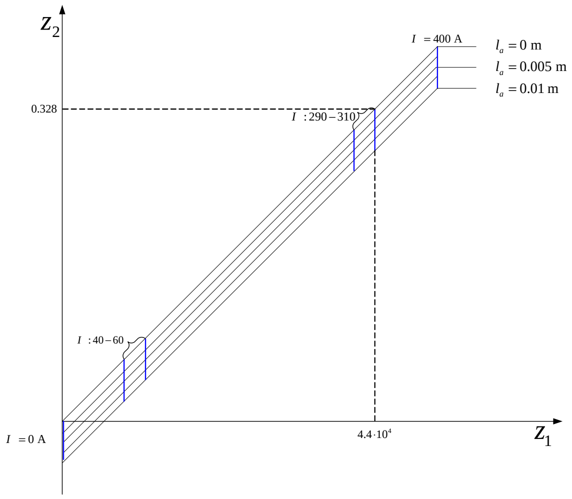


Figure 3.6: Plot of state transformation: $[l_a \; I]^T \mapsto z$

This deficiency is however remedied by relaxing the arc length constraint, which again makes the spaces of interest squares.

As described in [Tabuada and Pappas, 2003b] the partitioning needs to be equidistant, which would seem rather cumbersome for these domains due to the large ratio between z_1 and z_2, thus a scaling transformation is introduced, S_i, which transform each domains into a sufficiently equiproportional domain. In this case it is only desirable to divide the spaces into a 3 by 3 grid to prove the concept, thus a transformation that scales the 3 domains into squares are used. Following this the drop forming dynamics is modelled as a 5 state timed automaton as shown in figure 3.7. The shifting time between the drop sizes dependents only on the current. Estimated shifting values for different current intervals are shown in table 3.1. The drop formation always starts in state 1 and will propagate through the states over time. State 3 is the reference state, i.e. the state in which it is desirable to do a drop detachment.

3.3.3 Control System Imposed on $\mathbb{Z}^2$

In section 3.3.2 a new state space $\mathbb{Z}^2$ was introduced. Utilizing that the system is in shift register form, insures a well defined controlled dynamics between the partition blocks.

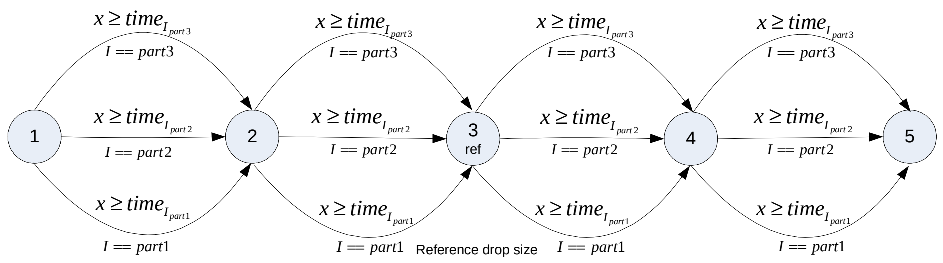

$x \geq time_{I_{part3}}$
$I == part3$
$x \geq time_{I_{part2}}$
$I == part2$
$x \geq time_{I_{part1}}$
$I == part1$
1
2
3
ref
4
5
Reference drop size

Figure 3.7: The drop dynamics timed automaton. The automaton structure for the drop dynamics is the same in each domain q₁ and q₂.

Current Partition	Current Interval [A]	Time Between Partitions [s]	Current Interval [A]	Time Between Partitions [s]
1	40.0 - 46.6	$9.9\ 10^{-3}$	290 - 296.6	$3.9\ 10^{-4}$
2	46.6 - 53.3	$8.0\ 10^{-3}$	296.6 - 303.3	$3.7\ 10^{-4}$
3	53.3 - 60.0	$6.6\ 10^{-3}$	303.3 - 310	$3.5\ 10^{-4}$

Table 3.1: Estimated time between drop size partitions in $dom(q_1)$ to the left and $dom(q_2)$ and $dom(q_3)$ to the right.

This means that under appropriate inputs the blocks will move into other partitions of equal division. To insure such *appropriate* inputs, a control law is needed.

The control law is constructed as described in [Tabuada and Pappas, 2003b] by starting with (3.11) and realizing that from a given position (z_1, z_2)=(p, q) the reachable set in one step is (z_1, z_2)=(q, r), where $r \in \mathbb{Z}$ is dependent on the input, thus it can be seen that the control law only has to ensure that z_2 will be within a control section of height δ, which is ensured by the control law:

$$u_z(k) = z_2(k) + \delta y(k)\ , \quad y \in \mathbb{Z} \tag{3.14}$$

which inserted into (3.11) results in the following system:

$$\begin{aligned} z_1(k+1) &= \quad z_2(k) \\ z_2(k+1) &= \quad z_2(k) \quad + \delta y(k) \end{aligned} \tag{3.15}$$

which is not on shift register form any longer. This is however easily remedied by introducing the control law:

$$\epsilon(k) = z_2(k) + \delta y(k) \tag{3.16}$$

Which results in the system given by

$$\begin{bmatrix} z_{\varepsilon_1}(k+1) \\ z_{\varepsilon_2}(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} z(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \varepsilon(k) \tag{3.17}$$

3.4 Example of Implementation in UPPAAL

UPPAAL is a validation- and verification tool[Behrmann *et al.*, 2001; Larsen *et al.*, 1997]. The tool consists of two main parts: A graphical user interface and a model checker engine. The idea in this paper is to model a system using timed automata, simulate it and then verify the system properties on it.

A system consists of a network of automata which are running in parallel. It is possible to step through the system, in order to check if the system behaves as intended and the system can be checked by the verifier to verify that it satisfies certain temporal specifications, such as if a certain state is reachable or if there is any deadlocks in the system. More generally speaking, the verifier can check all possible dynamical behaviors of a system[Larsen *et al.*, 1997].

3.4.1 The Controlled GMAW Process in UPPAAL

An overview of the implemented system is shown in figure 3.8, where the supervisor automaton controls the underlying automata, the drop dynamics automata and the GMAW dynamics automata. The supervisor decides by its two transitions which control mode the GMAW process should be in by a parallel composition with a shared label space in the sense of Milner[Milner, 1989], which is illustrated in figure 3.8. As pointed out in the previous section then the GMAW dynamics is only partitioned into 9 parts. This leads to a timed automaton for the arc length dynamics as shown in the middle of figure 3.8 with some of the possible transitions displayed. The automaton consists of 9 states, where each state represents a partition of the state space. The transitions between the states are decided by the shift register form, which is derived in the previous section.

In order to include disturbances into the model a disturbance automaton is included as shown in figure 3.9(a). It is designed to give a disturbance in the arc length in the base period. If the disturbance automation enables a disturbance (increase/decrease the arc length), it will affect the GMAW dynamics automaton as shown in figure 3.9(b).

3.4.2 Model Checking

It is possible in UPPAAL to use the model checker to get answers on specific questions, e.g. to check if there are deadlocks in the system. The deadlock check can be seen as a basic check of the systems behavior. By checking reachability and liveness properties the performance of a supervisor or controller can be analyzed. In UPPAAL the query language used is a simplified version of Computation Tree Logic (CTL)[Huth and Ryan, 2004].

In the following specific questions regarding the GMAW process will be discussed.

Do deadlocks exists in the system?

Query:

```
A[] not deadlock
```

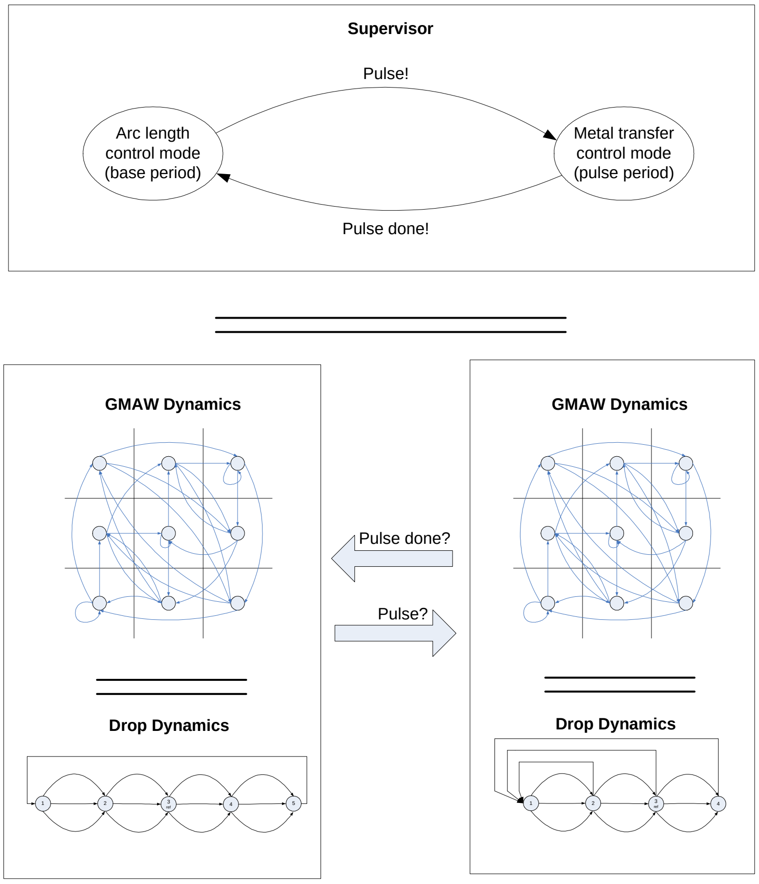
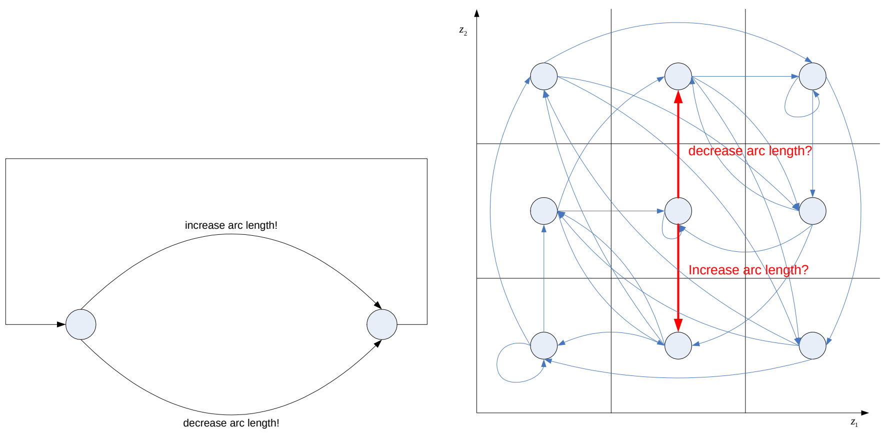


Figure 3.8: The figure illustrates that it is possible to be in two different control modes; arc length control mode and metal transfer control mode. The supervisor controls which of the two modes to be in. In each control mode the processes are running in parallel.



(a) Disturbance automaton (b) Effect of a disturbance

Figure 3.9: (a) The disturbance automaton. (b) The two thick arrows shows the effect of a disturbance from the disturbance automaton to the GMAW dynamics.

Numerous factors can result in a deadlock in the system; A supervisor design flaw, faulty implementation etc.

Answer:
The property is satisfied.

Do the supervisor continuously cycle between the base period and the pulse period?

Query:
```
Supervisor.Arc_length --> Supervisor.Metal_transfer
Supervisor.Metal_transfer --> Supervisor.Arc_length
```

To guarantee the basic operation of the supervisor, a continuously cycle between the base period and the pulse period should take place. The first expression checks if the path between the states *Supervisor.Arc_length* and *Supervisor.Metal_transfer* will eventually be taken. The second expression checks if the path back from the state *Supervisor.Metal_transfer* to the state *Supervisor.Arc_length* will eventually be taken.

Answer:
The question is satisfied

Is the duration of the pulse period as specified?

Query:
```
A[] Supervisor.Metal_transfer imply x<=600
```

The duration in the pulse period is set to 600 clock cycles. This question checks if it is possible for the supervisor to jump from metal transfer control to arc length control before the specified time.

Answer:
The question is satisfied

The first question checks if there is some states from which the system cannot switch away from, which it is found that there are not. Secondly the liveliness of the supervisor is tested. This test can be seen as a check of the supervisor shown in figure 3.3. It is further interesting to verify if the system is staying too long in the different states, which is tested in the third query, where the time spend in the metal transfer state is tested. Similarly to the third query it could be tested if the supervisor is switching too fast between the different states, which will reveal if there is a possibility for Zeno behavior in the system.

3.5 Discussion

The objective of this paper was to show that it is possible to apply the theories developed in [Tabuada and Pappas, 2003b] to a given process, in this case the Pulsed GMAW process.

As seen from section 3.3 it is possible to formulate the GMAW welding process as a network of timed automata, which can be directly implemented in the simulation and verification tool, UPPAAL, thus giving the possibility of posing such questions as; is state A always reachable from state B or is it possible to end up in a deadlock - Questions, which is impossible to answer with classical control theory.

3.6 Acknowledgements

We would like to thank Jesper Sandberg Thomsen, Christian Krogh Nielsen and Jakob Brøchner Vilhelmsen for their insightful discussions and help with the first simulations.

Chapter 4

A Piecewise Affine Hybrid Systems Approach to Fault Tolerant Satellite Formation Control

In this paper a procedure for modelling satellite formations including failure dynamics as a piecewise-affine hybrid system is shown. The formulation enables recently developed methods and tools for control and analysis of piecewise-affine systems to be applied leading to synthesis of fault tolerant controllers and analysis of the system behaviour given possible faults. The method is illustrated using a simple example involving two satellites trying to reach a specific formation despite of actuator faults occurring.

4.1 Introduction

Several future Earth and space science missions [Léger, 2007; Baker *et al.*, 2007] involve operation of multiple coordinated spacecrafts.This has resulted in the development of a number of control strategies[Scharf *et al.*, 2004; 2003] for satellite formations, but only few consider fault tolerance on the formation level. Reducing or eliminating the effects of faults is an important aspect of spacecraft design as it increases the amount of time a satellite formation can perform scientific observation by extending the mission lifetime and keeping the formation in science mode for a higher fraction of the time. As described in [Tsiotras and Doumtchenko, 2000] several catastrophic failures of spacecraft in orbit have been due to malfunctions in the control subsystem and it is thus of particular importance to develop fault tolerant control strategies.

Current fault control design focuses mostly on individual satellites, but when introducing such satellites as part of a formation new opportunities and restrictions arise. In [Meskin and Khorasani, 2006] it is shown that an actuator fault on a satellite potentially can destabilise a whole formation. Due to the distributed nature of a satellite formation it possesses an inherent redundancy. A fault occurring on one satellite could be offset by actions taken by other satellites in the formation e.g. an actuator fault on one satellite can be compensated for by control of the other satellites in the formation, which is to maintain relative positions. This interaction and the possible measures that can be taken to remedy faults is therefore important to describe and analyse in order to develop automated/autonomous fault handling strategies on a formation[Mueller and Thomas, 2005].

Previous efforts in this area has been focusing on e.g. passive fault tolerance via robust or adaptive control design method [Thanapalan *et al.*, 2006; Garcia-Sanz *et al.*, 2007]. Similar problems are found in control of formations of mobile robots and unmanned aerial vehicles. [Desai *et al.*, 1999] uses the popular leader-follow control structure [Lafferriere *et al.*, 2005; Scharf *et al.*, 2004; 2003] to design a reconfiguration algorithm that can be used to reconfigure a formation after a fault has occurred. In [Boskovic and Mehra, 2003] the same architecture and a controller for formations capable of handling actuators faults is designed while failures in communications systems is investigated in [Berger *et al.*, 2003].

This paper focuses on deriving satellite formation models described as piecewise affine hybrid systems (PAHS). A PAHS formulation allows continuous time satellite formation dynamics to be extended with discrete events representing faults in formation components. Given that the formation and potential faults can be modelled as a PAHS, then modern control synthesis methods [Habets *et al.*, 2006; Wisniewski and Larsen, 2008] support automatic generation of fault tolerant control strategies. Specifically [Wisniewski and Larsen, 2008] demonstrates how controllers can be synthesised by abstracting the control problem to a combinatorial objects or a state machine. The aim of this paper is therefore to capture the behavioral aspects and fault modes of a satellite formation in a PAHS to be able to leverage the research within PHAS.

In Section 4.2 a deep space satellite formation model is presented using discrete events initiate state transitions to model faults resulting in a hybrid system. By simplifying the model and partitioning the state space into a simplicial set a piecewise affine hybrid system is constructed. An example of this modelling approach is illustrated in Section 4.3 and the results are summarised and discussed in Section 4.4.

4.2 Methodology

The procedure described in this section aims at providing a method for modelling satellite formations with faults. The model is constructed in such a way as to be able to synthesise a fault tolerant controller. The basic strategy is one of abstraction from a hybrid system

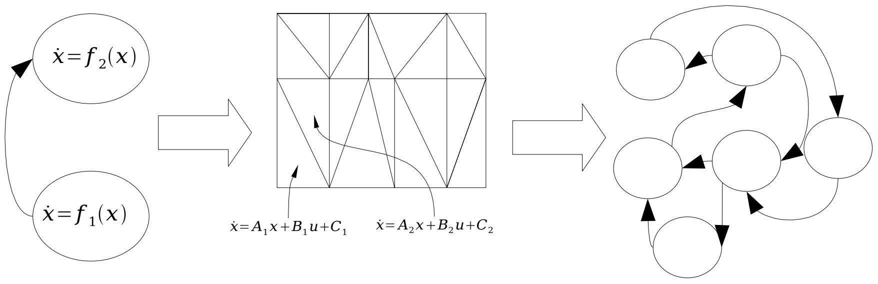


Figure 4.1: Abstraction of the formation model from a hybrid system to a discrete automaton.

to a discrete description. The spacecraft formation is modeled as a generic hybrid system which is translated into a PAHS, for which discrete abstraction and analysis methods exist [Habets *et al.*, 2006]. Specifically a discrete abstraction such as an automaton, for which numerous synthesis and analysis methods exist, can be extracted from the PAHS description [Wisniewski and Larsen, 2008; Larsen *et al.*, 1997]. The procedure is illustrated in fig. 4.1 showing how a hybrid system is translated into a PHAS which can be abstracted into an automaton.

4.2.1 Spacecraft Formation Model

The model presented in this section describes an deep space (e.g.. heliocentric or Lagrange point) satellite formation with n satellites. The model does not include the spacecraft attitude as it for most formations can be controlled independently of the position.

The relative position dynamics of two spacecrafts in a deep space formation can be described as a simple double integrator with the thrust force as input,

$$\dot{\begin{bmatrix} r \\ v \end{bmatrix}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} r \\ v \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ m_1^{-1} & m_2^{-1} \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} , \tag{4.1}$$

where r is the relative position of the spacecrafts, $v = \dot{r}$ is the relative velocity, m_i is the mass of the i'th spacecraft and T_i is the thrust.

As only the position dynamics are considered the standard 12 thruster setup can be reduced to 6 thrusters, two for each of the 3 spatial dimensions [Smith and Hadaegh, 2005]. Furthermore each thruster-pair can be abstracted into one thruster capable of providing both negative and positive thrust.

The result is a dynamic model of the n-satellite formation, with the relative position, $r_{i,j}$, and velocity, $v_{i,j}$, for all satellite pairs, (i, j) , $i \neq j$ as the state[1],

[1]Only one of the pairings (i, j) (j, i) are used i.e. $r_{i,j}$ and $r_{j,i}$ are not both in the state vector.

$$
\frac{\mathrm{d}}{\mathrm{dt}}\underbrace{\begin{bmatrix} r_{1,2} \\ r_{1,3} \\ \vdots \\ r_{n-1,n} \\ v_{1,2} \\ v_{1,3} \\ \vdots \\ v_{n-1,n} \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} 0_{3k\times 3k} & I_{3k\times 3k} \\ 0_{3k\times 3k} & 0_{3k\times 3k} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} r_{1,2} \\ r_{1,3} \\ \vdots \\ r_{n-1,n} \\ v_{1,2} \\ v_{1,3} \\ \vdots \\ v_{n-1,n} \end{bmatrix}}_{x} +
$$

$$
\underbrace{\begin{bmatrix} 0 & \dots & \dots & \dots & 0 \\ M_1 & -M_2 & \dots & \ddots & \vdots \\ M_1 & 0 & -M_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & M_{(n-2)} & \dots & -M_n \\ 0 & \dots & \dots & M_{(n-1)} & -M_n \end{bmatrix}}_{B} \underbrace{\begin{bmatrix} T_{1,x} \\ T_{1,y} \\ T_{1,z} \\ \vdots \\ T_{n,x} \\ T_{n,y} \\ T_{n,z} \end{bmatrix}}_{u}, \tag{4.2}
$$

where $k = \frac{n(n-1)}{2}$ is the number of unique spacecraft pairings, $M_i = Im_i^{-1}$[2] is a matrix describing the mass of each spacecraft and $T_{i,l}$ is the input thrust of spacecraft i in direction l.

The nominal formation model is an ideal model as it does not take disturbances such as gravity or radiation pressure into account, as these disturbances are very small in a deep space formation.

4.2.2 Fault Modelling

When modelling faults affecting the control of a satellite formation the focus is usually on actuator, sensor and control computer faults. For satellite formations the communication structure is also a possible failure point which further complicates the situation. In order to truly model the faults which can occur on a satellite formation a full fault modelling and effect analysis (FMEA) is to be performed. The focus here will be on simple actuator faults as they are easily modelled and have a very visible effect on the formation dynamics.

The actuator faults considered in this paper are loss of a thruster, fuel leak and stuck thruster.

Loosing a thruster can be modeled by changing the input matrix, B, of the system. In case thrust in the l'th direction is lost on the i'th spacecraft it is modelled by multiplying

[2] I being the identity matrix

the input matrix with a unit matrix where the corresponding row is zeroed, e.g.

$$B_{fault} = B \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} , \tag{4.3}$$

In case the thrusters in the 2nd thruster pair on the first satellite $T_{1,y}$ fails.

A fuel leak can be described as an additive constant thrust, which is modelled by adding a constant vector, C, to Eq. 4.2:

$$\dot{x} = Ax + Bu + C \tag{4.4}$$

$$C = \begin{bmatrix} 0_{3k\times 1} \\ F_{leak,1,2} \\ \vdots \\ F_{leak,n-1,n} \end{bmatrix} \tag{4.5}$$

Here $F_{leak,i,j}$ describes the thrust force produced by the leak affecting the (i, j)'th spacecraft pair.

Combining the two can be used to model a stuck thruster.

4.2.3 Hybrid System Formulation

By combining the formation model with the fault model a hybrid system can be constructed. Occurrence of faults are equated with discrete (uncontrollable) input events and the continuous mode of a discrete state describes the corresponding dynamics of the formation under failure.

Throughout the literature there are many different ways of describing a hybrid system. One of the standard ways of doing so is in the sense of [Henzinger, 1996a] where a hybrid system is described as a graph with a dynamical system defined on each vertex. Presented here is specifically the open hybrid automaton [Lygeros and et. al., 1999].

Definition 3 (Open Hybrid Automata)**.** *An open hybrid automaton, H, is a collection*

$$H = (Q, X, V, Y, Init, f, h, Inv, E, G, R)$$

where

- *Q is a finite collection of discrete state variables.*
- *X is a finite collection of continuous state variables.*
- *V is a finite collection of input variables. Assuming $V = V_D \cup V_C$, where V_D contains discrete and V_C continuous variables.*

- *Y is a finite collection of output variables. Assuming $Y = Y_D \cup Y_C$, where Y_D contains discrete and Y_C continuous variables.*
- init $\subset Q \times X$ *is a set of initial states.*
- *$f : Q \times X \times V \to \mathbb{R}^n$ is a vector field describing the evolution of the continuous state variables;*
- inv*: $Q \to 2^{X \times V}$ assigns to each $q \in Q$ an invariant set.*
- *$h : Q \times X \to Y$ is an output map.*
- *$E \subset Q \times Q$ is a collection of discrete transitions.*
- *$G : E \to 2^{X \times V}$ assigns to each $e = (q, q') \in E$ a guard.*
- *$R : E \times X \times V \to 2^X$ assigns to each $e = (q, q') \in E$ and $v \in V$ a reset relation.*

With $f(q, x, v)$ and $h(q, x)$ being globally Lipschitz continuous in x and $f(q, x, v)$ being continuous in v.

Remark 1. *In order to model external events forcing a state transition e.g. a fault, a simple modelling trick is used. Each external event is associated with a discrete variable, $v_e \in V_D$, and for each state pair, q, q', where the event should trigger the transition $e = (q, q')$ a guard is assigned of the form $g(e) : u_e = 1$ with $inv(q) : u_e = 0$ and $inv(q') : u_e = 1$.*

The satellite formation model including fault modelling can thus be represented as an open hybrid automata as follows:

System 1. *Hybrid system satellite formation model*

- *Each state, $q \in Q$ represents a fault scenario $i = 0 \ldots n$ given n possible faults. q_i designates the fault state where $i = 0$ signifies the nominal case. Considering scenarios with several simultaneous faults occurring requires more states e.g.. $q_{i,j}$ could denote fault i and j occurring at the same time.*
- *X describes the relative positions and velocities used as the state variable in Eq. 4.2.*
- *V_C describes the input thrust as shown in Eq. 4.2 where V_D is used to denote the occurrence of faults as per remark 1.*
- *As the satellite formation is considered fully observable for the purposes of this paper the set of continuous output the set of continuous state variables $Y_C = X$,and the set of discrete outputs is equal to the set of discrete states $Y_D = Q$.*
- init *is the initial configuration of the formation which can vary according to the formation deployment, but will typically contain the fault free state q_0.*

- *f Is the flow given by the differential equation Eq. 4.2 and the fault model Eq. 4.3,4.4 of the current failure mode. Notice that V_D is not used as input to the flow in this model.*
- *The invariant,* inv, *for all states is the entire domain of the continuous variables $\mathbb{R}^n$ with the discrete part of the invariant V_D constructed as per remark 1.*
- *For the automaton to be fully observable $y_C = h_c(x, q) = x$ and $y_D = h_D(x, q) = q$.*
- *E describes the transitions occurring at the event of a fault.*
- *G is the guard set which is constructed per remark 1 to enable fault events.*
- *The reset maps, R, are all identity maps as no jumps in the continuous state happens when a failure occurs.*

As the Open Hybrid Automata (OHA) as defined by [Lygeros and et. al., 1999] is a very general and syntactic description of a hybrid system, it is hard to perform any computations on a system described by it. It is therefore desirable to transform the formation model from an OHA to another description which yields better to computation. In this case the piecewise-affine hybrid system has been chosen as control synthesis and analysis of such systems is an active research area.

4.2.4 Piecewise Affine Hybrid System

PAHS are a subset of hybrid systems which are restricted to affine dynamics with domains, guard sets etc. defined on polytopes. A polytope can be defined as the convex hull of a finite number of points. Specifically a polytope of dimension n is called a simplex if it is the convex hull of $n + 1$ points[3] e.g. a triangle in $\mathbb{R}^2$ or a tetrahedron in $\mathbb{R}^3$.

Following the method in [Habets *et al.*, 2006] with the addition of discrete input events a PAHS can be defined as:

Definition 4 (Piecewise-Affine Hybrid System)**.** *A piecewise-affine hybrid system, $\mathcal{H}$, is a collection*

$$\begin{aligned}\mathcal{H} \quad = \quad & (\mathcal{Q}, \mathcal{E}, \mathcal{T}, \mathcal{W}, \mathcal{U}, \{\mathcal{X}_q, \mathcal{A}_q \mid q \in \mathcal{Q}\}, \{\mathcal{R}_q(e), \mathcal{G}_q(e) \mid (q, e) \in \mathrm{dom}(\mathcal{T})\}, \\ & \{\mathcal{I}(w)_q \mid q \in \mathcal{Q}, w \in \mathcal{W}\})\end{aligned}$$

where

- *$\mathcal{Q}$ is the set of discrete states $q \in \mathcal{Q}$.*
- *$\mathcal{E}$ is the set of edges connecting states $e \in \mathcal{E}$.*

[3]The points can not coincide

- *$\mathcal{T}$ is a transition map (partial), $\mathcal{T} : \mathcal{Q} \times \mathcal{E} \rightarrow \mathcal{Q}$, relating states and edges.*

- *$\mathcal{W}$ is the set of input events $w \in \mathcal{W}$.*

- *$\mathcal{U}_q$ is a polytope describing the continuous input $u_q \in \mathcal{U}_q$.*

- *$\mathcal{X}_q$ is a politopic set of continuous state variables $x_q \in \mathcal{X}_q$, defining the invariant set of the state q. Note that the state variables in different discrete states can differ.*

- *$\mathcal{A}_q$ denotes an affine system, $\dot{x}_q = A_q x_q + B_q u_q + C_q$, governing the evolution of the continues variables in state q.*

- *$\mathcal{G}_q(e)$ The guard sets, $G_q(e) \subset \partial X_q$, is the boundary of the domain, X_q, for the affine system A_q.*

- *$\mathcal{R}_q(e)$ Reset maps are defined for each transition, e, as a function from guard sets to the domain of the state $R_q(e) : \mathcal{X}_q \rightarrow \mathcal{X}_{f(q,e)}$ for any $(q, e) \in \mathrm{dom}(\mathcal{T})$.*

- *$\mathcal{I}_q(w)$ The input map causes transitions, $e \in \mathcal{E}$, to be taken based on the input events $w \in \mathcal{W}$, $\mathcal{I}_q(w) : \mathcal{W} \times \mathcal{Q} \rightarrow \mathcal{E}$, according to $\mathcal{T}$.*

If each of the continuous state sets, $\mathcal{X}_q$, are simplices then $\mathcal{H}$ is a piecewise-affine hybrid system on simplices for which there exist control synthesis methods [Habets *et al.*, 2006; Wisniewski and Larsen, 2008].

Focusing on the simplicial version of a PAHS it is evident that the domains of the hybrid system formation model 1 are not compatible with the requirement that the state sets, X_q, are simplices as all of the invariants in H are $\mathbb{R}^n$. The solution lies in selecting a bounded region of $\mathbb{R}^n$ as the state space and discretising it into a finite number of simplices as shown in fig. 4.2. The optimal division of a space into simplices is still an open question, but a common method is to use a finer division near areas of interest in the state space e.g. reference point such as desired formations.

The end result is that each discrete state in H is split into several discrete states with the same affine dynamics in $\mathcal{H}$. However the events caused by failures changes the affine dynamics without changing the continuous state variable, $R_q(e) = id$, and this is easily visualised as a transition between two state spaces, see fig. 4.3.

It is worth to note that even if the underlying dynamics are non-linear this approach can be used as for each simplex the non-linear system can be linearised [Sontag, 1981] and yield a piecewise affine approximation to the nonlinear system. The end result is a controller strategy very similar to gain scheduling.

By converting the hybrid system 1 into piecewise-affine hybrid system the satellite formation can be described as:

System 2. *PAHS satellite formation model*

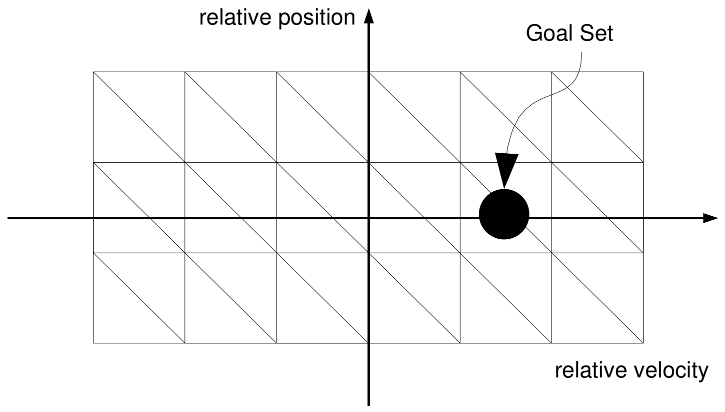


Figure 4.2: A 2 dimensional state space divided into simplices.

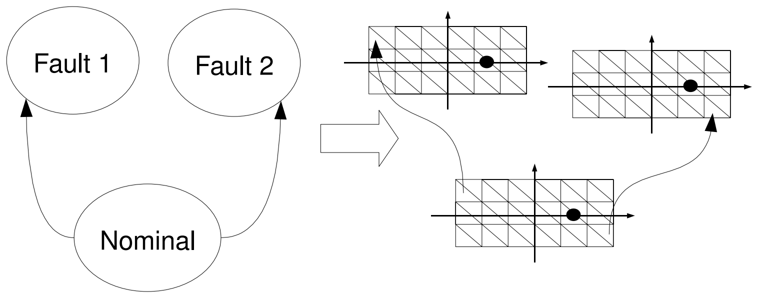


Figure 4.3: An external event causes the dynamics to be changed. Left the hybrid system H. Right the PAHS $\mathcal{H}$

- *For each state, $q \in Q$, representing a failure mode, i, the domain of the continuous state, $x \in X$, is partitioned into simplexes represented by a set of discrete states $\mathcal{Q}_i \subset \mathcal{Q}$.*

- *The set of discrete states is the union of the discrete states generated by partitioning the domain of each failure mode $q \in Q$ into simplices.*

$$\mathcal{Q} = \bigcup_{i=0}^{|Q|-1} \mathcal{Q}_i$$

- *For each pair of adjacent simplices represented by the states $(q, q') \in \mathcal{Q}$ a transition $e \in \mathcal{E}$ exist with the corresponding transition relation $\mathcal{T}(e, q) = q'$. Furthermore each transition $e = (q, q') \in E$ from the system 1 corresponding to a fault, i, generates a set of transitions, $\mathcal{E}_i \subset \mathcal{E}$; one for each discrete state in $\mathcal{Q}_i$. The transition relation, $\mathcal{T}(e_s, q_s) = q'_s$, maps each transition from state q_s representing simplex, s, to the state representing the corresponding simplex, q'_s, in the failure mode.*

- *The set of discrete input events, $\mathcal{W}$, is obtained from the one-to-one mapping, $L(v_d) : V_D \to \mathcal{W}$ such that each discrete variable in the hybrid system, H, corresponds to a discrete event in $\mathcal{H}$.*

- *The input map, $\mathcal{I}_q(w)$, is defined s.t. a fault, i, generating input event, w_i, maps to the intersection between the set of transitions starting in $q \in \mathcal{Q}$, denoted $\mathcal{E}_q$,[4] and the set of transitions associated with the i'th failure, $\mathcal{E}_i$.*

$$\mathcal{I}_q(w_i) = \mathcal{E}_q \cap \mathcal{E}_i$$

- *The set of continuous input variables V_C can be defined such that $v_{i,j} \in V_C$ is limited by the maximal thruster force $-T_{i,j,max} \leq v_{i,j} \leq T_{i,j,max}$. Constructing $\mathcal{U}$ as the Cartesian product of the input variables $\mathcal{U} = v_{1,1} \times v_{1,2} \times \cdots \times v_{n,3}$ yields a hyperrectangle thus fulfilling the condition that $\mathcal{U}$ must be a polytope.*

- *As the discrete states, $q \in \mathcal{Q}$, are defined with respect to simplicial a decomposition of the state space, X, it naturally follows that the set of the continuous state variables, $\mathcal{X}_q$, for each discrete state is exactly the simplex for which the discrete state is defined.*

- *As the domain of each failure mode, $q \in Q$, is divided into simplices, then for the set of states, $\mathcal{Q}_i \subset \mathcal{Q}$, the affine dynamics are equal to the dynamics of the failure mode i. $\mathcal{A}_{\mathcal{Q}_i} = \{A_i, B_i, C_i\}$.*

- *The guard sets are defined per definition 4 to lie on the boundaries of the domains of each state, $q \in \mathcal{Q}$, i.e. on the facets of the simplices.*

- *The reset maps from the open hybrid automata description of the formation carries over and as the evolution of the continuous state, X, in each failure mode, $q \in Q$, includes no discontinues jumps the reset maps for the PAHS model are all identity maps.*

In order to obtain a discrete abstraction of the formation model each of the transitions, $e \in \mathcal{E}$, must be evaluated to determine whether they are reachable and controllable. For the transitions caused by traversing a simplicial facet the method as described in [Wisniewski and Larsen, 2008; Habets *et al.*, 2006] determines whether a facet can be blocked

[4] $\mathcal{E}_q$ is the set of states for which $\mathcal{T}(e, q) \neq \emptyset, e \in \mathcal{E}$

(controllable) or not (uncontrollable), and for the transitions caused by failure events they are all uncontrollable. Therefore an abstraction describing the possible traversal of the simplices can be represented as e.g. a discrete automaton or as a simplicial set as shown in the example in the next section.

Furthermore a number of tools exist which can aid in system analysis and control synthesis. Some examples are; UppAal[Larsen *et al.*, 1997] which can synthesise controllers and check properties of the discrete abstraction, and Ariadne and PHAver [Balluchi *et al.*, 2006; Frehse, 2005] which computes reachable and safe sets for piecewise-affine hybrid systems.

4.3 Example

In this section an example of how a two spacecraft formation can be controlled using the aforementioned formalisms will be illustrated. The goal in this example is bringing the two spacecrafts together, and the problem therefore reduced to the rendezvous problem, but with the added complication of possible faults occurring.

The two-satellite formation problem, which was stated in Eq. 4.1, is sufficiently simple to be grasped in lower dimensions and contains enough complexity to show the strength of the proposed methodology, which easily scales to a greater number of satellites.

Since the system state space is four dimensional it needs to be divided into a simplicial set containing a number of 4-dimensional simplices. However, as can be seen from Eq. 4.1, the system is naturally decoupled in the two spatial dimensions. Thus the system can, without loss of generality be represented as a parallel composition of two two-dimensional simplicial sets. One of these spatial dimensions is shown in fig. 4.4. Here the relative position error, in one of the dimensions, is displayed along the horizontal axes and the relative velocity between the two satellites along the vertical axes. The allowable regions are the white simplices, in which the satellite formation is allowed to move freely and the shaded areas are the goal sets, which is the control objective. There is also the possibility of adding dark shaded regions comprising avoid sets which signify regions of the state space which is forbidden to enter e.g. where collisions will occur or communication/sensors will cease working.

Now, for each simplex the possibilities for blocking a facet can be found, as described in [Wisniewski and Larsen, 2008]. An example of this for an initial position between 100 and 500 meters from the goal state is shown in fig. 4.5. The initial state is here marked by a black ball. For each simplex traversed along the path towards the goal state the facets, which it is possible to leave through is marked by an arrow indicated the direction of leaving. If it is possible to block that state the tail of the arrow is additionally marked by a black dot corresponding to controllable and uncontrollable transitions in a digraph description.

As this formalism does not specify what is to happen in the goal state, but just that it should be reached, the transition taken entering it might just as well be a transition to

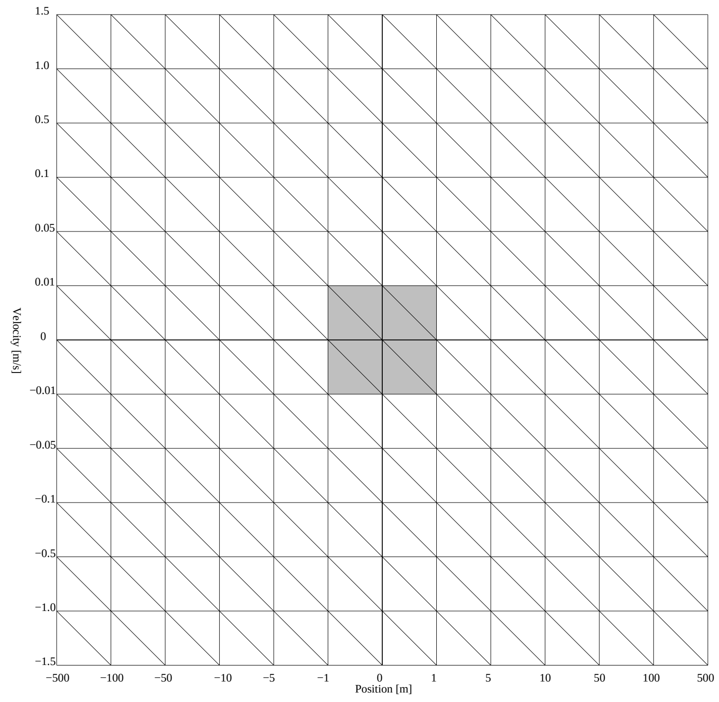


Figure 4.4: Example division of the two-satellite formation problem along one spatial dimension.

another PAHS, just with a finer state division or alternatively if all facets can be blocked and a fix point created a reference in the state space can be reached [Habets *et al.*, 2006].

To illustrate how faults affects the system fig. 4.6 shows a superposition of the state spaces of nominal and a fault mode, with identical simplicial divisions. The solid line shows the evolution of the continuous states in the fault free mode while the dashed line shows the evolution of the state after the fault occurs. As can be seen the flows differ from the point of failure and this is a consequence of the fault causing facets to change from being blockable (controllable) to unblockable (uncontrollable), shown as white balls.

4.4 Discussion and Conclusion

In this paper a model of a satellite formation including thruster faults has been presented and a conversion to a piecewise-affine hybrid system has been shown, leading to the possibility of leveraging recent methods developed to analyse and synthesise controllers for PAHS. The control synthesis method has been outlined and an illustrative example of

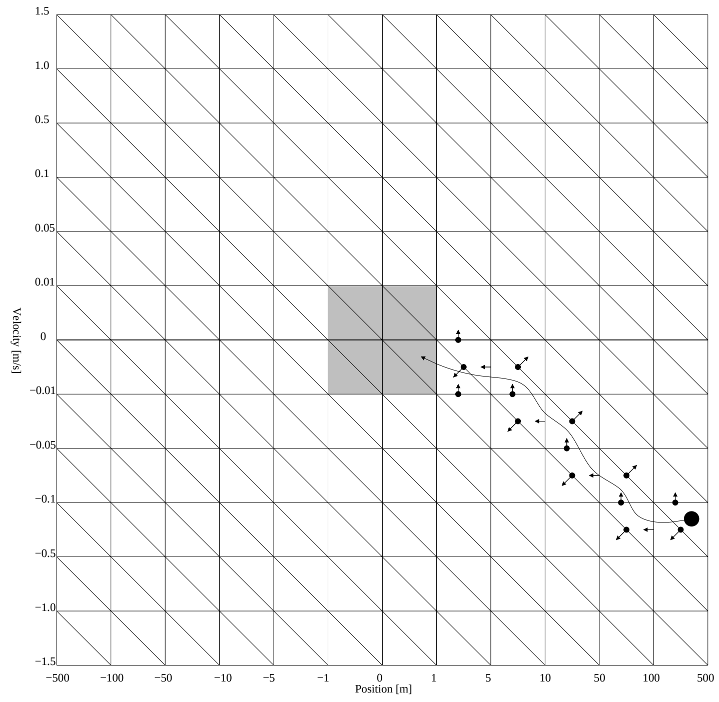


Figure 4.5: Example of how the goal state could be reached from the initial state.

a two-satellite formation has been used to clarify the concept.

There are still a number of open questions and place for improvements. Specifically adapting the method to handle distributed as opposed to the centralised control as shown in this paper and also the ability to include some measure of (fuel) optimality into the approach.

Furthermore the optimal partitioning into simplices is an open question. Especially considering that the number of simplices required to maintain a fixed grid resolution increases exponentially with the number of continouos dimensions.

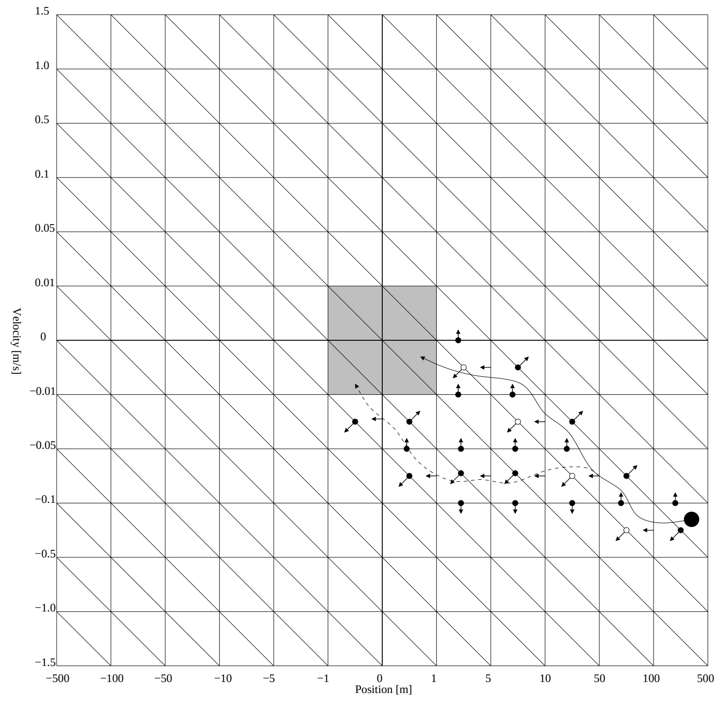


Figure 4.6: Example of how a fault changes the controlability of transitions and thereby the flow of the continuous states.

Chapter 5

Combinatorial Control Systems

The paper introduces the concept of a combinatorial control system. It is a discrete abstraction of a hybrid- and thus in particular a continuous control system. We associate to it notions of a flow map, a Lyapunov function and feedback. The state space is partitioned into polyhedral sets and the control action forces a shift from a polyhedron to its neighbor. Locally in each polyhedral set the system is approximated by an affine model. The concept is particularly useful for combined guidance and control of e.g. autonomous robots and satellites; as an example in the paper illustrates.

5.1 Introduction

Automatic control synthesis for large scale systems is a central capability for creation of next generations of autonomous systems, such as automatic robotic manufacturing systems, and mobile robotics operating in uncharted environments.

Previously, it has been shown that feedback control can be automatically generated for a class of nonlinear systems. In [Pedersen *et al.*, 2002] evolutionary algorithms and in [Prajna *et al.*, 2004] Lyapunov functions were used for automatic control synthesis. These methods are not devised to handle switches between operation modes characterized by different dynamic. Nevertheless, mode change is essential for autonomous dynamical systems. On the other hand, considerable progress has recently been made in the direction of discrete abstractions of continuous and hybrid control systems. In e.g. [Tabuada and Pappas, 2003b] the state space was partitioned by higher dimensional cubes, and controllable system has been shown to be bi-similar to a discrete register.

In the current paper a unified modeling framework is developed. It takes into account both possibility of switches between the systems dynamics and non-linearities in the system model. An overview of the proposed framework is sketched in Fig. 5.1. At first, three components are given: the system in question, admissible control and the control objectives. The state space of the system is partitioned into polyhedral sets, here simplices. In each simplex the system dynamics is approximated by an affine model. The totality of simplices is called in the diagram a simplicial complex. Admissible control is used to assign to each simplex the set of reachable simplices in its direct neighborhood - the control determines the future simplices. This control action will be called a control combinatorial vector field (CCVF). Motivated by the continuous counterpart a control objective is expressed by a function whose minimum is at the goal simplex and which is decreasing along the system trajectories - this is a combinatorial Lyapunov function. A Combinatorial Control System designates in the figure the selection of a discrete control action satisfying the control objectives.

The paper is organized as follows. Sect. 5.2.1 gives a brief introduction to simplicial complexes. It is shown in Sect. 5.2.2 how to steer the system such that each trajectory exits a simplex through a desired face. Combinatroial vector fields are introduced in Sect. 5.2.3. It constitutes a discrete abstraction of a control system. The concept of a combinatorial Lyapunov function is brought in in Sect. 5.2.4. It is used in this paper to assert the control objectives. Finally, Sect. 5.3 provides a simple example, which illustrates how the methodology developed in this work can be used for automatic control synthesis.

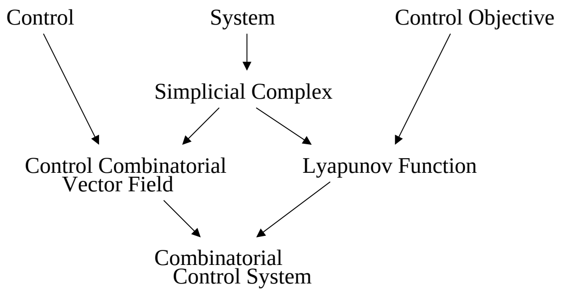


Figure 5.1: Overview of the combinatorial control system framework.

5.2 Methodology

The different components of Fig. 5.1 will be elaborated in this section. The main attention will be given to how a dynamic system can be transformed into a simplicial complex and how the possible control actions can be embedded into a CCVF. The two remaining parts, the Lyapunov function and finally combinatorial vector field will merely be sketched and the relevant references will be provided at the end. For a thorough handling of the theoretical background the reader is referred to [Wisniewski and Larsen, 2008; Larsen and Wisniewski, 2008a] and the references therein.

5.2.1 Simplicial Complex

A n-dimensional simplex, i.e. an n-simplex, is the geometric space spanned by $n + 1$ independent points, hereafter called vertices, i.e. in 2-dimensions a simplex is a triangle, in 3-dimension a tetrahedron etc. The sides of a simplex are called facets, and each facet of a simplex is in itself a simplex. All such facets and facets of facets are called faces.

A simplicial complex K in $\mathbb{R}^n$ is a collection of simplices in $\mathbb{R}^n$ such that [Munkres, 1984]: Every face of a simplex of K is in K and the intersection of any two simplexes of K is a face of each of them.

Since any polytope can be constructed by the combination of a finite number of simplices the methods describe in the following holds for any polytope. For a more thorough introduciton to simplicial complexes the reader is refered to [Grünbaum, 2003; May, 1992; Zomorodian, 2005].

Transforming a dynamical system into a simplicial complex can be divided into the following two steps: triangulation and linearization.

Partitioning

The first step in obtaning a discrete abstraction of a system is to partition the state space into a simplicial complex. So far, there does not exist one optimal way of doing this without extensive knowledge of the underlying dynamical system, thus we suppose that there exists a partition. The space on which robotic systems are modelled can be described by a bounded subset of $\mathbb{R}^n \times \mathrm{T}^m \times \mathrm{S}^o$, $n, m, o \in \mathbb{N}$, which has the property, that it can be represented as a simplicial complex with a finite number of simplices, thus making this abstraction into a simplicial complex valid for such systems.

Linearization

To represent the dynamical system as a simplicial complex a dynamic model approximating the original systems dynamics locally is assigned to each simplex. In this work an affine model is used to represent the dynamics in each simplex, because it gives a good trade-off between being able to represent an arbitrary system locally and at the same time being able to handle it mathematically [Sontag, 1981; 1982]. An affine system takes the following form:

$$\dot{x} = Ax + Bu + a,$$

with $x \in \mathbb{R}^n$ being the state-space, $A \in \mathbb{R}^{n\times n}$ the autonomous dynamics, $u \in \mathbb{R}^m$ the input space, $B \in \mathbb{R}^{n\times m}$ the input matrix and $a \in \mathbb{R}^n$ any constant dynamics.

The first method for obtaining an affine system valid on a given simplex is based on barycentric linearization, and is found by considering the system

$$\dot{x} = f(x, u) \text{ with Jacobian } \mathrm{D}f(x, u)$$

and linearization point x_0 - the barycenter of the simplex.

The linearization of interest thus becomes

$$\dot{x} = f(x_0, 0) + \mathrm{D}f(x_0, 0) \begin{pmatrix} x - x_0 \\ u \end{pmatrix},$$

which results in the following system matrices: $A = \mathrm{D}_x f(x_0, 0)$, $B = \mathrm{D}_u f(x_0, 0)$ and $a = f(x_0, 0) + \mathrm{D}_x f(x_0, 0) \begin{pmatrix} -x_0 \end{pmatrix}$ This linearization represents a good linear approximation of the system in the simplex and it is very simple and fast to calculate in practical applications. However, the linearization has the downside, that the linearizations in the neighboring simplices do not agree on the facets, which leads to a discontinuous system model. In order to remedy this a method for linearization, which agrees on the boundary will be presented.

In order to generate a linearization, which agrees on the facets of the simplices with neighboring simplices the systems dynamics at the vertices of the simplex is used.

First the autonomous part of the system is identified, which is done by considering the dynamics of the system at each vertex. The autonomous system is: $\dot{x} = Ax + a$, which on each vertex gives: $\dot{x}_{v_i} = Av_i + a \Leftrightarrow v_i^T A^T + a^T = \dot{x}_{v_i}^T \Rightarrow$

$$\begin{pmatrix} v_1^T & 1 \\ v_2^T & 1 \\ \vdots & \vdots \\ v_{n+1}^T & 1 \end{pmatrix} \begin{pmatrix} A^T \\ a^T \end{pmatrix} = \begin{pmatrix} \dot{x}_{v_1} \\ \dot{x}_{v_2} \\ \vdots \\ \dot{x}_{v_{n+1}} \end{pmatrix}. \tag{5.1}$$

Since the vertices of the simplex are independent, (5.1) always has a unique solution. What is left is to find the input matrix, B, which can be computed as described previously under barycentric linearization.

Not only does this linearization possess the nice property, that it agrees with the dynamics of neighboring simplices on common facets, but this method of linearization is also applicable on systems, where the underlying model of the system is unknown, by replacing the derivative $\dot{x}_{v_i}$ with the approximation $\dot{x}_{v_i} \approx \frac{v_{i+\tau} - v_i}{\tau}$.

Note, that all linearizations does not necessarly need to be available apriori. A receding horizon approach could be considered where only the nearest neighbours are linearized whenever entering a new simplex.

5.2.2 Control Combinatorial Vector Fields

In this section control on the individual simplices is investigated. Such control has previously been studied in [Habets and van Schuppen, 2004] and [Habets *et al.*, 2006] where the control objective was decomposed into two control problems posed for each n-simplex σ (in fact in [Habets and van Schuppen, 2004] the authors treat more general problem of control synthesis on a polytope):

Problem 1 (Problem 4.1 in [Habets *et al.*, 2006]). *Let $\sigma \in K_n$, K_n being the n-dimensional complex. Given a subset S of maximal faces, i.e. co-dimension 1 simplices, of σ find a* *control law*

$$k_\sigma : \mathrm{Im}(\sigma) \to \mathbb{R}^m,\ k_\sigma(x) = F_\sigma x + g_\sigma, \tag{5.2}$$

where F_σ is an m by n matrix and g_σ is an n-vector, such that it guarantees that all flow lines of the closed-loop system

$$\dot{x} = (A_\sigma + B_\sigma F_\sigma)x + (a_\sigma + B_\sigma g_\sigma), \tag{5.3}$$

starting at a $p \in \mathrm{Im}(\sigma)$ leaves the simplex σ in finite time by crossing one of the faces in S.

Problem 2 (Problem 4.2 in [Habets *et al.*, 2006])**.** *For a given $\sigma \in K_n$ find a control law (5.2) such that for any $p \in \mathrm{Im}(\sigma)$ the flow line $\phi_p(t)$ of the closed-loop system (5.3) satisfies $\phi_p(t) \in \mathrm{Im}(\sigma)$ for any $t \geq 0$.*

Problems 1 and 2 are solved in [Habets *et al.*, 2006] by blocking facets that are complementary to the set S. A facet F_1, with outward unit normal n_1, is called blocking if $\langle \dot{x}_\sigma, n_1 \rangle < 0$. We observe that if $S' \subset S$ and the control law k_σ blocks all the faces in S then it also blocks the faces in S'; thus the more blocking faces the more restrictive control becomes. For a treatment of the necessary and sufficient conditions for guaranteeing control to a certain facet of a simplex for the piece wise affine system the reader is referred to [Habets *et al.*, 2006] and the references therein.

Thus for each simplex there exists a set of controllers k_{σ_n}, possibly empty, which guarantees that the system will leave the simplex through a known subset of its facets.

Other types of controllers, e.g. nonlinear controllers, may of course be considered as long as it guarantees exit through a specific facet of subset of facets. Affine controllers are considered here because they can be generated automatically.

5.2.3 Combinatorial Vector Fields

Let K be a simplicial complex. A **combinatorial vector field** V on K is a family $\{V_n |\ n \in \mathbb{N}\}$ of maps [Wisniewski and Larsen, 2008]

$$V_n : K_{n-1} \to K_n \cup \{0\}$$

that satisfies $V_n \circ V_{n-1} = 0$, that is if $\sigma \in \mathrm{Image}(V_{n-1})$ then $V_n(\sigma) = 0$.

Alternatively a combinatorial vector field is a set $\bar{V}$ of pairs of simplices $\langle \alpha, \sigma \rangle$, where α is a maximal face of σ. It is helpful to picture a combinatorial vector field on K by arrows, where the tail is at α and the arrow at σ, as illustrated in Fig. 5.2.

Using the combinatorial vector field as an abstraction of the original system it is possible to calculate flow and flow line of the combinatorial vector field. For this the notion of a combinatorial scalar product is needed, which for simplicial complexes is $\langle \cdot, \cdot \rangle : K_n \times K_n \to \{0, 1\}$, defined by

$$\langle \sigma, \alpha \rangle = \begin{cases} 1 & \text{if} \quad \sigma = \alpha \\ 0 & \text{otherwise} \end{cases}$$

We extend it to the bilinear product $\langle \cdot, \cdot \rangle : C_n(K) \times C_n(K) \to \mathbb{Z}$. In particular

$$\left\langle \sum_j n_j \sigma_j, \sigma_k \right\rangle = n_k.$$

Define $\theta_n : C_n(K) \to C_{n-1}(K)$ by

$$\theta_n(\sigma) = \sum_{i=0}^{n} (-1)^i \langle V_n \circ d_i^n(\sigma), \sigma \rangle \, d_i^n \sigma,$$

with $d_i^n(\sigma)$ being the face map of σ [May, 1992]. The map θ takes $\sigma \in K_n$ to a linear combination of the simplices in $V_n^{-1}(\sigma)$, i.e. for a given simplex it pronounces which facets the flow possibly came from.

With this we define the notion of a flow: A **flow** (of a nondeterministic combinatorial vector field) is the map $\Phi_n : C_n(K) \to C_n(K)$ given by

$$\Phi_n = (\partial_{n+1} - \theta_{n+1}) \circ V_{n+1} + V_n \circ (\partial_n - \theta_n).$$

Consider the nondeterministic combinatorial vector field defined in Fig. 5.2. It is

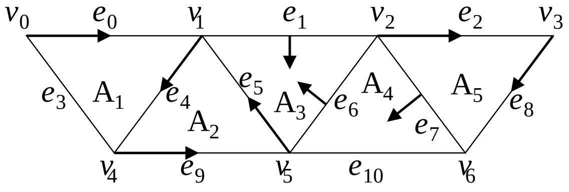


Figure 5.2: An example of a simplicial complex with its associated vector field. The vertex v_3 is a rest point.

possible to calculate the flow starting at the 2-simplex A_4:

$$\begin{aligned} \Phi_2(A_4) &= (\partial_3 - \theta_3)V_3 + V_2(\partial_2 - \theta_2) \\ &= (\partial_3 - \theta_3)0 + V_2(e_7 - e_{10} + e_6 - e_7) = A_3. \end{aligned}$$

The result is the 2-simplex A_3, which corresonds to our expectation, as seen from Fig. 5.2. Correspondingly, a flow line, γ, is used to denote a sequence of flows, i.e. $\gamma = \sigma, \Phi(\sigma), \Phi \circ \Phi(\sigma), \Phi \circ \Phi \circ \Phi(\sigma), \ldots$

5.2.4 Lyapunov Function

In order to specify the control objectives for the system we introduce the notion of a Lyapunov function, $f : K \to \mathbb{R}$. It has been shown in [Franks, 1980] and [Forman, 1998] that such Lyapunov functions for combinatorial dynamical systems exist with the same properties as for continuous dynamical systems, i.e. is descreasing along flow lines of the system. There are no general limitation to such a function except that it should be monotonic with a global minimum at the goal simplex.

One obvious Lyapunov candidate is the distance to the goal simplex. Assigning a value to each simplex in the simplicial complex is done by iteratively associating to every simplex an increasing number starting from the goal simplex. Thus - for example all neighboring simplices to a goal are given a value higher than the goal value, and etc. until the entire complex has been covered. At this step simplices, which represent unsafe areas, i.e. prohibited areas which the system under no circumstances is allowed to enter is not given any value, and is thereby not considered in the following synthesis.

Another possibility is to use a potential field alike function as described in [Hwang and Ahuja, 1992; Koren and Borenstein, 1991] with minimum at σ_{goal} and maximum at σ_{init}, and then for each simplex apply the average over the simplex as a value of the Lyapunov candidate function at this simplex. The disadvantage of potential field functions is that they might have local minima.

5.2.5 Control Synthesis

Let L_0 be a finite set (of control actions). A **combinatorial control system** is a map

$$\mu : K_n \times L_0 \to C_{n-1}(K)$$

such that for every $l \in L_0$, the restriction of μ to $K_n \times \{l\}$, is an n-combinatorial vector field.

By having the control objective specified by a Lyapunov function f it is now possible to synthesize a control system by selecting a control action from L_0 that satisfies the conditions of the Lyapunov function.

Denote the set of all possible combinatorial control systems satisfying the control objective given by the Lyapunov function f by Ω. If $\Omega \neq \emptyset$, then all flow-lines of the closed loop system $\Phi^{\omega}(\sigma)$ stops at σ_{goal}, where Φ^{ω} is the the flow for the n-combinatorial vector field with controller $\omega \in \Omega$ There might be, however, more than one combinatorial control system in Ω. This flexibility can be used to select the control $\overline{\omega}$, which minimizes a certain cost function. Let $H : K_n \times L_0 \to \mathbb{R}$ be a function,

$$\overline{\omega} = \arg\min_{\omega\in\Omega}\max_{\gamma_\omega}\sum_{\sigma\in\gamma_\omega} H(\sigma,\omega(\sigma)),$$

where γ_ω is a flow line for the n-combinatorial vector field $\Phi^{\omega}(\sigma)$ that starts at σ_{init} and stops at σ_{goal}.

In practice one wants that the decision of the control action is local. For each simplex the set of possible control actions is determined to find the next feasible controller. Let $J : K_n \to \mathbb{R}$ be a function. For an initial simplex σ_{init} and a goal simplex σ_{goal} we want to find a combinatorial control such that at each simplex σ, $\max J(\Phi^{\omega}(\sigma))$ is minimal - The following procedure can be used:

```
$\sigma = \sigma_{\text{init}}$
While $\sigma \neq \sigma_{\text{goal}}$
  $\overline{\omega} = \arg\min_{\omega \in \Omega} \max J(\Phi^{\omega}(\sigma))$
  $\sigma = \Phi^{\omega}(\sigma)$
```

5.3 Example

In this section a numerical example is given. It comprises path planning and obstacle avoidance for a small robot modeled as a unicycle, as described in [Oriolo *et al.*, 2002].

State Space. The domain of the example is a 100 by 100 square, which has been triangulated into the simplicial complex shown in Fig. 5.3, where the diamond indicates the starting location, the circle - the goal location, the squares - unsafe regions and stars - safe regions.

Lyapunov function. To aid the control synthesis a Lyapunov function is used. In this example the Lyapunov function f maps a simplex to the smallest number of simplices joining this simplex with the goal simplex. For simplicity only the values of the Lyapunov function at the 2-simplices are depicted in Fig. 5.3.

Modeling. The model of a unicycle robot is derived in [Oriolo *et al.*, 2002]. The work shows that the robot's mathematical model, through exact feedback linearization, can be linearized wrt. angle and position.

In a cartesian coordinate system the position is described by

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \dot{p} \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}. \tag{5.4}$$

To simplify the problem two controllers are used. The first controller keeps the robot at a constant speed. This is accomplished by a standard proportional control with a proportional gain of $k_p = 10$ and a reference of $\dot{p}_{ref} = 2$. The second controller regulates the robot's angle, which is carried out by using a proportional-differential controller with $k_p = 20$ and $k_d = 30$ and as input reference the desired angle θ_{ref}.

By using these two controllers the system given by (5.4) can be reduced to

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \dot{p}_{ref} \begin{pmatrix} \cos\theta_{ref} \\ \sin\theta_{ref} \end{pmatrix}. \tag{5.5}$$

Since, throughout this example, $\dot{p}_{ref} = 2$ the affine system approximation of (5.5) becomes

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = 2 \begin{pmatrix} \sin\bar{\theta} \\ -\cos\bar{\theta} \end{pmatrix} \theta_{ref} + 2 \begin{pmatrix} \cos\bar{\theta} \\ \sin\bar{\theta} \end{pmatrix}, \tag{5.6}$$

with $\bar{\theta}$ being the linearization point for the given simplex.

Controllability. Given the system model (5.6) the admissible and blocking facets are identified. Suppose that the flow enters a simplex σ perpendicular to a facet and enumerate the facets of σ from F_1 to F_3 starting from the facet through which the simplex was entered. Then it is seen from (5.6) that $\{F_1, F_2\}$ and $\{F_1, F_3\}$ are sets of blocking facets, which correspond to admissible facets F_2 or F_3, i.e. there exists a controller forcing the system to exit either through F_2 or F_3. Note that the system is not stabilizable in the equilibrium point, due to the assumption of constant velocity. Including this in the control synthesis it would however become stabilizable.

Control synthesis. The algorithm selecting the combinatorial control systems Ω, satisfying the Lyapunov function f is applied. No cost function has been utilized in this example, thus when the algorithm finds two identical Lyapunov values for neighbouring simplexes the first encountered by the program is chosen. In Fig. 5.3 the solid line shows the control synthesis.

Numerical simulation. To verify that the developed controller executes the control actions as desired, the original non-linear system is simulated with the combinatorial control law. Fig. 5.3 depicts the resulting trajectory of the closed loop system by the dashed line. The solid line represents the discrete control. It is seen that the synthesized controller makes the original non-linear system behave as specified.

5.4 Conclusion

In has been shown how control synthesis for a large class of dynamical systems including both non-linear and switched systems can be performed automatically. The state space has been partitioned into simplices, in which the system dynamics has been approximated by a set of affine systems with agreeing dynamics along the facets. The control objectives have been expressed in terms of the combinatorial Lyapunov function and the discrete control has been devised that fulfills these objectives. The methodology has been illustrated by application to robot guidance, where the unicycle trajectory planning problem was considered.

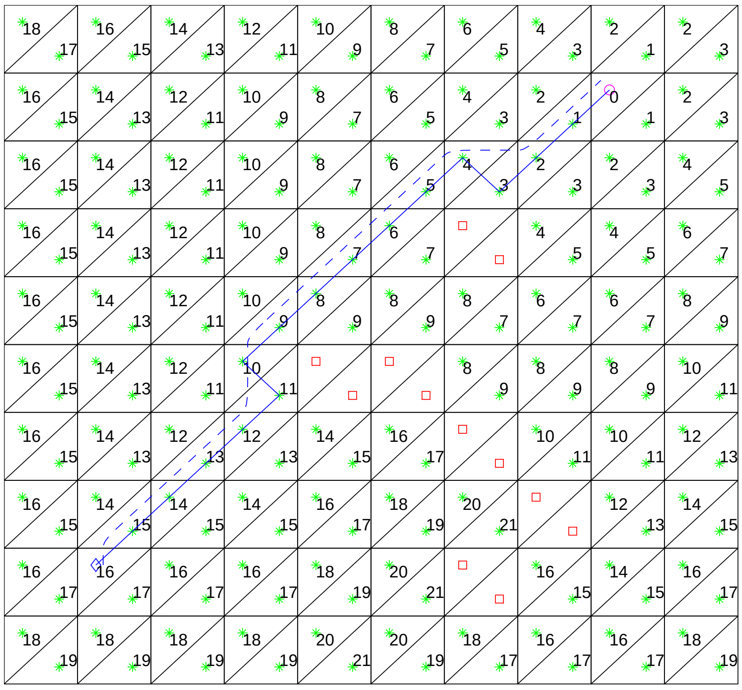


Figure 5.3: The simplicial complex considered in the example along with assigned Lyapunov function values, the synthesized combinatorial control system, indicated by the solid line, and the control system applied to the original system shown by the dashed line.

Part II

Combinatorial Approach

This part contains the contributions made towards establishing a combinatorial equivalent of continuous hybrid systems.

Chapters

Chapter 6

Introduction to Combinatorics

The following chapter will try to emphasize some of the advantages of developing a purely combinatorial equivalent of continuous dynamical systems as well as hybrid systems along with a comparison of the different proposed methods.

6.1 Relevance of a Combinatorial Equivalent

As shown in the first part of this thesis, hybrid systems intrinsically contains both discrete parts, in form of switching between locations, and continuous parts, in form of the continuous dynamics in each location. In order to handle, i.e. analyse and control, such systems there are basicly two approaches:

- Develop methods for handling hybrid systems and thereby encompassing both the continuous and discrete nature of the system in one framework.

- Translate everything into one system being either discret or continuous.

The first approach has the advantage, that it is keeping quite close to the original way of thinking about the problem, i.e. as a set of locations with continuous dynamics in each location. However, when comming to building analysis and control synthesis tools for such systems the task becomes quite difficult. A few works in this direction are [Chutinan and Krogh, 2003; Larsen *et al.*, 2004].

Going in the other direction and translating the entire system into either a discrete or continuous system has the advantage, that many tools have already been developed for purely discrete and purely continuous systems. However, the downside is, that in order to use these tools the continuous dynamics first needs to be translated into a discrete equivalent or visa versa. Something which can be quite a dificult task in its own.

- *Translating the discrete transactions into continuous equivalents.* In order to make a purely continuous model of the system the discrete nature of hybrid systems, i.e. the switch between locations needs to be translated into a continuous equivalent. One work in this direction is shown in [Wisniewski, 2006], where it is shown how a continuous realization can be made by connecting the continuous dynamics through a flow strip with pure integrater dynamics defined on it as shown in figure 6.1. This

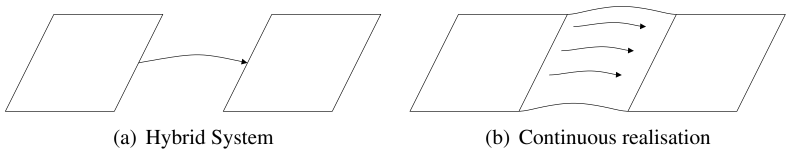

(a) Hybrid System (b) Continuous realisation

Figure 6.1: (a) The original hybrid system consisting of two locations and one jump. (b) A continious realisation of the system with the jump replaced by a flow strip with pure integrators defined on it.

theory has been shown used on a stability analysis for a distributed cooling system in [Li and Wisniewski, 2006].

- *Translating the continuous dynamics into a discrete equivalent.* As mentioned earlier it is also possible to go the other way, i.e. translate all continuous parts into discrete equivalents. In order to do this the continuous section of the system can be split into a number of smaller parts, each which can be approximated by a linear or possibly affine system. Which of these cells it is possible to go between as a function of the applied control law and input restriction, can now be found. A simple illustration of this is shown in figure 6.2.

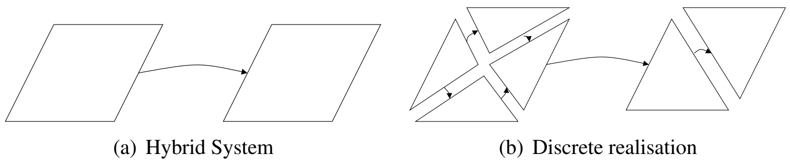

(a) Hybrid System (b) Discrete realisation

Figure 6.2: (a) The original hybrid system consisting of two locations and one jump. (b) A discrete realisation of the system with the contiuous dynamics replaced by discrete abstractions.

Since a large amount of research effort has been put into handling purely discrete systems - primarily by the computer science community, e.g. [Larsen *et al.*, 2004] it was found, that the shortest path towards a working theory will be to focus on developing a discrete abstraction of the contiunous part of the system and to develop an algebra capable of handling such discrete systems.

In the following only the key differences of the three methods will be mentioned, which are their definitions of vector fields and flow.

6.2 Previous Work

One of the main works in developing a discrete abstraction of dynamic systems is done in [Forman, 1998]. In this paper the following definitions for vector fields and flows are introduced.

Definition 5. *The Forman vector field is given as:* $V : K_n \rightarrow K_{n+1} \cup \{0\}$*, with the following restrictions:*

1. *For each* $\sigma^{(p)} \in K_p$*, either* $V(\sigma) = 0$ *or* σ *is a regular face of* $V(\sigma)$*.*
2. *If* $\sigma \in \mathrm{Image}(V)$ *then* $V(\sigma) = 0$*.*
3. *For each* $\sigma^{(p)} \in K_p$*:* $\sharp\{v^{(p-1)} \in K_{p-1} | V(v) = \sigma\} \leq 1$*,*

with $\sharp$ denoting the cardinality. In [Forman, 1998] the notion of illustrating the vector fields by arrows is introduced analog to how arrows are used to indicated flow directions in phase plots for continuous dynamical systems. An example of such is shown in figure 6.3. The above definition can in terms of these arrows be expressed as follows:

- An arrow goes from a lower dimension to a one dimension higher simplex.
- A simplex cannot be both a head and a tail for a vector field.
- Each simplex can at maximum be the head or tail of one vector field.

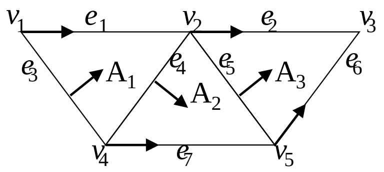


Figure 6.3: A combinatorial vector field. The vertex v_3 is a rest point.

Simplices which are neither the head nor the tail of a vector field are called rest points. Another aspect of the paper is Formans definition of flow, which is as follows:

Definition 6 (Def. 4.2 in [Forman, 1998])**.** *Define the (discrete time) flow* $\Phi : C_*(M, \mathbb{Z}) \rightarrow C_*(M, \mathbb{Z})$ *by*

$$\Phi = 1 + \partial V + V\partial \tag{6.1}$$

This flow definition is nice in the sence that it preserves homology, however when starting to do computations with it future simplices are not always the ones which intuitively should be the future simplices. Furthermore, it is difficult to interpretate the multiplicity of a future simplex, i.e. how to interprete $\Phi(\sigma_1) = 3\sigma_2 - 2\sigma_3$. Finally, from the definition of the vector field it is only allowed for the vector field to split into more directions, but it is not allowed to merge flows, something which whould be nice to have in practice as a dual to splitting.

6.3 First Method

To remedy these shortcommings of Formans method a modified version of his combinatorial vector fields is introduced. A more thorough introduction of this is given through chapter 7 and 8 where a practical application is also introduced.

In changing the combinatorial vector field, firstly the definition of vector fields is changed in order to allow for merging flows, which result in the following definition:

Definition 7. *Let K be a simplicial complex. A* ***nondeterministic combinatorial vector field*** *V on K is a family $\{V_n | \, n \in \mathbb{N}\}$ of maps*

$$V_n : K_{n-1} \to K_n \cup \{0\}$$

that satisfies $V_n \circ V_{n-1} = 0$.

That is, a simplex cannot be the head of an arrow and at the same time the tail of another arrow, however it is now possible for a simplex to have multible heads.

The definition of flow is also modified from Formans definition in order to remedy the counter-intuitive behaviour of the flow. This is done by introducing the θ operator, which can be interpreted as a *pre* operator returning co-dim 1 simplices which has the current simplex as a result of its flow.

Definition 8. *Let $\theta_n : C_n(K) \to C_{n-1}(K)$ be given by*

$$\theta_n(\sigma) = \sum_{i=0}^{n} (-1)^i \langle V_n \circ d_i^n(\sigma), \sigma \rangle \, d_i^n(\sigma),$$

where d_i^n is the ith n-dimensional face, and $\langle \cdot, \cdot \rangle : K_n \times K_n \to \{0, 1\}$ the combinatorial scalar product, defined by

$$\langle \sigma, \alpha \rangle = \begin{cases} 1 & \text{if} \quad \sigma = \alpha \\ 0 & \text{otherwise} \end{cases}$$

By utilizing definition 8 it is possible to define the combinatorial flow as:

Definition 9. *$\Phi_n : C_n(K) \to C_n(K)$ given by*

$$\Phi_n = (\partial_{n+1} - \theta_{n+1}) \circ V_{n+1} + V_n \circ (\partial_n - \theta_n),$$

with ∂_n being the boundary map given by $\partial_n = \sum_{i=0}^{n}(-1)^i d_i^n$

Using this extention of the combinatorial vector field and flow it is now possible to handle both splitting and merging of flows, an important duality, which is needed in order to consider both forward and backward flow. However, one downside of this definition is, that it does not preserve homology.

6.4 Second Method

In the previously described method a move was taking away from a rigirously and nice formulation of combinatorial vector fields and flows towards a more intuitive and practical formulation. In this second method a purely geometric interpretation of the combinatorial vector fields and flows will be considered, which also allows for extending the notation to include polyhedral complexes. A thorough introduction to this method is given in chapter 9.

Firstly, by changing the direction of flow, i.e. letting it be directed as $V_n : K_n \to C_{n-1}$ it gives a more intuitive feel of the flow from an engineering point of view. Thus the flow is defined as:

Definition 10. *Let K be a complex. An i-vector field is a map*

$$\nu_i : K_i \to C_{i-1}(K)$$

that satisfies

1. *for any $P \in K_i$, $\nu_i(P) \in C_{i-1}(\mathrm{cl}(P))$.*
2. *for any pair $(F, P) \in K_{i-1} \times K_i$, $\langle F, \nu_i(P)\rangle \nu_{i-1}(F) = 0$.*
3. *for any pair $(P, Q) \in K_i \times K_i$ with $P \neq Q$, $\langle \nu_i(P), \nu_i(Q)\rangle = 0$*

The first requirement states, that the vector field of a cell will end at a linear combination of its facets. The second condition states that a cell cannot both be the head of a vector field and at the same time a tail for one. The final condition states, that a cell can only be the head of one vector field.

For this combinatorial vector field it is now possible to associate the following flow.

Definition 11. *An i-flow (of a combinatorial vector field) is the homomorphism*[1] *$\Phi_i : C_i(K) \to C_i(K)$ given by*

$$\Phi_i(P) = \sum_{Q \in \mathrm{Ad}_i(P)} \langle \partial_i Q, \nu_i(P)\rangle\, Q + \sum_{Q \in \mathrm{St}_{i+1}(P)} \langle \partial_{i+1} Q, P\rangle\, \nu_{i+1}(Q) + P,$$

[1] Φ_i is a homomorphism if and only if for any $P, Q \in C_i$, $\Phi_i(P + Q) = \Phi_i(P) + \Phi_i(Q)$.

for $P \in K_i$, with Ad_i and St_{i+1} being the Adjacent neighborhood and Star of a polytope respectively, as described in section 9.5.

6.5 Comparison of Methods

Given the three methods described in the previous, a comparison of the methods will be given in the following looking at five distinct cases, which are: Source, sink, merging, splitting and flow of combinatorial vector fields.

6.5.1 Source

A pictorial representation of a source is given in figure 6.4 for a 2-dimensional source using the first method and the second method respectively.

The flow calculations for a source using the three mentioned methods looks like the following:

- Forman:

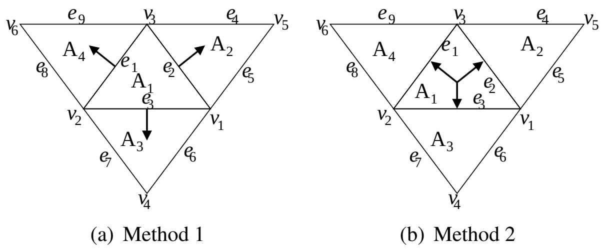
$$\Phi(A_1) = A_1 + \partial V(A_1) + V\partial(A_1) = A_1 - A_2 + A_3 + A_4$$

- First method:

$$\Phi(A_1) = (\partial - \theta)V(A_1) + V(\partial(A_1) - \theta(A_1)) = A_4 - A_2 + A_3$$

- Second method:

$$\Phi_2(A_1) = \sum_{Q \in \mathrm{Ad}_2(A_1)} \langle \partial_2 Q, V_2(A_1) \rangle Q + \sum_{Q \in \mathrm{St}_3(A_1)} \langle \partial_3 Q, A_1 \rangle V_3(Q) + A_1 = A_2 + A_3 + A_4$$



(a) Method 1 (b) Method 2

Figure 6.4: (a) A source in the formulation of Forman and the first method. (b) A source using the second method.

6.5.2 Sink

The three different sink representations for the three methods are shown in figure 6.5 for a 2-dimensional sink.

The flow calculations for a sink using the three mentioned methods looks like the following:

- Forman:

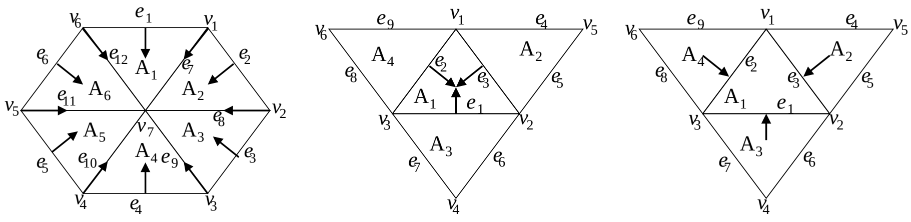
$$\Phi(A_1) = 0, \quad \Phi(e_1) = 2e_7, \quad \Phi(e_7) = 0, \quad \Phi(v_7) = v_7$$

- First method:

$$\Phi(A_1) = 0, \quad \Phi(e_1) = 0$$

- Second method:

$$\Phi(A_1) = A_1, \quad \Phi(e_1) = 0$$



(a) Forman (b) Method 1 (c) Method 2

Figure 6.5: (a) A sink in the formulation of Forman. (b) A sink using the first method. (c) A sink using the second method.

6.5.3 Splitting

Splitting using Formans and the first method is shown in figure 6.6(a) and for the second method in figure 6.6(b).

The flow calculations for this split using the three mentioned methods looks like the following:

- Forman:

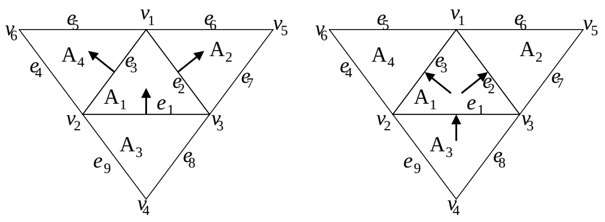
$$\Phi(A_1) = A_2 + A_3$$

- First method:

$$\Phi(A_1) = -A_2 + A_3$$

- Second method:

$$\Phi_2(A_1) = A_2 + A_3$$



(a) Forman & Method 1 (b) Method 2

Figure 6.6: (a) Splitting of flow in the formulation of Forman and the first method. (b) Splitting of flow using the second method.

6.5.4 Merging

Merging of flow using the first method is shown in figure 6.7(a) and for the second method in figure 6.7(b).

The flow calculations for this merging using the three mentioned methods looks like the following:

- Forman:

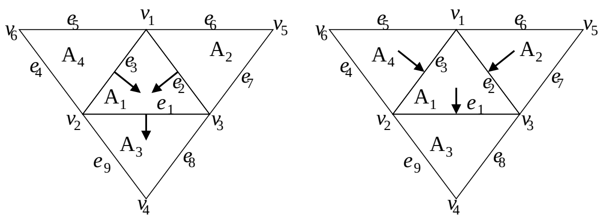
$$\text{Not possible}$$

- First method:

$$\Phi(A_1) = A_3$$

- Second method:

$$\Phi_2(A_1) = A_3$$



(a) Method 1 (b) Method 2

Figure 6.7: (a) Merging of flow in the formulation of the first method. (b) Merging of flow using the second method.

6.5.5 Flowing

The last example given will be a bit bigger, which aim to show that the flow is well-behaving, in the sense that it gives the expected future polytopes.

The simplicial complex considered here is depicted in figure 6.8, which can be seen as a partition of a space in $\mathbb{R}^2$ with the system $\dot{x} = [1\,0]^T$ defined on all of it.

The flow calculations for this using the three mentioned methods looks like the following:

- Forman:

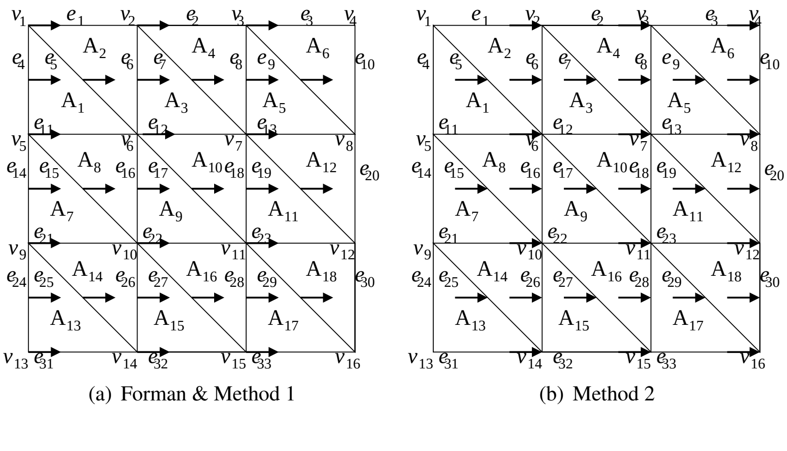
$$\Phi(A_{10}) = A_{11}, \quad \Phi(e_{17}) = e_{18}+2e_{12}-e_{23}, \quad \Phi(e_{22}) = 2e_{22}-e_{23}, \quad \Phi(v_{10}) = v_{11}$$

- First method:

$$\Phi(A_{10}) = A_{11}, \quad \Phi(e_{17}) = e_{18} + 2e_{12} - e_{23}, \quad \Phi(e_{22}) = e_{23}, \quad \Phi(v_{10}) = v_{11}$$

- Second method:

$$\Phi(A_{10}) = A_{11}, \quad \Phi(e_{17}) = e_{18}, \quad \Phi(e_{22}) = e_{18} + e_{23} + e_{29}, \quad \Phi(v_{10}) = v_{11}$$



(a) Forman & Method 1 (b) Method 2

Figure 6.8: (a) A flow in the formulation of Forman and the first method. (b) The same flow using the second method.

6.6 Conclusion

By comparing the results of the flow computations performed in the previous section table 6.1 can be produced listing the different methods properties agains each other.

For a source it is not desirable that the flow calculation will show that it potentially stays at the location, since it is a fixed point with measure 0. However, when calculating the flow of a sink the location itself should be returned. For the big flow example it is seen, that both method 1 and 2 gives nice results for dimension n and dimension 0 flows. However, they both have problems handling the intermediate dimensions.

	Source	Sink	Split	Merge	Flow
Forman	$\div$	$\surd$	$\surd$	$\div$	$\div$
Method 1	$\surd$	$\div$	$(\surd)$	$\surd$	$(\surd)$
Method 2	$\surd$	$\surd$	$\surd$	$\surd$	$(\surd)$

Table 6.1: Comparison of methods for combinatorial flow calculations.

Chapter 7

Combinatorial Hybrid Systems

As initially suggested by E. Sontag [Sontag, 1981; 1982] it is possible to approximate an arbitrary nonlinear system by a set of piecewise linear systems. In this work we concentrate on how to control a system given by a set of piecewise linear systems defined on simplices. By using the results of L. Habets and J. van Schuppen [Habets and van Schuppen, 2001] it is possible to find a controller for the system on each of the simplices thus guaranteeing that the system flow on the simplex only will leave the simplex through a subset of its faces. Motivated by R. Forman [Forman, 1998], on the triangulated state space we define a combinatorial vector field, which indicates for a given face the future simplex. In the suggested definition we allow nondeterminacy in form of splitting and merging of solution trajectories. The combinatorial vector field gives rise to combinatorial counterparts of most concepts from dynamical systems, such as duals to vector fields, flow, flow lines, fixed points and Lyapunov functions. Finally it will be shown how this theory extends to switched dynamical systems and an algorithmic overview of how to do supervisory control will be shown towards the end.

7.1 Introduction

Recently a number of initial steps have been taken towards truly automatic control of switched dynamical systems [Tabuada and Pappas, 2003b; Bemporad *et al.*, 2000] and the references therein. Giving a system the ability to automatically reconfigure itself has received enormous attention during the past decade, particularly in the fields of fault tolerant control (FTC) and navigation of autonomous robots. In the first case, the system dynamics can exhibit rapid changes, in the latter it is the environment's dynamics that changes. Common for both applications however are, that whenever a change occurs, the system will start to either recalculate a new control law, or select it from a set of pre-analyzed and -programmed controllers. Albeit doable, such reconfigurations requires great computational resources; recourses, which are often not available on smaller autonomous platforms, thus a more efficient method is desired.

In the autonomous robot case the guidance problem has previously been handled by two different methods, one being model predictive control, [Falcone *et al.*, 2007; Kouvaritakis and Cannon, 2001], where the constraints describe the safe area and the system trajectory is being simulated within a finite horizon through which the system is verified not to hit an unsafe area. Another approach has been to use energy shaping, in which obstacles are modeled as maxima of potential fields, and the control goal as the minimum, thus following the negative gradient vector field will lead to the goal [Barraquand *et al.*, 1992; Koren and Borenstein, 1991]. Finally, and more recently, the problem has been addressed by [Tabuada and Pappas, 2003b], where the system is modelled as an automaton. The control design is then reduced to a finite discrete state supervisor that satisfies system requirements. An application example of such a system in multi-robot planning can be found in [Quottrup *et al.*, 2004] and in [Larsen *et al.*, 2007] it is applied to a switched system.

In the FTC case each fault scenario has a dedicated controller that has been developed for each of them. The switching between these controllers have then been left to a supervisory controller, which relies on reliable fault detection and isolation (FDI) methods to decide when to switch [Izadi-Zamanabadi and Larsen, 2007].

This paper deals with combinatorial formulation for piecewise-affine control systems. It is thought as a carry-over of [Habets and van Schuppen, 2004] and [Habets *et al.*, 2006], which address the control problem for piecewise-affine systems on an arbitrary polytope that forces the solution trajectories of the closed loop system to either leave it or stay in it for ever. This paper merely considers the underlying discrete system. Interest of the control community in systems defined on simplicial objects has been initiated by [Sontag, 1981; 1982]. Reachability and controllability on such systems have been studied before in [Asarin *et al.*, 2000a; Bemporad *et al.*, 2000]. Whereas previous methods have been based on the concept of a transition system, this paper focuses on its higher dimensional generalization, a simplicial complex.

In this paper a complete design flow for control on combinatorial hybrid systems will be given. Firstly the concepts of embedding a control system on a simplicial complex is introduced in section 7.2 and 7.3. The main contribution of this paper is a formulation of a combinatorial dynamical system in section 7.3. Section 7.4 gives a practical relation between the combinatorial system and the original continuous system, which will be illustrated through an example. Finally control synthesis is addressed in section 7.5, where the concept of a combinatorial dynamical system is expanded to include the hybrid nature of a class of hybrid systems. The section gives an overview and a conceptual example.

We use the following notation: $\mathbb{Z}$ is the set of integers, $\mathbb{N}$ is the set of natural numbers, $\mathbb{Z}_+ = \{n \in \mathbb{Z} | \, n \geq 0\}$. A facet is a maximum dimensional face of a simplex. Throughout the paper it is assumed, that the system is given as a simplicial complex with associated affine systems defined on each simplex. It is outside the scope of this paper to go into the details of simplicial objects. For a more mathematical background on the usage of simplicial complexes the reader is referred to [Wisniewski and Larsen, 2008], and more generally [Munkres, 1973] and [Lee, 2002].

Paraphrasing the definition of the hybrid automata in [Henzinger, 1996a] and a hybrid system in [Branicky, 1998] the hybrid systems definition employed in this paper is a pair of sets.

- A finite set of dynamic systems defined on manifolds.
- A finite set of transfer maps, which maps a subset of one manifold to another.

7.2 Combinatorial Vector Fields

In this section we introduce the central notion of this paper - a combinatorial vector field. The notion has been developed by R. Forman in [Forman, 1998] for studying topological invariants of CW complexes. Here it is extended to deal with nondeterminacy encountered in hybrid systems. The attention in this paper is restricted to geometrical properties of a combinatorial vector field. It is treated as a generator of flow. The notion of combinatorial flow lines is used in Section 7.5 for synthesis of supervisory control. For a more elaborate treatment of combinatorial vector fields see [Wisniewski and Larsen, 2008].

Definition 12 (Definition 1.2, [Forman, 1998]). *Let K be a simplicial complex. A* ***combinatorial vector field*** *V on K is a family $\{V_n |\, n \in \mathbb{N}\}$ of maps $V_n : K_{n-1} \to K_n \cup \{0\}$ that satisfies*

1. *$V_n \circ V_{n-1} = 0$, that is if $\sigma \in \mathrm{Im}(V_{n-1})$ then $V_n(\sigma) = 0$.*
2. *For each $\sigma \in K_n$, the number of elements of the pre-image $V_n^{-1}(\sigma) \equiv \{\alpha \in K_{n-1} |\, V_n(\alpha) = \sigma\}$ is 0 or 1,*

where K_n is the set of n-simplices in K.

Alternatively a combinatorial vector field is a set $\bar{V}$ of pairs of simplices (α, σ), where α is a maximal face of σ, and for which no simplex is in more than one pair. It is helpful to picture a combinatorial vector field on K by arrows, where the tail is at α and the arrow at σ, see Fig. 7.1.

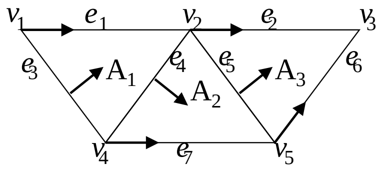
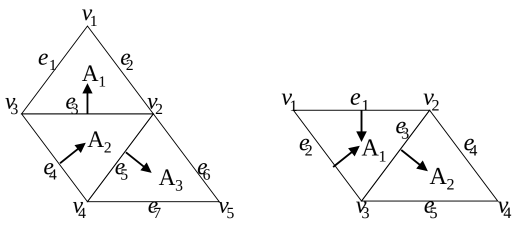


Figure 7.1: A combinatorial vector field. The vertex v_3 is a rest point.

Intuitively condition 1 of Definition 12 means that the system is of the first order; geometrically it implies that the future simplices do not increase the dimension, see the definition of the flow map below. Condition 2 excludes merging the future cells. It is illustrated in Fig. 7.2 that splitting of flow is allowed whereas merging is excluded. In Definition 14 of a nondeterministic combinatorial vector field we shall allow both situations.



(a) Splitting (b) Merging

Figure 7.2: Illustration of Definition 12. Merging on the right hand side is excluded in whereas splinting on the left hand side is allowed.

Since no simplex is in more than one pair in $\bar{V}$, every cell σ of the simplicial complex K satisfies precisely one of the following conditions:

1. σ is the tail of exactly one arrow;
2. σ is the head of exactly one arrow;
3. σ is neither the tail nor the head of any arrow.

A simplex that satisfies condition 3. is called a rest point.

Definition 13 (Definition 1.3 of [Forman, 1998])**.** *Let V be a combinatorial vector field on K. We say that $\sigma \in K_n$ is a **rest point** of V of index n if*

1. $V_{n+1}(\sigma) = 0$ *and*

2. $\sigma \notin \mathrm{Im}(V_n)$.

Section 7.3 indicates that the discrete behavior of a piecewise affine control system involves nondeterminacy induced by blocking more than one facet of a simplex. It seems therefore natural to omit condition 2 of Definition 12.

Definition 14. *Let K be a simplicial complex. A **nondeterministic combinatorial vector field** V on K is a family $\{V_n|\ n \in \mathbb{N}\}$ of maps*

$$V_n : K_{n-1} \to K_n \cup \{0\}$$

that satisfies $V_n \circ V_{n-1} = 0$.

In the remainder of this section we shall develop a notion of flow of a nondeterministic combinatorial vector field, that is a map $C_n(K) \to C_n(K)$ which takes an n-simplex to its future n-chain (a linear combination of the simplices in the very next future, for more details on chain groups see [Bredon, 1997]).

Remark 2. *The linear combination of simplices indicates nondeterminism in the future evolution. Thus for example $\tau \mapsto \sigma + \gamma$ means that the future of τ is σ or γ. In fact, the semantics adopted in this paper is such that any flow $\tau \mapsto a\sigma + b\gamma$ for $a, b \in \mathbb{Z} \setminus \{0\}$ indicates that the future of τ is σ or γ.*

For simplicial complexes we may introduce the following definition of a **combinatorial scalar product** $\langle \cdot, \cdot \rangle : K_n \times K_n \to \{0, 1\}$, defined by

$$\langle \sigma, \alpha \rangle = \begin{cases} 1 & \text{if} \quad \sigma = \alpha \\ 0 & \text{otherwise} \end{cases}$$

We extend it to the bilinear product $\langle \cdot, \cdot \rangle : C_n(K) \times C_n(K) \to \mathbb{Z}$. In particular

$$\left\langle \sum_{j \in J} n_j \sigma_j, \sigma_k \right\rangle = n_k, \; k \in J.$$

Define $\theta_n : C_n(K) \to C_{n-1}(K)$ by

$$\theta_n(\sigma) = \sum_{i=0}^{n} (-1)^i \langle V_n \circ d_i^n(\sigma), \sigma \rangle \, d_i^n(\sigma),$$

where d_i^n is the ith n-dimensional face. The map θ takes $\sigma \in K_n$ to a linear combination of the simplices in $V_n^{-1}(\sigma)$, as shown in Fig. 7.3.

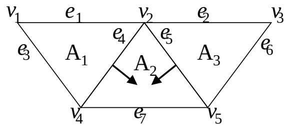


Figure 7.3: $\theta_2(A_2) = e_4 - e_5$.

Discrete dynamics of a combinatorial control systems is encapsulated in the following definition of the flow.

Definition 15. *A **flow** (of a nondeterministic combinatorial vector field) is the map $\Phi_n : C_n(K) \to C_n(K)$ given by*

$$\Phi_n = (\partial_{n+1} - \theta_{n+1}) \circ V_{n+1} + V_n \circ (\partial_n - \theta_n),$$

with ∂_n being the boundary map given by $\partial_n = \sum_{i=0}^{n} (-1)^i d_i^n$

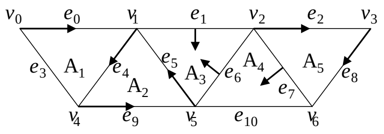


Figure 7.4: An example of a simplicial complex with its associated vector field.

Example 1. *Consider the nondeterministic combinatorial vector field defined in Fig. 7.4.*

Firstly, the flow at e_0 *is computed:*

$$\begin{aligned}\Phi_1(e_0) &= (\partial_2 - \theta_2)V_2(e_0) + V_1(\partial_1 e_0 - \theta_1 e_0)\\ &= (\partial_2 - \theta_2)0 + V_1(v_0 - v_1 - v_0) = -e_4.\end{aligned}$$

The result is the 1*-simplex* e_4*, which corresponds to our expectation, as seen from Fig. 7.4.*
Another flow of interest could be the flow from the 2*-simplex* A_4*:*

$$\begin{aligned}\Phi_2(A_4) &= (\partial_3 - \theta_3)V_3 + V_2(\partial_2 - \theta_2)\\ &= (\partial_3 - \theta_3)0 + V_2(e_7 - e_{10} + e_6 - e_7) = A_3.\end{aligned}$$

Again the flow from A_4 *to* A_3 *is found as expected.*

The flow, Φ_n, generates an n-**flow line**. An n-simplex, $\sigma \in K_n$, belongs to the n-flow line with the initial n-simplex τ if there is $k \in \mathbb{Z}_+$ such that $\langle \sigma, \Phi_n^k(\tau)\rangle \neq 0$. It is worth noticing that a flow line born in an n-simplex σ - a source - does not die in a sink, since it is a vertex (0-simplex). It dies in fact in an n-simplex belonging to the star of a sink.

With combinatorial vector fields defined, it possible to define the combinatorial counterpart of flow lines, which in the combinatorial setting will be called a V-path.

Definition 16. *A* ***V-path*** *of index* p *is a sequence of length* r*,*

$$\gamma : \sigma_0^{(p)}, \tau_0^{(p+1)}, \sigma_1^{(p)}, \tau_1^{(p+1)}, ..., \tau_{r-1}^{(p+1)}, \sigma_r^{(p)}, \tag{7.1}$$

such that for all $i \in \{0, 1, ..., r-1\}$

1. $\tau_i = V(\sigma_i)$
2. $\sigma_i \neq \sigma_{i+1} = \Phi(\sigma_i)$

If $\sigma_0 = \sigma_r$ the V-path is called closed. Two closed V-paths, γ, $\tilde{\gamma}$, are equivalent if $\tilde{\gamma}$ can be produced by selecting another starting point of γ.

A V-path is calculated by taking an initial simplex, σ_0, and propagating its flow, i.e.:

$$\sigma_0 \to V(\sigma_0) \to \Phi(\sigma_0) \to V\Phi(\sigma_0) \to \Phi\Phi(\sigma_0)\dots \tag{7.2}$$

Moreover, since non-determinism is allowed in this definition of flow the V-path is allowed to split into more paths, thus resulting in a tree of reachable locations compared to just a single track in the deterministic case.

Chain recurrent sets are sets in which the flow of an element of the set will be cyclic within the set. Thus intuitively this is the case for rest points and non-stationary closed V-paths. More formally this means that:

Definition 17 (Definition 2.1 in [Forman, 1998]). *$\sigma^{(p)} \in K$ is an element of the chain recurrent set $\mathcal{R}$ if either*

1. *σ is a rest point of V or*
2. *there is a non stationary closed V-path γ with $\sigma \in \gamma$*

As for dynamical systems, the notion of a Lyapunov function for a simplicial complex is desirable, and as shown in [Forman, 1998] and [Franks, 1980] it is possible to find a similar Lyapunov function, which has the property, that it is constant on the chain recurrent set, and outside the set it is the negative gradient of the function towards the set.

Definition 18 (Theorem 2.4 in [Forman, 1998]). *Let $\mathcal{R}$ be a chain recurrent set. There is a function $f : K \to \mathbb{R}$ such that*

1. *if $\sigma^{(p)} \notin \mathcal{R}$ and $\tau^{(p+1)} > \sigma$ then*
$\begin{cases} f(\sigma) < f(\tau) \text{ if } \tau \neq V(\sigma) \\ f(\sigma) \geq f(\tau) \text{ if } \tau = V(\sigma) \end{cases}$
2. *if $\sigma^{(p)} \in \mathcal{R}$ and $\tau^{(p+1)} > \sigma$ then*
$\begin{cases} f(\sigma) = f(\tau) \text{ if } \tau \sim \sigma \\ f(\sigma) < f(\tau) \text{ if } \tau \nsim \sigma \end{cases}$

where $\tau \sim \sigma$ means that that they belong to the same path.

This is to be understood in the following way: From the first definition, then a given V-path, not being a chain recurrent set,

$$\gamma : \sigma_0^{(k)}, \tau_0^{(k+1)}, \sigma_1^{(k)} \ldots$$

will have the following relation:

$$f(\sigma_0^{(k)}) \geq f(\tau_0^{(k+1)}) > f(\sigma_1^{(k)}) \geq \ldots,$$

which, as with continuous dynamical systems, means that the Lyapunov function is decreasing along the flow. The second condition in the definition above is saying, that for the chain recurrent set

$$\gamma_{\mathcal{R}} : \sigma_0^{(k)}, \tau_0^{(k+1)}, \sigma_1^{(k)}, \tau_1^{(k+1)}, \ldots, \tau_{r-1}^{(k+1)}, \sigma_r^{(k)} = \sigma_0^{(k)}$$

the following relation:

$$\begin{aligned} f(\sigma_0^{(k)}) = f(\tau_0^{(k+1)}) =& f(\sigma_1^{(k)}) = f(\tau_1^{(k+1)}) = \ldots = \\ & f(\tau_{r-1}^{(k+1)}) = f(\sigma_r^{(k)}) = f(\sigma_0^{(k)}), \end{aligned}$$

holds true.

Having drawn the parallels between continuous dynamical systems and combinatorial dynamical systems it is now possible to change the view to how a control system can be imposed on such combinatorial systems.

7.3 Piecewise-Affine Control System on Combinatorial Manifolds

In this section we recall the notion of a combinatorial manifold M, as described in [Hudson, 1969] - a simplicial complex of particularly regular structure. We associate to each simplex of maximal dimension in M a piecewise affine control system.

Let σ be an n-simplex. We say that a control vector field $\xi : \mathrm{Im}(\sigma) \times \mathbb{R}^m \to \mathbb{R}^n$ is piecewise-affine n-control system if ξ is defined by the piecewise-affine map

$$\xi(x, u) = Ax + Bu + a,$$

where x is an n vector, A is an n by n matrix, B is an n by m matrix, u is an m vector and a is an n-vector.

Definition 19. *A **combinatorial** n**-control system** is a pair (M, ξ), where $M = \{M_0, ..., M_n\}$ is a combinatorial n-manifold, and $\xi = \{\xi_\sigma | \ \sigma \in M_n\}$ is a family of piecewise affine n-control systems.*

With a combinatorial n-manifold defined as in [Lickorish, 1999a]. Let (M, ξ) be a combinatorial n-control system. A control objective for (M, ξ) is decomposed in [Habets *et al.*, 2006] and [Habets and van Schuppen, 2004] into two control problems posed for each n-simplex σ (in fact in [Habets and van Schuppen, 2004] the authors treat more general problem of control synthesis on a polytope):

Problem 3. *Fix $\sigma \in M_n$. Given a subset S of facets of σ find a control law*

$$k_\sigma : \mathrm{Im}(\sigma) \to \mathbb{R}^m, \ k_\sigma(x) = F_\sigma x + g_\sigma, \tag{7.3}$$

where F_σ is an m by n matrix and g_σ is an n-vector, such that it guarantees that all flow lines of the closed-loop system

$$\dot{x} = (A + BF)x + (a + Bg), \tag{7.4}$$

starting at $p \in \mathrm{Im}(\sigma)$ leave the simplex, σ, in finite time by crossing one of the facets in S.

Problem 4. *For a given $\sigma \in M_n$ find a control law (7.3) such that for any $p \in \mathrm{Im}(\sigma)$ the flow line $\phi_p(t)$ of the closed-loop system (7.4) satisfies $\phi_p(t) \in \mathrm{Im}(\sigma)$ for any $t \geq 0$.*

We say that the control law (7.3) **blocks** a facet γ of a simplex σ if the vector field ξ_σ^c of the closed loop system - defined by the right hand side of equation (7.4) - satisfies the equality

$$\langle \xi_\sigma^c(x), n_\gamma \rangle \leq 0 \tag{7.5}$$

for any $x \in \mathrm{Im}(\gamma)$, where n_γ is the outward normal vector to γ and $\langle \cdot, \cdot \rangle$ is the standard scalar product on $\mathbb{R}^n$.

Problems 3 and 4 are solved in [Habets *et al.*, 2006] by blocking facets that are complementary to the set S. We observe that if $S' \subset S$ and the control law k_σ blocks all the faces in S then it also blocks the faces in S'; thus the more blocking faces the more restrictive control it is.

7.4 Controllability on Simplices

In this section the controllability of a given simplex is studied. Firstly some general results on when it is possible to control the system to a given facet is shown. This is followed by a number of actual design procedures to form the desired control. This leads to a set of controllers, which can be selected in the combinatorial design procedure described in the next section.

The resulting control options after this section is a set of combinatorial control systems given by the map

$$V_c : K_{n-1} \times U_{n-1} \to K_n \cup \{0\},$$

from which the final controller will be selected later on.

It is desirable to look for the controllers which guarantees a single exit facet first, since this results in a deterministic flow later on. Such a test for control to a specific facet, F_1, can be done by regarding the normal vector to the facet n_1, and requires that the systems flow at the vertices of the facet will be in the same direction as the normal vector, i.e. out of the simplex. Furthermore it is required, that the flow from the opposite vertex of the face, v_1, is flowing towards the convex hull of the simplex, which is formalized in the following.

Proposition 1. *Given a n-simplex, σ_n, by the vertex set $V = v_1, \ldots, v_{n+1} \in \mathbb{R}^n$ then there exists an input sequence u belonging to the convex input set $U \subset \mathbb{R}^n$, s.t. the linear affine system, $\dot{x} = Ax + Bu + a$ defined on the simplex can be controlled to the facet, given by the normal vector n_1 if, for $i \in \{2, \ldots, n+1\}$*

$$\begin{aligned} & n_1^T(Av_i + Bu + a) > 0 \wedge \\ & n_i^T(Av_j + Bu + a) \leq 0 \forall j = \{1, \ldots, n+1\} \setminus \{i\} \wedge \\ & \Sigma_i < n_i, \dot{x}_{v_1} >< 0 \end{aligned} \tag{7.6}$$

has a solution.

This proposition is a reformulation of the results in [Habets and van Schuppen, 2001], where it is shown for convex polytopes, but is reproduced here in the language of this contribution. Having found the feasible input set it is straight forward to calculate an affine control law, as shown in [Habets and van Schuppen, 2004] and [Habets *et al.*, 2006].

7.5 Synthesis

Having the definition of simplicial complexes, combinatorial vector fields, combinatorial control systems and combinatorial flow is now possible to combine them in a supervisory control algorithm, which forces the trajectory of the closed-loop system to a reach simplex. However, before treating this algorithm the hybrid aspect of the method will be addressed.

From the definition of hybrid systems, it consists of a family of dynamical systems defined on manifolds, which in the previous sections have been transformed into discrete simplicial complexes. These are glued together by transition maps, which can be described by three properties which needs to be covered before we have a complete discrete hybrid system. These are the transition relations, E, which describe from which manifolds it is possible to jump to other manifolds, the guards, G, which describe when it is possible to leave a given manifold and the reset maps, R, which describe, where the state ends in the destination manifold.

Thus these three elements together describes a relation between how the underlying manifold of a dynamical system is connected to another underlying manifold of a dynamic system. By making this relation into an tube, and identifying the one end of it with the guard condition, and the other end with the reset condition a binding manifold is created, as depicted in fig. 7.5(a) joining i.e. M_1 and M_2.

Iteratively the entire system can now be bound together into one manifold by, for each transition relation, attaching the two continuous dynamics manifolds with a binding manifold.

Definition 20. *A continuous realization of a hybrid systems is a topological space iteratively defined by: for each $e(q, q') \in E$ glue the two manifolds $X_q, X_{q'} \in X$ by*

$$X_q \cup X_{q'} \cup G(x_q) \times \{1\}/G(x_q, 1) \sim R(x_q).$$

On the binding manifolds given by $G(x_q) \times \{1\}/R(x_q)$ the system dynamics obviously has to be given, which quite naturally is selected to be an integrator, thus giving a constant flow across the surface.

Remark 3. *Definition 20 does not give rise to a topological manifold, however if the guard functions, G, are restricted to the boundaries of the manifolds the resulting combined manifold is a PL-manifold, thus preserving the structure of the combinatorial dynamical system.*

Remark 4. *The integrator action corresponds to the time it takes to traverse the transition. In this setting however it is not further considered, since the approach taken here is time abstract.*

In the following a combined algorithm will be presented, which takes the original hybrid system, transforms it into a combinatorial hybrid system and finds a trajectory of the system satisfying the given requirements.

Algorithm 1. *Algorithm for going from a continuous hybrid system to control on a combinatorial hybrid system.*

1. *Construct connecting manifolds. One joined manifold is build for the entire hybrid system using Definition 20.*
2. *Triangulate manifold. The joined manifold is now triangulated. This can be done in a number of ways and with larger or smaller discretisation [Goodman, 2004].*
3. *Barycentric linearization. In order to obtain a piecewise affine system in each simplex the dynamics in each simplex is linearized around its barycenter.*
4. *Calculate controllability. Calculate all possible exit faces of the given simplex. This results in a combinatorial control system.*
5. *Find shortest path. Once the combinatorial vector field is in place it is possible to calculate all possible V-paths and thereby finding the shortest.*
6. *Ensure path is followed. Finally the controllers needed to ensure that the desired path is followed are selected.*

This algorithm is obviously not optimal, since it calculates all possible ways to traverse the simplicial complex before returning the shortest one. Thus practically the first two points in the algorithm will be performed first. Following this the next 3 points will be performed iteratively. This is done by taking the initial n-simplex, linearize it around its barycenter and calculate all the possible n-simplices reachable from this simplex. This is then repeated for the resulting simplices of the previous operation until the target simplex is reached. In order to avoid loops, any simplices, which previously have been visited are omitted from the new set of simplices to be checked. This operation can be seen as a tree search algorithm, and in order to find the shortest path a breath-first search algorithm will be preferable compared to a depth-first algorithm.

Once the traversing reaches the target simplex the algorithm is terminated, and the shortest possible V-path from the initial to the target simplex, γ_{op}, is found.

The supervisory control task now is to make sure that γ_{op} is followed, which is ensured by blocking undesired exit facets of simplices with more than one exit facet. This is practically achieved by altering the flow lines in the continuous system through control, as described in Section 7.3.

Remark 5. *It is often desirable to address the problem of avoiding forbidden or unsafe sets. The advantage of the formalism developed in this paper is that there is no combinatorial vector field thus no flow defined on the forbidden simplices. Therefore branches of the search tree hitting such simplices will automatically be abandoned.*

To show how the algorithm is used an example will be given in the following.

Example 2. *The original continuous hybrid system under consideration in this example can be seen in fig. 7.5(a). It consists of 3 different squares, along with one transition relation from M_1 to M_2 and again from M_2 to M_3, which are indicated here as joining manifolds.*

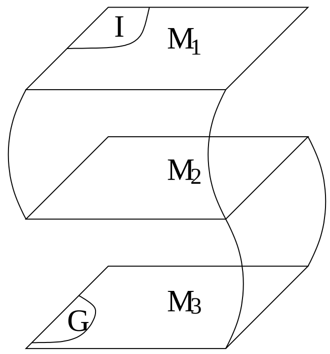
The first step according to Algorithm 1 is to build one joined manifold, as seen in fig. 7.5(a). Secondly the joined manifold needs to be triangulated. In this example a coarse triangulation is used to show the principle. This can be seen in fig. 7.5(b).

Now, for each triangle the system dynamics is linearized around its barycenter forming an affine system. On each of these affine systems the controllability of the system is calculated, thus given all possible directions to exit a given simplex. All these possibilities are shown in fig. 7.5(c).

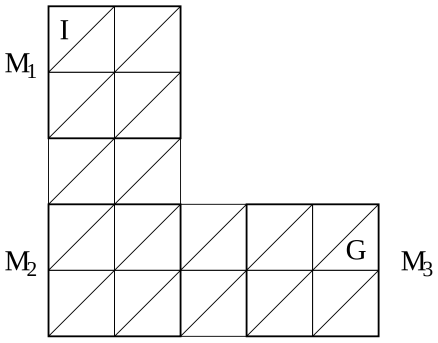
The final two steps in the algorithm are now first to find a suitable route of the combinatorial dynamical system. One possible objective could be to minimize the number of traversed simplices. Finally, when the desired route has been found the necessary controllers to ensure that the route is followed are selected.

7.6 Conclusion

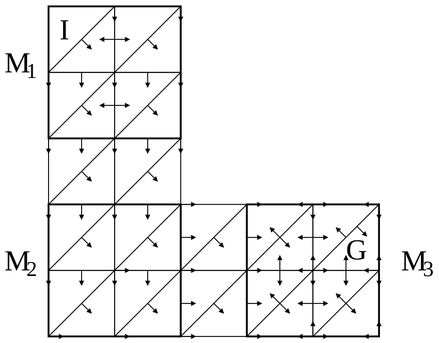
Through this paper the concept of a combinatorial hybrid systems defined on a simplicial complex has been introduced. Concepts from ordinary dynamical systems, such as flow, flow lines and Lyapunov functions have been shown for combinatorial hybrid systems. Furthermore an algorithm for automatic control design has been suggested along with examples showing how the proposed methods are used.



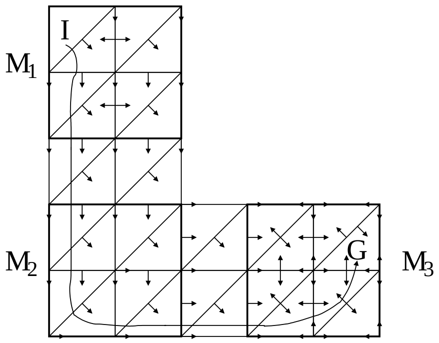
(a) Original Hybrid System



(b) Combined and triangulated manifold



(c) Maximum combinatorial vector field



(d) Combinatorial Hybrid System

Figure 7.5: Illustration of Combinatorial hybrid system

Chapter 8

Combinatorial Fault Tolerant Control System Design - with Application to Attitude Control Systems for Satellites.

This work presents a novel method for designing automated fault tolerant and fault probability minimizing control by abstracting the dynamics of the system, in both nominal and faulty mode, into a finite number of discrete partitions. On this discrete abstraction it is possible to analyze the system through the discrete equivalents of vector fields and flows, and to design controllers given a set of control objectives. The key feature is now, that all considered faults are embedded into the discrete object, and that it is possible to automatically design controllers for it. The theory is shown applied to the attitude control system for a small satellite.

8.1 Introduction

Autonomous satellite missions is an area which is gaining increasingly attention, since the complexity of future missions is increasing, and the stringent budget demands calls for less permanent ground operators. A number of missions have since the turn of the millennium showed that it is indeed possible to run such complex space missions nearly autonomous, with a minimum of operator intervention. Of interesting missions could be mentioned NASAs Deep Space 1[Pell *et al.*, 2004] , ESAs SMART-1[Elfving *et al.*, 2003] and most recently the Automated Transfer Vehicle (ATV)[McInnes, 2000] by ESA, which demonstrated autonomous rendezvous and docking with minimal operator intervention. The basic requirements to a space vehicle for such missions is that it include a system which is capable of detecting faults, and a supervisory controller, which is in charge of directing the satellite according to the mission objectives and any possible faults that may have occurred. It is the latter part which is the focus of this paper.

The supervisory controller can in many cases be modeled as a switched dynamical system. Through resent years a number of initial steps have been taken towards truly automatic control of switched dynamical systems, as seen in the works of [Tabuada and Pappas, 2003b; Bemporad *et al.*, 2000] and the references therein. Giving a system the ability to automatically reconfigure itself has received enormous attention during the past decades, particularly in the fields of fault tolerant control (FTC). The problem arises whenever a change occurs, the system will start to either recalculate a new control law, or select it from a set of pre-analyzed and -programmed controllers. Albeit doable, such reconfigurations requires great computational resources; recourses, which are often not available on smaller autonomous platforms like satellites, thus a more efficient method is desired. Previously, in the FTC case each fault scenario had a dedicated set of controllers. The switching between these controllers have then been left to a supervisory controller, which relies on reliable fault detection and isolation (FDI) methods to decide when to switch[Izadi-Zamanabadi and Larsen, 2007] .

In the present effort the problem of automatically designing a control system under the presence of possible faults has been addressed. This paper deals with combinatorial formulation for piecewise-affine control systems. It is thought as a carry-over of the work of[Habets and van Schuppen, 2004; Habets *et al.*, 2006] , which address the control problem for piecewise-affine systems on an arbitrary polytope that forces the solution trajectories of the closed loop system to either leave it or stay in it forever. Reachability and controllability of such systems have been studied before in[Asarin *et al.*, 2000a; Bemporad *et al.*, 2000] . Whereas previous methods have been based on the concept of a transition system, this paper focuses on its higher dimensional generalization, a simplicial complex. In 2 dimensions a simplicial complex is a set of triangles glued together along the edges. Likewise in 3 dimensions it is build from tetrahedrons.

A simplex is to be thought of as a generalized n-dimensional triangle. The simplices are then used to cover the system's state space. If the system evolves in a bounded 6-dimensional space, then this state space would be covered with a finite number of 6-simplices.

For each simplex in the simplicial complex it is now possible to analyze which adjacent simplices are reachable from the said simplex. By doing such an analysis for all simplices in the simplicial complex it is possible to design a controller, which is able to move the system from one position in the state space to another.

This kind of method bears resemblance to the work done on timed automata within the computer science community. However, there are a number of distinct differences between the possibilities using timed automata and the method presented here. First and foremost the presented method here also considers lower dimensional flows on the edges of the simplices. Secondly a number of standard formulations from continuous dynamical systems, such as: vector field, flows and Lyapunov functions are shown to have discrete equivalents in the notion of a combinatorial dynamical system defined on a simplicial complex. By using this formulation it is now possible to automatically design a control strategy for reaching a certain goal region in the simplicial complex. A small example of such can be seen in figure 8.1. The arrows indicate the flow of the system, and they always go from an n-simplex to an (n+1)-simplex. Thus flow from a point runs to a line and from a line to a surface. It is seen in figure 8.1 that the system will end up in the corner v_3. The system is forced to follow the prescribed direction through an affine controller for each simplex.

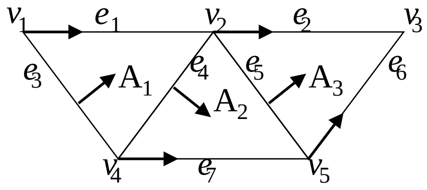


Figure 8.1: An example of the combinatorial abstraction of a state space. The arrows on the edges illustrate combinatorial vector fields, which bears resemblance to vector fields from continuous systems.

Having a controller in the nominal case it is possible to consider how to include possible faults in the formulation. When having the combinatorial dynamical system, then a number of different possible trajectories, with different controllers arise to the same control objective, thus by utilizing health information about all the instruments on the satellite, it is possible to always choose the control strategy, which poses the smallest probability of instrument failure during the maneuver. This phenomena becomes particularly important when performing control near critical regions, where a fault in a sensor or actuator could result in entering a critical region thus leading to mission failure.

In this paper the design procedure for a fault-tolerant control system will be recalled in the following section. After this the theory behind combinatorial dynamical systems defined on simplices will be presented along with how most concepts from classical control theory applies to it. Finally it will be shown how the theory applies to automatically designing a fault tolerant control system for a satellite, which in this case is a micro class satellite using momentum wheels.

8.2 Fault-Tolerant Control System Architecture

Design of a complete fault tolerant supervisory control involves a number of activities. It requires methods that can help the designers to rigorously analyze the system, identify all possible/potential faults, identify the monitoring/diagnosis possibilities, design control and sensor fusion algorithms for different scenarios, design the dedicated decision logic to ensure correct decision when an event has occurred and then take the appropriate action to accommodate for the situation.

As it is illustrated in figure 8.2 a modular architecture for implementation of the fault tolerant control system is employed. The detector modules monitor the system and when

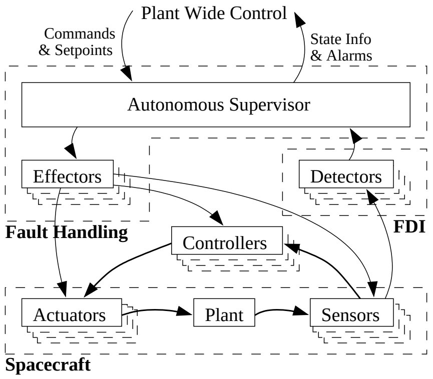


Figure 8.2: An overview of fault tolerant control system architecture.

a fault occurs the supervisor is informed. Based on the received information from detector modules and/or operator, the supervisor switches to appropriate state. The effector modules translate the new state and carry out the necessary changes (including changing the control strategy or activating proper sensor fusion). The modularity provides both flexibility when changes are needed and also less complicated testing procedures.

As shown on Figure 8.2, the spacecraft could be considered as a complex system with different sensors, actuators and controllers. Due to the system complexity as described previously it is necessary to apply a structured way of analysing the system in steps, as shown in figure 8.3 from [Bøgh, 1997] . The content of the individual blocks will be described in the following.

8.2.1 Fault Analysis

In the fault modeling step the system is divided into components and each component is analyzed for possible fault using a proper Hazard analysis technique. For satellites, being electro-mechanical systems, the Failure Mode and Effect Analysis (FMEA) technique is appropriate[International, 1998] . The output of this analysis is a number of possible faults, which should be considered in the following.

In the fault assessment step all the faults identified using the FMEA are assessed according to their severity to the space craft mission and their probability of occurrence, which leads to a Severity Occurrence (SO) index. Each severity is quantized with a value

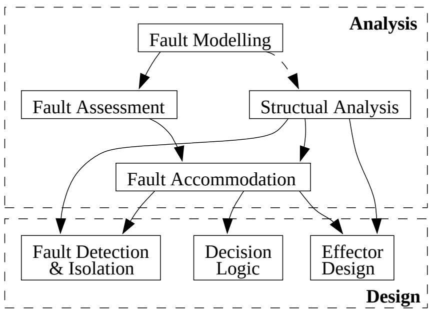


Figure 8.3: Systematic fault tolerant control system development approach.

between 1 and 10, where 10 is the highest rating, i.e. very severe. Likewise occurrences are quantized in the interval from 1 to 10, where 10 is very likely, and 1 is highly unlikely [International, 1998] . The SO index is now obtained by multiplying the two numbers for each fault, thus giving a measure suitable for discriminating which faults should be dealt with, and which are minor (either low severity or low occurrence). The severity of the considered faults are assessed either through fault injection on the actual plant or through fault simulation on a model of the plant. Since the system at hand is a satellite, it is not possible to do fault injection, thus all the faults needs to be simulated.

The structural analysis is used to identify the parts or subsystems of the system at hand which contain redundant information. A general framework for this is the structural approach[Declerck and Staroswiecki, 1991; Izadi-Zamanabadi and Staroswiecki, 2000; Blanke *et al.*, 2003] . The redundant information of the system can then be analyzed and used for identifying and diagnosing faults. It is worth noticing, that the structural model of the system does not depend on detailed knowledge of the plant parameters or dynamic relations, only between the system variables and constraints (i.e. differential equations, algebraic equations, rules), thus enabling the method to be applied very early in the system design phase. The analysis is always performed on the nominal system (i.e. without faults), thus it also makes it suitable for identifying possibilities for sensor fusion. For further information about the method see [Izadi-Zamanabadi, 2002] .

It is now possible to design the fault accommodation strategies. This involves developing the strategies for handling the selected faults in all operational mode by evaluating each subsystem with its instrumentation with respect to possible redundancies in order to meet the desired requirements. In the case of sensor faults, the fault accommodation strategy is normally based on the sensor fusion possibilities and takes advantage of the redundant information found through the structural analysis. In the case of actuator faults, these are mostly handled through controller reconfiguration or by taking advantage of hardware redundancy.

These four elements constitute the analysis phase of fault tolerant control design. They provide a comprehensive method for analyzing the system with focus on providing fault tolerance. The final output of the analysis phase is a list of severe faults that needs to be detected and handled and a clear view of the means by which they should be handled.

8.2.2 Fault Tolerance Design

The next phase in developing a fault tolerant control system is the design phase. This phase is again divided into three parts:

- Fault Detection and Isolation. Design of a number of algorithms to correctly detect identified faults.
- Decision Logic. Development of a decision logic, which react to the possible faults, or commands and determine a corresponding logical state.
- Effector Design. Algorithms or procedures, which, given the new state, provide the required corresponding functionality.

Depending on the redundancies in hardware and software, and the underlying systems dynamics complexity a number of methods and algorithms for fault diagnosis purposes exist. The fault detection and isolation usually benefits from the structural analysis by taking advantage of the redundant information available, and employing model-based methods such as observer-based or parity space methods[Chen and Patton, 1999] . The output of this block is normally a fault vector $\vec{F} = \{F_{nom}, F_{e_1}, F_{e_2}, \ldots, F_{e_k}\}$ where $F \in \{0, 1\}$. However, in this contribution we will consider the actual fault probabilities directly, giving by the fault probability vector $\vec{F_p} = \{P_{nom}, P_{e_1}, P_{e_2}, \ldots, P_{e_k}\}$ where $P \in [0, 1]$.

On the basis of the output from the FDI system the decision logic decides which actions should be taken in order to maintain fulfillment of the mission mode objectives in the future. For each mission mode different control modes can be defined reflecting the set of sensors and actuators available by which the given mission modes objectives are achievable albeit degrade performance.

The last step in the procedure is the effector design. This is the actual design of procedures and algorithms for performing the desired mission objectives given a certain set of instruments. In the case of sensor faults, it is often possible to handle the fault through sensor fusion, whereas actuator faults normally relies on hardware redundancy.

Having described the procedure for producing the fault tolerant control system it is now needed to describe how the dynamics of the system is abstracted into a combinatorial dynamical system, on which the final control system will be designed.

8.3 Discrete Abstraction

To arrive at a discrete abstraction of the dynamical system in question the system dynamics needs to be divisions into a number of objects, which later on will be used in the design process. In the following section some basics of doing the partitioning will be discussed along with how to define a dynamical system on the discretized objects. For a thorough handling of how to perform automatic control synthesis on discrete dynamical systems, see [Wisniewski and Larsen, 2008; Larsen and Wisniewski, 2008a] .

8.3.1 State Partition

As the main object of interest in the following will be on compact manifolds[Lee, 2002] and partitions of these into simplices the formal definitions of these will be stated in the following.

Definition 21. *A polyhedron is a set that equals the intersection of a finite number of closed half spaces.*

Definition 22. *An n-simplex is a set that equals the intersection of $n + 1$ closed half spaces.*

The following definitions for a partition of a given set is used.

Definition 23. *A collection of polyhedral sets $K = \{\theta_1, \dots, \theta_n\}$ is a partition for a set Θ if*

1. $\bigcup_{i=1}^{N} \theta_i = \Theta$
2. *If $\theta \in K$ and γ is a face of θ, then $\gamma \in K$.*
3. *If $\gamma = \theta \cap \phi$ then γ is a face of θ and ϕ.*

In the following we will exclusively be focusing on simplicial partitions. We denote the set of all n-simplices in a simplicial complex K by K_n,

$$K_n \equiv \{\sigma \in K |\ \sigma \text{ is an } n\text{-simplex}\}.$$

On each partition a dynamical system will be defined in terms of a piecewise affine (PWA) approximation of the original system.

Definition 24. *A function $f : \Theta \to \mathbb{R}^k$ where $\Theta \subseteq \mathbb{R}^k$, is simplicial PWA if there exist a simplicial partition, $R_1, \dots, R_N$ of Θ and $f(\theta) = A^i\theta + B^iu + a^i, \forall\theta \in R_i, i = 1, \dots, N$*

Thus for each simplex the possible control possibilities can be studied through the corresponding n-control system for that simplex.

Definition 25. *Let σ be an n-simplex. We say that a control vector field $\xi : \mathrm{Im}(\sigma) \times \mathbb{R}^m \to \mathbb{R}^n$ is a piecewise-affine n-control system if ξ is defined by the piecewise-affine map*

$$\xi(x, u) = Ax + Bu + a,$$

where A is an n by n matrix, B is an n by m matrix and a is an n-vector.

8.3.2 Discrete Dynamical System

Having the domain of the dynamical system in question given as a simplicial complex it is now possible to define a discrete dynamical system on top of it as an abstraction of the original system. Firstly, some important properties of simplices will be recaptured from a geometric point of view followed by the definition of control systems on these objects. This leads up to the main contribution in this section: the combinatorial vector field, which shares many properties with ordinary vector fields as known from dynamical systems.

Definition 26 (Definition IV.1.1 in [Bredon, 1997]). *Let $\mathbb{R}^n$ have standard basis $e_0, ..., e_n$. Then* ***standard*** ***n-simplex*** *is*

$$\triangle_n \equiv \left\{ x = \sum_{i=0}^{n} \lambda_i e_i \,\middle|\, \sum_{i=0}^{n} \lambda_i = 1,\ 0 \leq \lambda_i \leq 1 \right\}.$$

The λ_i are called **barycentric coordinates**.

Definition 27 (Definition IV.1.2 in [Bredon, 1997]). *Given $n \leq N$ independent points $v_0, ..., v_n \in \mathbb{R}^N$, $[v_0, ..., v_n]$ is an affine map $\triangle_n \to \mathbb{R}^N$ defined by*

$$\sum_i \lambda_i e_i \mapsto \sum_i \lambda_i v_i.$$

We shall call $[v_0, ..., v_n]$ an ***affine*** ***n-simplex*** *or just a simplex.*

The image of $[v_0, ..., v_n]$ is the convex span of the points v_i. We shall often identify an (affine) simplex with its image

$$\text{Image}[v_0, ..., v_n] = \left\{ \sum_{i=0}^{n} \lambda_i v_i \,\middle|\, \sum_{i=0}^{n} \lambda_i = 1,\ 0 \leq \lambda_i \leq 1 \right\}.$$

This definition is equivalent to the corresponding definition based on the intersection of $n + 1$ half-planes. For the n-simplex there will be $n + 1$ outward pointing normal vectors, $\bar{n}_0, \ldots, \bar{n}_n$, thus the simplex can be described by the set of points satisfying the following LMI, with v_i being the only point not in the face with $\bar{n}_i$ as a normal vector:

$$\Lambda x \leq \beta, \text{ with } \quad \Lambda = \begin{bmatrix} \bar{n}_0^T \\ \vdots \\ \bar{n}_n^T \end{bmatrix}, \quad \beta = \begin{bmatrix} \bar{n}_0^T v_1 \\ \vdots \\ \bar{n}_{n-1}^T v_n \\ \bar{n}_n^T v_0 \end{bmatrix}$$

Definition 28 (Definition IV.1.5 in [Bredon, 1997]).
*The n-****chain group*** *$C_n(K)$ of the simplicial complex K is the free abelian group generated by n-simplices*

$$C_n(K) = \bigoplus_{\sigma \in K_n} \mathbb{Z}_2,$$

which is equivalent to $\mathbb{Z}_2[\sigma_1] \times \mathbb{Z}_2[\sigma_2] \times \ldots$ where $\sigma_i \in K_n$. Thus an n-chain is a formal sum

$$c = \sum_{\sigma \in K_n} n_\sigma \sigma$$

of n-simplices σ with integer coefficients n_σ.

This can be understood as, that C_n consists of all possible linear combinations of simplices in K_n. An example of this would be if $K_n = \{a,\ b,\ c\}$, then $C_n = \{0,\ a,\ b,\ c,\ a+b,\ a+c,\ b+c,\ a+b+c\}$.

Furthermore, the boundary map of a simplex, denoted by ∂, is defined by systematically omitting one vertex from the simplex in question and then adding the result together with alternating sign as shown below, where $[a,b,c]$ means span$\{a,b,c\}$.

$$\partial[a,b,c] \equiv [\hat{a},b,c] - [a,\hat{b},c] + [a,b,\hat{c}] = [b,c] - [a,c] + [a,b].$$

Notice the notion of direction: $[a,b] = -[b,a]$.

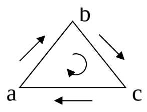


Figure 8.4: Example of boundary map calculation.

We introduce a relation on the set of simplices: $\gamma \prec \sigma$ if γ is a face of σ. We say that γ is a maximal face of σ if $\gamma \prec \sigma$ and $\dim \gamma + 1 = \dim \sigma$.

Now it is possible to formulate the notion of a combinatorial manifold M - a simplicial complex of particularly regular structure. We associate to each simplex of maximal dimension in M a piecewise affine control system.

Definition 29. *An **combinatorial** n**-control system** is a pair (K,ξ), where $K = \{K_0, ..., K_n\}$ is a simplicial complex, and $\xi = \{\xi_\sigma |\ \sigma \in K_n\}$ is a family of piecewise affine n-control systems.*

Let (K,ξ) be a combinatorial n-control system. A control objective for (K,ξ) is decomposed in [Habets and van Schuppen, 2004; Habets *et al.*, 2006] into two control problems posed for each n-simplex σ (in fact in [Habets and van Schuppen, 2004] the authors treat more general problem of control synthesis on a polytope):

Problem 5 (Problem 4.1 in [Habets *et al.*, 2006])**.** *Let $\sigma \in K_n$. Given a subset S of maximal faces of σ find a control law*

$$k_\sigma : \mathrm{Im}(\sigma) \to \mathbb{R}^m,\ k_\sigma(x) = F_\sigma x + g_\sigma, \tag{8.1}$$

where F_σ is an m by n matrix and g_σ is an n-vector, such that it guarantees that all flow lines of the closed-loop system

$$\dot{x} = (A + BF)x + (a + Bg), \tag{8.2}$$

starting at a $p \in \mathrm{Im}(\sigma)$ leaves the simplex σ in finite time by crossing one of the faces in S.

Problem 6 (Problem 4.2 in [Habets *et al.*, 2006])**.** *For a given $\sigma \in K_n$ find a control law (8.1) such that for any $p \in \mathrm{Im}(\sigma)$ the flow line $\phi_p(t)$ of the closed-loop system (8.2) satisfies $\phi_p(t) \in \mathrm{Im}(\sigma)$ for any $t \geq 0$.*

We say that the control law (8.1) **blocks** a maximal face γ of a simplex σ if the vector field ξ^c_σ of the closed loop system on σ - defined by the right hand side of equation (8.2) - satisfies the equality

$$\langle \xi^c_\sigma(x), n_\gamma \rangle \leq 0 \tag{8.3}$$

for any $x \in \mathrm{Im}(\gamma)$, where n_γ is the outward normal vector to τ and $\langle \cdot, \cdot \rangle$ is the standard scalar product on $\mathbb{R}^n$. Inequality (8.3) indicates that γ is an **exit face**.

Problems 5 and 6 are solved in [Habets *et al.*, 2006] by blocking maximal faces that are complementary to the set S. We observe that if $S' \subset S$ and the control law k_σ blocks all the faces in S then it also blocks the faces in S'; thus the more blocking faces the more restrictive control it is.

The focus in this work is on the combinatorial part of the control synthesis problem, i.e. on a supervisor that selects blocking faces of a combinatorial n-control system such that every trajectory of the closed loop system starting in an n-simplex σ_s reaches the target n-simplex σ_t in finite time. For a treatment of the necessary and sufficient conditions for guaranteeing control to a certain facet of a simplex for the PWA system the reader is referred to [Habets *et al.*, 2006] and the references therein.

Combinatorial Vector Fields

Here we introduce the central notion of this section - a combinatorial vector field. The notion has been developed by R. Forman in [Forman, 1998] for studying topological invariants of CW complexes. The attention in this section is restricted to geometrical properties of a combinatorial vector field. We extend the notion of a combinatorial vector field to encompass non-determinism in Definition 32. It is treated as a generator of flow.

Definition 30 (Definition 1.2, [Forman, 1998])**.** *Let K be a simplicial complex. A* ***combinatorial vector field*** *V on K is a family $\{V_n |\, n \in \mathbb{N}\}$ of maps*

$$V_n : K_{n-1} \to K_n \cup \{0\}$$

that satisfies

1. *$V_n \circ V_{n-1} = 0$, that is if $\sigma \in \mathrm{Image}(V_{n-1})$ then $V_n(\sigma) = 0$.*
2. *For each $\sigma \in K_n$, the number of elements of the pre-image*

$$V_n^{-1}(\sigma) \equiv \{\alpha \in K_{n-1} |\, V_n(\alpha) = \sigma\}$$

   is 0 *or* 1.

Alternatively a combinatorial vector field is a set $\bar{V}$ of pais of simplices $\langle \alpha, \sigma \rangle$, where α is a maximal face of σ, and for which no simplex is in more than one pair. It is helpful to picture a combinatorial vector field on K by arrows, where the tail is at α and the arrow at σ, see Figure 8.5.

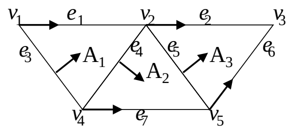


Figure 8.5: A combinatorial vector field. The vertex v_3 is a rest point.

Intuitively condition 1. of Definition 30 means that the system is of the first order; geometrically it implies that the future simplices do not increase the dimension, see the definition of the flow map below. Condition 2. excludes merging the future cells. It is illustrated in Figure 8.6 that splitting of flow is allowed whereas merging is excluded. In Definition 32 of a nondeterministic combinatorial vector field we shall allow both situations.

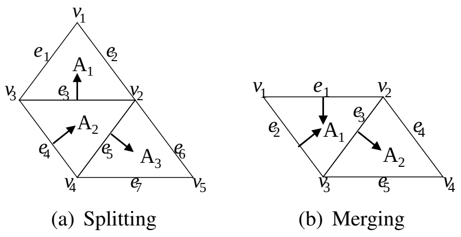


Figure 8.6: Illustration of Definition 30. Merging on the right hand side is excluded, whereas splinting on the left hand side is allowed.

Since no simplex is in more than one pair in $\bar{V}$, every cell σ of the simplicial complex K satisfies precisely one of the following conditions:

1. σ is the tail of exactly one arrow;
2. σ is the head of exactly one arrow;
3. σ is neither the tail not the head of any arrow.

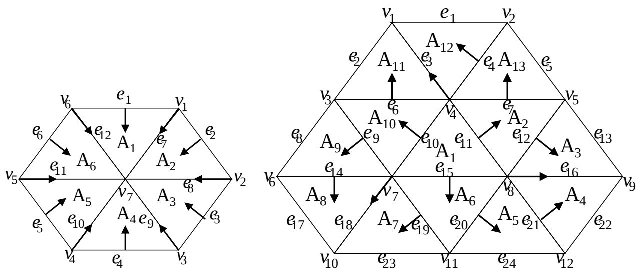
A simplex that satisfies condition 3. is called a rest point.

Definition 31 (Definition 1.3 of [Forman, 1998]). *Let V be a combinatorial vector field on K. We say that $\sigma \in K_n$ is a **rest point** of V of index n if*

1. $V_{n+1}(\sigma) = 0$ *and*

2. $\sigma \notin \mathrm{Image}(V_n)$.

Figure 8.7 illustrates a rest point of minimal index 0 - a sink and the maximal index n - a source.



v_1 v_2 v_3 v_4 v_5 v_6 v_7
e_1 e_2 e_3 e_4 e_5 e_6 e_7 e_8 e_9 e_{10} e_{11} e_{12}
A_1 A_2 A_3 A_4 A_5 A_6
v_1 v_2 v_3 v_4 v_5 v_6 v_7 v_8 v_9 v_{10} v_{11} v_{12}
e_1 e_2 e_3 e_4 e_5 e_6 e_7 e_8 e_9 e_{10} e_{11} e_{12} e_{13} e_{14} e_{15} e_{16} e_{17} e_{18} e_{19} e_{20} e_{21} e_{22} e_{23} e_{24}
A_1 A_2 A_3 A_4 A_5 A_6 A_7 A_8 A_9 A_{10} A_{11} A_{12} A_{13}

(a) Sink (b) Source

Figure 8.7: (a) v_7 is a rest point of index 0; (b) A_1 is a rest point of index n.

Since piecewise linear control systems indicates that discrete behavior of a piecewise affine control system involves nondeterminacy induced by blocking more than one maximal faces of a simplex. It seems therefore natural to unleash condition 2 of Definition 30.

Definition 32. *Let K be a simplicial complex. A **nondeterministic combinatorial vector field** V on K is a family $\{V_n | \; n \in \mathbb{N}\}$ of maps*

$$V_n : K_{n-1} \to K_n \cup \{0\}$$

that satisfies $V_n \circ V_{n-1} = 0$.

In the remaining of this section we shall develop a notion of flow of a nondeterministic combinatorial vector field, that is a map $C_n(K) \to C_n(K)$ which takes an n-simplex to its future n-chain (a linear combination of the simplices in very next future).

Remark 6. *The linear combination of simplices indicates nondeterminism in the future evolution. Thus for example $\tau \mapsto \sigma + \beta$ means that the future of τ is σ or β.*

For simplicial complexes we may use the following definition of a **combinatorial scalar product** $\langle \cdot, \cdot \rangle : K_n \times K_n \to \{0, 1\}$, defined by

$$\langle \sigma, \alpha \rangle = \begin{cases} 1 & \text{if} \quad \sigma = \alpha \\ 0 & \text{otherwise} \end{cases}$$

We extend it to the bilinear product $\langle \cdot, \cdot \rangle : C_n(K) \times C_n(K) \to \mathbb{Z}$. In particular

$$\left\langle \sum_j n_j \sigma_j, \sigma_k \right\rangle = n_k.$$

Define $\theta_n : C_n(K) \to C_{n-1}(K)$ by

$$\theta_n(\sigma) = \sum_{i=0}^{n} (-1)^i \langle V_n \circ d_i^n(\sigma), \sigma \rangle \, d_i^n \sigma.$$

The map θ takes $\sigma \in K_n$ to a linear combination of the simplices in $V_n^{-1}(\sigma)$, see Figure 8.8.

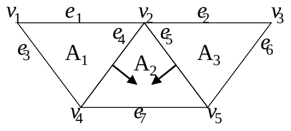


Figure 8.8: $\theta_2(A_2) = e_4 - e_5$.

Discrete dynamics of a combinatorial control systems is encapsulated in the following definition of the flow.

Definition 33. *A **flow** (of a nondeterministic combinatorial vector field) is the map Φ_n : $C_n(K) \to C_n(K)$ given by*

$$\Phi_n = (\partial_{n+1} - \theta_{n+1}) \circ V_{n+1} + V_n \circ (\partial_n - \theta_n).$$

Example 3. *Consider the nondeterministic combinatorial vector field defined in Figure 8.9.*

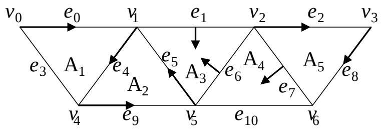


Figure 8.9: An example of a simplicial complex with its associated vector field.

Firstly, we compute flow starting at e_0:

$$\begin{aligned} \Phi_1(e_0) &= (\partial_2 - \theta_2)V_2(e_0) + V_1(\partial_1 e_0 - \theta_1 e_0) \\ &= (\partial_2 - \theta_2)0 + V_1(v_0 - v_1 - v_0) = -e_4. \end{aligned}$$

The result is the 1-simplex e_4, which corresponds to our expectation, as seen from Figure 8.9.

Another flow of interest is the one initiated at e_1:

$$\begin{aligned} \Phi(e_1) &= (\partial_2 - \theta_2)V_2 + V_1(\partial_1 - \theta_1) \\ &= (\partial_2 - \theta_2)A_3 + V_1(v_1 - v_2) \\ &= e_1 - e_6 + e_5 - e_1 + e_6 - e_2 = e_5 - e_2. \end{aligned}$$

Again it is seen that the resulting flow gives the foreseen adjacent edge, e_1, along with a possible side flow to e_2.

Lastly, we calculate the flow from the 2-simplex A_4:

$$\begin{aligned}\Phi_2(A_4) &= (\partial_3 - \theta_3)V_3 + V_2(\partial_2 - \theta_2)\\ &= (\partial_3 - \theta_3)0 + V_2(e_7 - e_{10} + e_6 - e_7) = A_3.\end{aligned}$$

Again the flow from A_4 to A_3 is found as expected.

The flow Φ_n generates an n-flow line. An n-simplex $\sigma \in K_n$ belongs to the n-flow line with the initial n-simplex τ if there is $k \in \mathbb{Z}_+$ such that $\langle \sigma, \Phi_n^k(\tau)\rangle \neq 0$. It will be seen below that the flow lines of dimension n and $n-1$ are the only important for control synthesis for combinatorial control systems. It is worth noticing that a flow line born in an n-simplex σ - a source - does not die in a sink, since it is a vertex (0-simplex). It dies in fact in an n-simplex belonging to star of a sink.

8.3.3 Lyapunov Function

With combinatorial vector fields defined, it possible to define the combinatorial counterpart of flow lines, which in the combinatorial setting is called a V-path.

Definition 34. *A V-**path** of index p is a sequence of length r,*

$$\gamma : \sigma_0^{(p)}, \tau_0^{(p+1)}, \sigma_1^{(p)}, \tau_1^{(p+1)}, ..., \tau_{r-1}^{(p+1)}, \sigma_r^{(p)}, \tag{8.4}$$

such that for all $i \in \{0, 1, ..., r-1\}$

1. $\tau_i = V(\sigma_i)$
2. $\sigma_i \neq \sigma_{i+1} = \Phi(\sigma_i)$

If $\sigma_0 = \sigma_r$ the V-path is called closed. Two closed V-paths, γ, $\tilde{\gamma}$, are equivalent if $\tilde{\gamma}$ can be produced by selecting another starting point of γ.

A V-path is calculated by taking an initial simplex, σ_0, and propagating its flow, i.e.:

$$\sigma_0 \to V(\sigma_0) \to \Phi(\sigma_0) \to V\Phi(\sigma_0) \to \Phi\Phi(\sigma_0)\ldots \tag{8.5}$$

Moreover, since non-determinism is allowed in this definition of flow the V-path is allowed to split into more paths, thus resulting in a tree of reachable locations compared to just a single track in the deterministic case.

Chain recurrent sets are sets in which the flow of an element of the set will be cyclic within the set. Thus intuitively this is the case for rest points and non-stationary closed V-paths. More formally this means that:

Definition 35 (Definition 2.1 in [Forman, 1998]). *$\sigma^{(p)} \in K$ is an element of the chain recurrent set $\mathcal{R}$ if either*

1. *σ is a rest point of V or*
2. *there is a non stationary closed V-path γ with $\sigma \in \gamma$*

As for dynamical systems, the notion of a Lyapunov function for a simplicial complex is desirable, and as shown in [Forman, 1998] and [Franks, 1980] it is possible to find a similar Lyapunov function, which has the property, that it is constant on the chain recurrent set, and outside the set it is the negative gradient of the function towards the set.

Definition 36 (Theorem 2.4 in [Forman, 1998]). *Let $\mathcal{R}$ be a chain recurrent set. There is a function $f : K \to \mathbb{R}$ such that*

1. *if $\sigma^{(p)} \notin \mathcal{R}$ and $\tau^{(p+1)} > \sigma$ then*
   $\begin{cases} f(\sigma) < f(\tau) \text{ if } \tau \neq V(\sigma) \\ f(\sigma) \geq f(\tau) \text{ if } \tau = V(\sigma) \end{cases}$
2. *if $\sigma^{(p)} \in \mathcal{R}$ and $\tau^{(p+1)} > \sigma$ then*
   $\begin{cases} f(\sigma) = f(\tau) \text{ if } \tau \sim \sigma \\ f(\sigma) < f(\tau) \text{ if } \tau \nsim \sigma \end{cases}$

where $\tau \sim \sigma$ means that that they belong to the same path.

This is to be understood in the following way: From the first definition, then a given V-path, not being a chain recurrent set,

$$\gamma : \sigma_0^{(k)}, \tau_0^{(k+1)}, \sigma_1^{(k)} \ldots$$

will have the following relation:

$$f(\sigma_0^{(k)}) \geq f(\tau_0^{(k+1)}) > f(\sigma_1^{(k)}) \geq \ldots,$$

which, as with continuous dynamical systems, means that the Lyapunov function is decreasing along the flow. The second condition in the definition above is saying, that for the chain recurrent set

$$\gamma_{\mathcal{R}} : \sigma_0^{(k)}, \tau_0^{(k+1)}, \sigma_1^{(k)}, \tau_1^{(k+1)}, \ldots, \tau_{r-1}^{(k+1)}, \sigma_r^{(k)} = \sigma_0^{(k)}$$

the following relation:

$$\begin{aligned} f(\sigma_0^{(k)}) = f(\tau_0^{(k+1)}) =& f(\sigma_1^{(k)}) = f(\tau_1^{(k+1)}) = \ldots = \\ & f(\tau_{r-1}^{(k+1)}) = f(\sigma_r^{(k)}) = f(\sigma_0^{(k)}), \end{aligned}$$

holds true.

Having the above definitions for the combinatorial equivalent of a Lyapunov function in place it is now possible to use the Lyapunov function as a specification of control on the simplicial complex.

One obvious Lyapunov candidate is based on the distance to the goal simplex. Assigning a value to each simplex in the simplicial complex is then done by iteratively assigning to every simplex a number starting from the goal simplex. Thus the goal simplex σ_g has associated a value, and all neighboring simplices to a goal are given a higher value than the goal value. Iteratively then all the neighboring simplices to these simplices which does not have a value yet will get a higher value assigned. Algorithmically, this is:

Procedure 1. *Iterative method for assigning to each simplex a Lyapunov value*

```
$\Delta_0 = \sigma_{goal}$, $V(\sigma_{goal}) = 0$
While $\Delta \neq K_n$
    $\rho = \{\sigma \in K_n | \sigma \notin \Delta \wedge \delta\sigma \cap \delta\Delta \neq \emptyset\}$
    $\forall \sigma \in \rho, V(\sigma) = \max(V(\Delta)) + 1$
    $\Delta_{i+1} = \Delta_i \cup \rho$
```

An example of this can be seen in figure 8.10. Here the goal simplex is denoted by G, and the initial simplex by I. The grayed out center cross is a forbidden region. By assigning Lyapunov values starting from the goal simplex and outwards it is ensured, that no local minimum occurs. Having this in place control can now locally always be determined by evaluating the Lyapunov value in the star of the current simplex and control towards the lowest one.

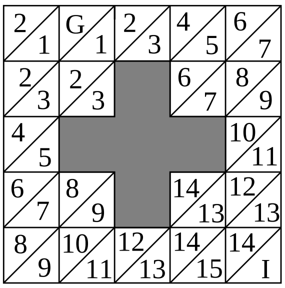


Figure 8.10: Example of Lyapunov value assignment to simplices in the Swiss flag example.

Another possibility would be to use a potential field kind of function[Hwang and Ahuja, 1992; Koren and Borenstein, 1991] with minimum at G and maximum at I, and then for each simplex utilize the average over the simplex as a lyapunov value for the simplex. The advantage of this would be, that it is possible to calculate the lyapunov value for an arbitrary simplex independent of the rest of the simplices. However it has the disadvantage, that it is possible to arrive at local minima, thus the gradient following method needs to be augmented to deal with such local minima.

8.3.4 Control Synthesis

From the section on combinatorial control systems we have, that there for each simplex will be a number of possible future simplices dependent on which controller is chosen in the specific simplex. That is:

Definition 37. *A control system is the map*

$$L_n : K_{n-1} \to 2^{C_n}, \tag{8.6}$$

where 2^{C_n} is the set of all subsets of C_n.

By having the control objective given through a Lyapunov function on the simplicial complex it is now possible to select the final control system by selecting controllers from L_n satisfying the Lyapunov requirements stated in Def. 36.

Definition 38. *The set of all possible control systems satisfying the control objective given by the Lyapunov function, f, is:*

$$L_n^c \equiv \{v^n \in L_n | f(\sigma) > f \circ v^n(\sigma)\}$$

Proposition 2. *If there exist a sequence of controllers for the system, s.t. $L_n^c \neq \emptyset$, then the combinatorial control system is said to be controllable wrt. the stated objectives.*

Remark 7. *It is worth noticing there, that given $L_n^c \neq \emptyset$ does not mean that it is impossible to control the system in question to the desired stated. It simply means, that the chosen simplicial division of the system might not allow for it.*

Looking at Definition 38 it is seen, that there might be more than one possible control system satisfying the Lyapunov inequality. This flexibility can then be used to select the controller, and thereby the V-path, which minimizes the cost in some sense. A typical minimization criterion in space missions is to minimize the consumption of propellant. However, near mission critical stages, another criterion, the probability of failure, is often sought minimized.

Remark 8. *It is worth noticing, that, since controllability to a given facet in the continuous case is independent of where the system is in the simplex or how it entered it, then the local decision on combinatorial controllability is not affected by previous selections.*

In practice the decision of which controller to use is a local decision, thus for each simplex the set of possible control systems is evaluated to find the next feasible controller. Thus for each simplex the following procedure is used:

Procedure 2 (Control Synthesis)**.** *Given the set of feasible control systems L_n^c, and the cost function, J, then synthesis of a controller obeying this specification is done by:*

```
$\sigma = \sigma_{\text{init}}$
*While* $\sigma \neq \sigma_{\text{goal}}$
    *Pick* $v_n^c \in L_n^c$ *s.t.* $\min J(\Phi^{v_n^c}(\sigma)$
    $\sigma = \Phi^{v_n^c}(\sigma)$
```

Where $\Phi^{v_n^c}(\sigma)$ means the flow of σ given the controller v_n^c. Here there are not put any restraints on the structure of J, it is also possible that instead of just evaluating one simplex it would also be able to handle a series of future simplices, thus resulting in a finite horizon optimization.

8.4 Case: The Rømer satellite

As an example of how the above described theory is used in practice it will be shown applied in the control system of the proposed danish satellite: Rømer.

The Rømer satellite is planned to carry two scientific experiments called the MONS (Measuring Oscillations in Nearby Stars) and the Ballerina experiment. Originally they were proposed as two separate missions, however they shared so many aims, that they were combined into one.

The satellite is being designed as a micro satellite with a mass of approximately 120 kg. This is to ensure, that the satellite can be launched as a secondary payload. The orbit specified for the Rømer satellite is a Molniya orbit.

The scientific objective of MONS is to observe stellar oscillations at a greatly improved level of sensitivity. A typical star is going to be observed for a period of 30-50 days continuously at a time. This is performed to probe the stellar interior to determine its composition, and and internal rotation. The primary instrument specified for the MONS mission is 340mm telescope with a CCD detector.

The scientific objective of Ballerian is the detection and localization of gamma-ray bursts (GRB). The physical mechanism leading to GRBs is poorly understood. The GRBs occur randomly and are distributed over the entire sky, and are known to be among the fastest objects in the universe. The scientific instrument specified for the Ballerina mission is an 80mm X-ray telescope. However, in order to first detect the GRBs Wide Angle Telescopes for Cosmic Hard x-rays (WATCH) telescopes are used. Four of such telescopes are placed on the satellite in a tetrahedron configuration, which ensures full sky coverage. After a GRB has been detected by one of the WATCH instruments, and localized at a precision of approximately 1 arc minute, the satellite turns autonomously within a few minutes to allow the X-ray telescope to observe the afterglow. The star imager and the X-ray telescope then determine the precise source of the burst, which subsequently is transmitted to the Earth.

Both the MONS and the X-ray telescope are highly sensitive, and exposure to direct light from the sun would destroy them immediately, thus the satellite is equipped with a sunscreen, which, after the satellite has been injected into its orbit, and it has entered into nominal mode, will unfold. After this point it is mission critical, that the satellite never will point directly at the sun.

The WATCH instruments have also been chosen in order to save both cost and weight, because they perform rotary motion. Since the scientific observations allow the instruments to have a varying angular frequency in the range $\pm[0.5,\,-2]$Hz, this gives the WATCH the dual purpose of both being a scientific instrument while at the same time being a part of the attitude control system. Along with the WATCH, acting as momentum wheels, the Rømer satellite is also equipped with magnetorquers, these are used to exchange momentum with the Earth when the satellite is near the Earth in its orbit, whereas only the momentum wheels are used further away.

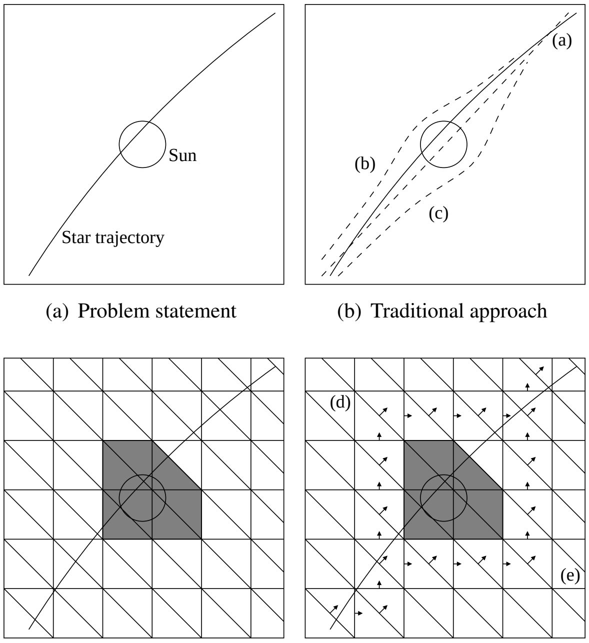
8.4.1 Fault Tolerant Control

Solutions for attitude control system of the Rømer spacecraft have been proposed in [Jensen and Wisniewski, 2001; Quattrup *et al.*, 2001] . In this paper the focus is on the supervisory control system. Moreover, it is on the decision process, which become evident, when the system is modeled as a discrete dynamical system, as described in the previous section. For the Rømer satellite in scientific mode the space in question is for the attitude, the unit quaternion, $S^3 \subset \mathbb{R}^4$ and for the angular velocity by the ball $B^3(2,0)rad/s \subset \mathbb{R}^3$. However, for ease of representation and understanding the following example will be given using a projection into $\mathbb{R}^2$, however it is clear that the theory is valid in any dimension.

One of the cases of particular interest for the Rømer satellite, is when it is close to pointing at the Sun. This situation arrises from the mission objective, which is to observe a given star for several days at a time, while, at the same time avoiding pointing at the Sun, which results in two conflicting objectives. Thus every time the target stars trajectory passes by the Sun, the supervisory controller needs to intervene in order no to point at the Sun.

This situation is depicted in figure 8.11(a). It is seen, that the observation bor axis passes through the Sun, thus would the supervisory system not intervene the satellite would continue its tracking across the sun leading to destruction of the onboard instruments and failure of the mission.

One standard approach, which is often used in guidance problems for autonomous systems is that of potential fields[Hwang and Ahuja, 1992; Koren and Borenstein, 1991] . This approach has also been proposed for the Rømer satellite in[Wisniewski and Kulczycki, 2005] . The basic idea of the method is shown in figure 8.11(b), where the straight dashed line denoted by (a) gives a division line, through the edge of the potential field. If the satellite is pointing to the left side of the line the satellite will follow the trajectory denoted by (b) around the sun, and had the satellite been pointing to the right side of the line it would have followed the trajectory denoted by (c). However, by introducing the potential field function for the Sun, unavoidably a choice has already been taking, saying if the satellite should choose trajectory (b) or (c). This choice is however made ”accidently” in the sense, that it is normally just taking without considering the consequence



(a) Problem statement (b) Traditional approach

(c) Combinatorial division (d) Combinatorial solution

Figure 8.11: Sun avoidance problem for the Rømer satellite

of the choice.

However, in the combinatorial setting, this choice is emphasized. The division of the space surrounding the sun is depicted in figure 8.11(c), where the simplices containing the sun have been marked grey, which means, that they should be avoided. In the last figure 8.11(d) the two possible V-paths, γ_d and γ_e, for tracking the star trajectory are shown. Up until reaching the avoid set the two paths are identical, however, at the last simplex before the two paths diverges the supervisor needs to perform a choice of which path to follow, either γ_d or γ_e. Looking closer at the two paths, then it is seen, that they are comprised of two parts, a critical part, which is from the splitting of the paths and until the corner, and a safe part, which is from the corner and until the trajectories merge again.

This idea is illustrated in figure 8.12, where the evolution of the V-path describing the star tracking is followed until time k, where a choice is to be made. As mentioned earlier, then this choice has often been taken implicitly a priori through the design of the avoidance algorithm. However, here, the choice is based on the fault probability infor-

$$\begin{tikzcd}
 & & \gamma_{d_{critical}} \arrow[r] & \gamma_{d_{safe}} \arrow[dr] & & \\
\gamma_{k-1} \arrow[r] & \gamma_k \arrow[ur, "0.01"] \arrow[dr, "0.07"'] & & & \gamma_{k+n} \arrow[r] & \gamma_{k+n+1} \\
 & & \gamma_{e_{critical}} \arrow[r] & \gamma_{e_{safe}} \arrow[ur] & &
\end{tikzcd}$$

Figure 8.12: Decision process for the Sun avoidance example

mation available. In this case the vector shows[1] $F_{P_k} = \{P_{nom} = 0.9, P_d = 0.01, P_e = 0.07, \ldots\}$, where P_d denotes the accumulated probability of a fault in the instrument set needed to perform $\gamma_{d_{critical}}$, and likewise for P_e. Having this information available, it is now possible to determine, which trajectory, γ_d or γ_e is the least probable to fail during the Sun avoidance maneuver, thus resulting in an overall higher mission reliability.

8.5 Conclusion

In this paper it has been shown how a given dynamical system can be represented as a combinatorial system. It has furthermore been shown how this formalism can be used in connection with fault tolerant control to increase the overall system reliability for critical control objectives. Finally it has been shown how the method applies to control of the Rømer satellite.

Normally faults have been treated by the supervisory control system when they arise, i.e. when they are too severe to continue nominal operation without switching to another control objective or another control strategy. The main feature of using projections of faults into the nominal simplicial complex is, that it becomes possible to make reasonable decisions based on a priori information about the satellites health state, thus making the supervisory controller proactive instead of reactive.

[1]The fault probabilities are accumulated for the two scenarios, and the remaining fault probabilities have been omitted.

Chapter 9

Combinatorial Vector Fields for Piecewise Affine Control Systems

The notion of a combinatorial dynamical system is rigorously introduced in this paper. A combinatorial dynamical system can be seen as a discrete abstraction of large class of systems including smooth non-linear systems, switched dynamical systems and switched control systems. A number of concepts from continuous dynamical systems are shown to have combinatorial counterparts, such as vector fields, flow lines and Lyapunov functions. Algorithms for automatic control synthesis are developed for combinatorial dynamical systems. Furthermore the presentation of these concepts is supported by a number of examples.

9.1 Introduction

During the past years a number of methods for automatic control synthesis have been proposed for dynamical systems and more recently for switched dynamical systems, as seen in the works of [Tabuada and Pappas, 2003b; Bemporad *et al.*, 2000] and the references therein.

Automatic control synthesis is particularly interesting in a number of fields including gain scheduling for control of non-linear systems, robotic navigation in changing environments and fault tolerant control.

Gain scheduling has received much attention in industry due to its intuitive methodology and ease of use. In its simplest form gain scheduling is performed by dividing the system space at hand into a number of sections, and then in each section finding a suitable controller for the system. An overview of gain scheduling methods can be found in [Rugh and Shamma, 2000], and more recent work on anti-windup and bump-less transfer between controllers can be found in [Zaccarian and Teel, 2002] and [Bendtsen *et al.*, 2005].

In robotic navigation the system's trajectory is to be designed. One of the main difficulties is, that the environment of the navigating robot can be non-static, thus new control strategies need to be derived online. An overview of the literature and examples of how to perform such online trajectory planning can be found in [Koenig and Likhachev, 2002] and the references therein.

Lastly, fault tolerant control systems makes great use of automatic control synthesis. This is not least true for space applications. Examples of actual space missions applying autonomously self-configuring controllers include missions such as NASAs Deep Space 1 reported by [Pell *et al.*, 2004] , ESAs SMART-1 described by [Elfving *et al.*, 2003] and most recently the Automated Transfer Vehicle (ATV) by ESA, see [McInnes, 2000], which demonstrated autonomous rendezvous and docking with minimal operator intervention.

This paper deals with a generalized framework for automatic control synthesis, thus encompassing the above mentioned fields into one unified framework. The paper can be seen as a carry-over and extension of the work of [Habets and van Schuppen, 2004; Habets *et al.*, 2006], which address the control problem for piecewise affine systems on an arbitrary polytope that forces the solution trajectories of the closed loop system to either leave it through a specified subset of facets or stay in it forever. Reachability and controllability of such systems have been studied in [Asarin *et al.*, 2000a; Bemporad *et al.*, 2000]. Whereas previous methods have been based on the concept of a transition system, this paper focuses on its higher dimensional generalization, a polyhedral complex. In 2 dimensions a polyhedral complex is a set of polytopes glued together along the edges. The polytopes are then used to cover the system's state space. Likewise, if the system evolves in a bounded n-dimensional state space, then it is covered by a finite number of n-polytopes.

As such systems potentially can be very large one of the main contributions in this paper is to rigorously derive an algebra for handling the complexity of such systems. Another very important feature of the theory presented here is that it is agnostic towards the underlying system, i.e. after the abstraction there is no distinction between a linear, non-linear or switched dynamical system.

For each polytope in the polyhedral complex it is now possible to analyze which adjacent polytopes are reachable from the said polytope. By doing such an analysis for all polytopes in the polyhedral complex it is possible to design a controller, which is able to move the system from one position in the state space to another.

This kind of method bears resemblance to the work done on timed automata. However, there are a number of distinct differences between timed automata and the method presented here. First and foremost the presented method also considers lower dimensional flows on the edges of the polytopes. Secondly a number of standard formulations from continuous dynamical systems, such as: vector field, flows and Lyapunov functions are shown to have discrete equivalents in the notion of a combinatorial dynamical system defined on a polyhedral complex. By using this formulation it is now possible to automatically design a control strategy for reaching a certain goal region in the polyhedral complex. Since the control strategy might give rice to a number of different solutions it is also possible to choose the optimal controller according to a given cost function.

Firstly, we define polytopes and polyhedral complexes in Section 9.3. A polyhedral complex is a division of the state space into polytopes. The possible control actions on each polytope makes the combinatorial flow either transversal or tangential to the facets of the polytope in question, which is introduced in Section 9.4. Having a control system on a polytope it is possible to establish a discrete abstraction of the vector field - a combinatorial vector field. The notion of a combinatorial vector field and the combinatorial equivalence of flow are introduced in Section 9.5. In Section 9.6 the concept of a Lyapunov function is defined for combinatorial vector fields, which is used as a specification for control objectives. The structure of the Lyapunov function is utilized in Section 9.7 for performing the control synthesis. Finally, the theory is illustrated by a numerical example in Section 9.8.

9.2 Preliminaries

Let $\langle\cdot,\cdot\rangle$ be the Euclidean scalar product in $\mathbb{R}^n$ and $||\cdot||$ the induced by $\langle\cdot,\cdot\rangle$ norm. The cardinality of a set K is denoted $|K|$. We shall denote the boundary of a set S by $\mathrm{bd}(S)$. The image of a map $f : U \to V$ is $\mathrm{Im}(f) = \{f(x) \in V |\, x \in U\}$.

9.3 State Partition

Let J be a finite index set. A **polyhedral set** P in $\mathbb{R}^n$ is the intersection of a family of closed half spaces $H_j^- \equiv \{x \in \mathbb{R}^n |\ \langle x, N_j\rangle \leq a_j\}$ for $N_j \in \mathbb{R}^n$ and $a_j \in \mathbb{R}$, where $j \in J$, i.e.

$$P \equiv \bigcap_{j\in J} H_j^- .$$

The polyhedral set P can be expressed by the inequality (9.1) to be understood components wise:

$$P = \{x \in |\ Nx \leq a\}, \tag{9.1}$$

where $N = \begin{bmatrix} N_1 & \dots & N_j \end{bmatrix}^{\mathrm{T}}$, $\alpha = \begin{bmatrix} a_1 & \dots & a_j \end{bmatrix}^{\mathrm{T}}$. The dimension of P is the dimension of the smallest subspace $V \subset E$ that contains P.

A subset F of a polyhedral set P is a **face** if either $F = \emptyset$ or $F = P$, or there exists a supporting hyperplane[1] H of P and $F = P \cap H$. Note that a face is again a polyhedral set. A face of co-dimension 1 (dimension $n-1$) is called a **facet**, a face of dimension 0 is a vertex. We write $F \preceq P$ to indicate that F is a face of P and $F \prec P$ if and only if $F \preceq P$ and $F \neq P$.

Let $K \equiv \{P_j |\ j \in I\}$ be a collection of polyhedral sets for some index set I. We define

$$|K| \equiv \bigcup_{j \in I} P_j \subseteq \mathbb{R}^n.$$

We say that K is a **(polyhedral) complex** if

1. $P \in K$ implies that any $F \prec P$ is also an element of K;
2. $P, Q \in K$ implies that $P \cap Q \prec P$ and $P \cap Q \prec Q$;
3. any point of $|K|$ has a neighborhood intersecting only finitely many elements of K.

Notice that if I is finite then the last condition is automatically satisfied. Observe that the intersection of two complexes is again a complex.

Let E be any polyhedral set ($\mathbb{R}^n$ inclusively). A **piecewise linear partition** (or for short partitioning) of E is a complex K such that $E = |K|$. The elements of K will be called **cells**.

We define sets $K_0, \ldots, K_n$ by

$$K_i \equiv \{P \in K |\ \dim(P) = i\}.$$

Thus, for instance K_n is the set of cells with full dimension n, and K_0 the set of points. Interchangeably by partitioning K we will understand the resulting $(n+1)$-tuple $(K_0, \ldots, K_n)$.

Example 4. *Consider $E = \mathbb{R}^2$. Let K_2 consist of four sets P_i - the i'th quadrant for $i = 1, \ldots, 4$, let $K_1 = \{P_5, \ldots, P_8\}$ with $P_{i+4} = P_i \cap P_{i+1}$ with the sum modulo 4. We let K_0 consist of a singleton $P_9 = \{0\}$. Then (K_0, K_1, K_2) is a piecewise partitioning of $\mathbb{R}^2$.*

Example 5. *Consider $E = \mathbb{R}^2$ with K_2 consisting of P_1 - the first quadrant, P_2 - the second quadrant and P_3 - the lower half plane; notice $E = P_1 \cup P_2 \cup P_3$. Let $K_1 = \{P_4, P_5, P_6, P_7\}$, where $P_4 = P_1 \cap P_3$, $P_5 = P_2 \cap P_3$, $P_6 = P_1 \cap P_2$ and $P_7 = \mathrm{bd}(P_1)$. We let $K_0 = \{0\}$. In spite of $|K| = \mathbb{R}^n$, K is not a partitioning since $P_4 = P_1 \cap P_3$ is not a face of P_3.*

A combinatorial equivalent to a neighborhood is now introduced. Let K be a polyhedral complex. A **star** of a cell Q consists of all cells that have Q as a face

$$\mathrm{St}(Q) \equiv \{P \in K |\ Q \preceq P\}.$$

[1]Recall a definition of a supporting hyperplane. For a given subset $U \subset \mathbb{R}^n$ we say that H is a supporting hyperplane of U if the distance between U and H is 0 but the intersection of the interior of U with H is empty.

Star is not a complex in general, since condition (1) of definition of the complex may not be satisfied. We can make star into a complex by adding all its missing faces - the closed star. For this we define a closure $\mathrm{cl}(H)$ of a subset H of a complex K as the smallest complex that contains H. Thus the closed star is the closure of the star. The closed star of Q can be restricted to the cells of dimension i

$$\mathrm{St}_i(Q) \equiv (\mathrm{cl}(\mathrm{St}(Q)))_i = \{P \in K_i |\, Q \preceq P\}.$$

For an i-cell Q we define an **adjacent neighborhood** as the set of all adjacent i-cells to Q

$$\mathrm{Ad}_i(Q) = \{P \in K_i | \exists F \in K_{i-1},\ K \prec Q,\ K \prec P\}$$

Let $S = \bigcup_{j \in J} P_j$ for some index set J, then the adjacent neighborhood of S is

$$\mathrm{Ad}_i(S) = \bigcup_{j \in J} \mathrm{Ad}_i(P_j).$$

Example 6. *Consider a complex as in Fig. 9.1. Then*

$$\begin{aligned} \mathrm{St}(e_5) &= \{P_2, P_3, e_5\},\ \mathrm{cl}(\mathrm{St}(e_5)) = \{P_2, P_3, e_2, e_4, e_5, e_6, e_7, v_2, v_3, v_4, v_5\}, \\ \mathrm{St}_2(e_5) &= \{P_2, P_3\},\ \mathrm{Ad}_2(P_2) = \{P_1, P_2, P_3\}. \end{aligned}$$

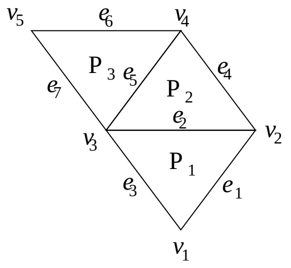


Figure 9.1: Illustration of the star and adjacent neighborhood.

The next two sections introduce two concepts fundamental for this exposition: a switched control system and a combinatorial control system. Both objects are associated to a piecewise linear partition K of the state space E. The switched control system is a continuous object living on the space $|K|$, whereas the combinatorial control system is a discrete object living on the complex K.

9.4 Switched Control Systems

We define a class of switched control systems that live on a piecewise linear partition of the state space.

A **switched control system** $\mathcal{S}$ is a quadruple $\mathcal{S} = (E, K, U, S)$, where E is a bounded polyhedral set (a polytope) in $\mathbb{R}^n$, K is a piecewise affine partition of E, U is a polyhedral set (of admissible inputs) in $\mathbb{R}^m$, and $S = \{s_P |\ P \in K_n\}$ is a family of piecewise affine control systems

$$s_P:\ \dot{x} = A_P x + B_P u + a_P,$$

where A_P, B_P, and a_P are $n \times n$, $n \times m$ and $n \times 1$ matrices, respectively. The state $x \in V_P$, where V_P is an open neighborhood of P, and $u \in U$.

Let $\mathcal{S}$ be a switched control system. Following [Habets and van Schuppen, 2004] and [Habets *et al.*, 2006] a control objective for $\mathcal{S}$ is decomposed into two control problems each formulated for a cell $P \in K_n$.

Problem 7. *Let $\mathcal{S}$ be a switched control system, and $P \in K_n$. Suppose R is an arbitrary nonempty subset of facets of P.*

Find an $(m \times n)$-matrix F_P and an m-vector g_P such that the control law

$$g_P : V_P \to \mathbb{R}^m,\ g_P(x) = F_P x + f_P, \tag{9.2}$$

satisfies:

1. $\mathrm{Im}(P) \subset U$ *and*
2. *all flow lines ϕ_x of the closed-loop system*

$$l_p:\ \dot{x} = (A_P + B_P F_P)x + (a_P + B_P f_P), \tag{9.3}$$

starting at a point $x \in P$ leave P in finite time by crossing one of the faces in R.

Problem 8. *Let $\mathcal{S}$ be a switched control system, and $P \in K_n$.*

Find a control law (9.2) such that

1. $\mathrm{Im}(P) \subset U$ *and*
2. *for any $x \in P$ the flow line ϕ_x of the closed-loop system (9.3) satisfies*

$$\phi_x(t) \in P \textit{ for all } t \geq 0.$$

Let F be a facet of a cell P with the supporting hyperplane[2]

$$H^- = \{x \in \mathbb{R}^n |\ \langle x, N\rangle \leq a\}.$$

[2] N is the outward normal vector to F

We say that the control law (9.2) **blocks** F if the vector field ξ of the closed loop system (9.3)

$$\xi_P : P \to \mathbb{R}^n,\ x \mapsto (A_P + B_P F_P)x + (a_P + B_P f_P)$$

satisfies the equality

$$\langle \xi_P(x), N \rangle \leq 0 \tag{9.4}$$

for all $x \in P$. A facet that can be blocked will be called a blocking facet.

Problems 7 and 8 are solved in [Habets *et al.*, 2006] by blocking facets in the set S - being the complement of R. We observe that if $S' \subset S$ and the control law k_P blocks all the faces in S then it also blocks the faces in S'; thus the more blocking faces the more restrictive control becomes.

A family of control laws solving Problems 7 and 8 give rise to a switched feedback law and a closed loop switched system. A **switched feedback law** for a switched control system $\mathcal{S} = (E, K, U, S)$ is a triple $\mathcal{C} = (E, K, G)$, where $G = \{g_P |\ P \in K_n\}$, where g_P is defined by (9.2).

A (closed loop) **switched system** is a triple $\mathcal{L} = (E, K, L)$, where $L = \{l_P |\ P \in K_n\}$ with l_P defined by (9.3). The overall dynamics of a switched system $\mathcal{L}$ is governed by the differential inclusion

$$\dot{x} \in \xi(x),\ \text{ where } \xi : E \to 2^{\mathbb{R}^2},\ x \mapsto \{v \in \mathbb{R}^n |\ v = \xi_P(x) \text{ if } x \in P\}.$$

For stability results for switched systems the reader is referred to [Leth and Wisniewski, 2009] and the references therein.

The focus in this work is on the combinatorial part of the control synthesis problem, i.e. on a supervisor that selects blocking faces of a switched control system such that every trajectory of the closed loop system starting in an n-cell P_{init} reaches the goal n-cell P_{goal} in finite time. For a treatment of the conditions for guaranteeing control to a certain facet of a cell system the reader is referred to [Habets *et al.*, 2006].

9.5 Combinatorial Vector Field

Taken as a whole, the goal of this work is to formulate a combinatorial counterpart of a control system. The object of interest is a polyhedral complex K.

An intermediate step is to impose a group structure on each of the sets K_i, introduced in Sec. 9.3. For this we adapt a concept of a chain group, Definition IV.1.5 in [Bredon, 1997]. The **i-chain group** $C_i(K)$ of the polyhedral complex K is the free abelian group generated by i-cells. Hence $C_i(K)$ is the product of $N \equiv |K_i|$ copies of $\mathbb{Z}_2$, each corresponding to an i-cell, $\mathbb{Z}_2[P_1] \times \mathbb{Z}_2[P_2] \times \ldots \times Z_2[P_N]$ with $P_j \in K_i$:

$$C_i(K) = \bigoplus_{P \in K_i} \mathbb{Z}_2,$$

Thus an i-chain is a formal sum

$$Q = \sum_{P \in K_n} n_P P$$

of i-cells P with coefficients $n_P \in \mathbb{Z}_2$. In articular if $i < 0$ or $i > n$ we have $K_i = \emptyset$, and $C_i(K) = 0$.

Example 7. *The 2-chain $C_2(K)$ in Fig. 9.1 is*

$$C_2(K) = \{0,\, P_1,\, P_2,\, P_3,\, P_1 + P_2,\, P_1 + P_3,\, P_2 + P_3,\, P_1 + P_2 + P_3\}.$$

For a polyhedral complex K we define an i-**combinatorial scalar product** $\langle \cdot, \cdot \rangle_i : K_i \times K_i \to \mathbb{Z}_2$ by

$$\langle P, Q \rangle_i = \begin{cases} 1 & \text{if} \quad P = Q \\ 0 & \text{otherwise} \end{cases}$$

We extend it to the bilinear product $\langle \cdot, \cdot \rangle_i : C_i(K) \times C_i(K) \to \mathbb{Z}_2$. In particular

$$\left\langle \sum_j n_j P_j, P_k \right\rangle_i = n_k.$$

In the sequel we will drop the subscript i in the notation of the i-combinatorial scalar products, i.e. we will write $\langle \cdot, \cdot \rangle$ instead of $\langle \cdot, \cdot \rangle_i$.

We define an i-**boundary map** $\partial_i : C_i(K) \to C_{i-1}(K)$ by first specifying it on i-cells

$$\partial_i(P) = \sum_{F \prec P,\ \dim(F) = i-1} F.$$

Hence the i-boundary map takes an i-cell to the sum of its facets. We then linearly extend the domain of the i-boundary map from K_i to C_i:

$$\partial_i(P + Q) = \partial_i(P) + \partial_i(Q).$$

Observe that $\partial_{i-1} \circ \partial_i = 0$. To prove this claim we observe that each facet F of a facet of polyhedral set P is the intersection of two facets of P, c.f. [Grünbaum, 2003]. For any $P \in K_i$ we have

$$\partial_{i-1} \circ \partial_i(P) = \sum_j \partial_{i-1}(F_j) = \sum_j \sum_k F_j \cap F_k,$$

but $F_{kj} \equiv F_k \cap F_j = F_j \cap F_k \equiv F_{jk}$, and $F_{kj} + F_{jk} = 0$. Hence, we conclude that $\sum_j \sum_k F_{jk} = 0$.

The definition of a combinatorial vector field below is motivated by the discussion in Sec. 9.4. Let $\mathcal{S}$ be a switched control system, and $\mathcal{C}$ a switched feedback satisfying Problem 7 and 8. It follows that for any $P \in K_n$ the flow line with the starting point in P

will leave it in final time through the facets $R = \{F_1, \ldots, F_N\}$, as given by the definition of R in Problem 7. The combinatorial information of this action is

$$P \mapsto F_1 + \ldots + F_N.$$

This will anchor a definition of an n-combinatorial vector field taking an n-cell to a linear combination of its facets. In this exposition we have chosen to state it more generally for any i-cell.

Let K be a complex. An i-**combinatorial vector field** is a map

$$\nu_i : K_i \to C_{i-1}(K)$$

that satisfies

1. for any $P \in K_i$, $\nu_i(P) \in C_{i-1}(\text{cl}(P))$.
2. for any pair $(F, P) \in K_{i-1} \times K_i$, $\langle F, \nu_i(P)\rangle \nu_{i-1}(F) = 0$.
3. for any pair $(P, Q) \in K_i \times K_i$ with $P \neq Q$, $\langle \nu_i(P), \nu_i(Q)\rangle = 0$

Condition (1) states as desired that a cell is mapped to a linear combination of its facets. Condition (2) means that if an $(i-1)$-cell $F \in \text{Im}(\nu_i)$ then $\nu_{i-1}(F) = 0$. Intuitively it means that the future cells do not increase the dimension. Condition 3. pronounces that two cells cannot be mapped to the same facet. Since, if there are two cells P and Q such that $\nu(P) = F_1 + F_2 + \sum_{j=3}^{N} F_j$ and $\nu(Q) = F_1 + F_2 + \sum_{j=3}^{M} G_j$ then $F_1 + F_2 \subseteq P \cap Q$. This contradicts with the definition of the polyhedral complex. If the cells Q and P maps to the same face F, then $\langle F, \nu_i(P)\rangle = \langle F, \nu_i(Q)\rangle = 1$, i.e. $\langle \nu_i(P), \nu_i(Q)\rangle = 1$, which is again a contradiction.

A similar concept of a combinatorial vector field was introduced in [Forman, 1998] for the purpose of discrete Morse theory.

Alternatively a combinatorial vector field is a set of pairs

$$V \equiv \{(P, F_1 + \ldots + F_N) |\ P \in K,\ F_i \text{ are facets of } P\}$$

such that no facet F_i is in more than one pair. It is helpful to picture a combinatorial vector field on K by arrows, where the tail is at P and the arrow at F_i, see Fig. 9.2. Fig. 9.3 indicates that both splitting and merging of flow directions are allowed. Intuitively the arrow notation gives the same information as a phase-plane plot for continuous systems.

Since no cell is in more than one pair in V, every cell $P \in K$ of the complex K satisfies precisely one of the following conditions:

1. P is a tail of an arrow;
2. P is the head of exactly one arrow;
3. P is neither the tail nor the head of any arrow.

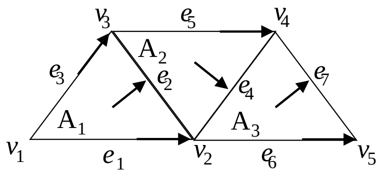
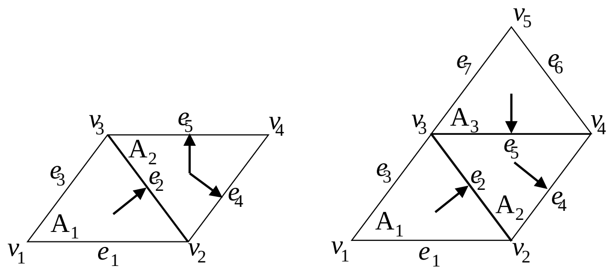


Figure 9.2: A combinatorial vector field. The vertex v_1 is a rest point.



(a) Splitting, $\nu_2(A_2) = e_4 + e_5$ (b) Merging, $\nu_2(A_1) = e_2 \prec A_2$, $\nu_2(A_3) = e_5 \prec A_2$

Figure 9.3: Both merging of the flow direction on the left hand side and splinting on the right hand side are allowed.

A simplex that satisfies condition 3 is called a rest point. In Fig. 9.2, the rest point is the vertex v_1.

Let ν be a combinatorial vector field on a complex K. We say that $P \in K_i$ is a **rest point** of ν of index i if

1. $\nu_i(P) = 0$ and

2. there is no $Q \in K_{i+1}$ such that $\langle P, \nu_{i+1}(Q)\rangle = 1$.

Example 8. *Fig. 9.4 illustrates a rest point of the minimal index* 0 *- a source and the maximal index* n *- a sink.*

Example 9. *In contradiction to the intuition the cell* A_2 *in Fig. 9.5 is not a source, since the interpretation of the figure is merely that non of the facets* e_2, e_6, e_7 *can be blocked.*

To a combinatorial vector field it is possible to associate a flow. An i-**flow** (of a

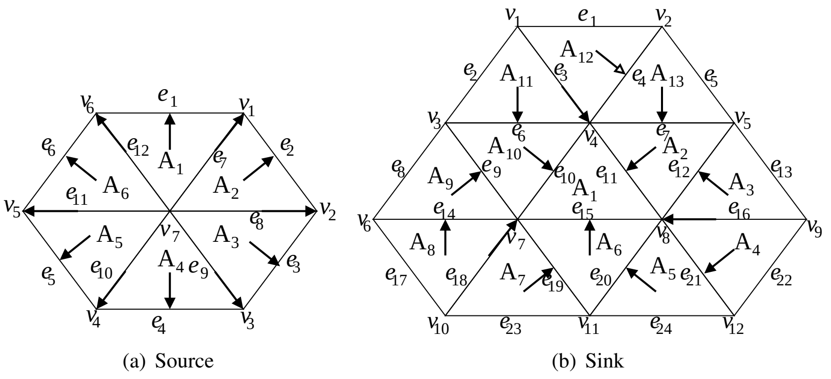

(a) Source (b) Sink

Figure 9.4: (a) v_7 is a rest point of index 0; (b) A_1 is a rest point of index n.

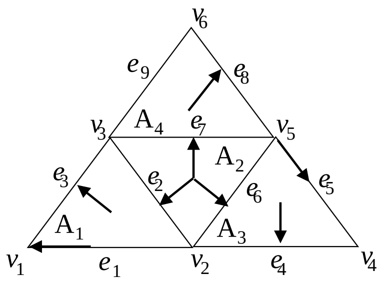


Figure 9.5: The cell A_2 is not a rest point.

combinatorial vector field) is the homomorphism[3] $\Phi_i : C_i(K) \to C_i(K)$ given by

$$\Phi_i(P) = \sum_{Q\in \mathrm{Ad}_i(P)} \langle \partial_i Q, \nu_i(P)\rangle\, Q \;+ \sum_{Q\in \mathrm{St}_{i+1}(P)} \langle \partial_{i+1} Q, P\rangle\, \nu_{i+1}(Q) + P,$$

for $P \in K_i$.

If $P \in K_i$ is a rest point, then $\nu_i(P) = 0$ and for any $Q \in \mathrm{St}_{i+1}(P)$ we have $\langle P, \nu(Q)\rangle = 0$. Hence $\langle P, \Phi_i(P)\rangle = 1$.

Example 10. *Consider the combinatorial vector field defined in Figure 9.6.*

Firstly, we compute the flow starting at e_0

$$\begin{aligned}\Phi_1(e_0) &= \langle \partial_1 e_0, \nu_1(e_0)\rangle e_0 + \langle \partial_1 e_3, \nu_1(e_0)\rangle e_3 + \langle \partial_2 A_1, e_0\rangle \nu_2(A_1) + e_0 \\ &= \langle v_0 + v_1, v_0\rangle e_0 + \langle v_0 + v_4, v_0\rangle e_3 + e_0 = e_3,\end{aligned}$$

[3]Φ_i is a homomorphism if and only if for any $P, Q \in C_i$, $\Phi_i(P+Q) = \Phi_i(P) + \Phi_i(Q)$.

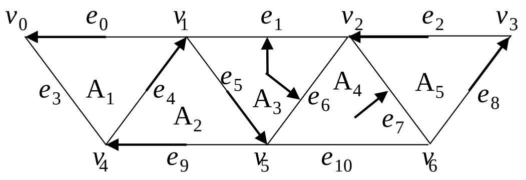


Figure 9.6: An example of a polyhedral complex with its associated vector field.

and $\Phi_1(e_3) = e_3$. We observe that e_3 is a rest point of the 1-combinatorial vector field ν_1 and a fixed point of the i-flow Φ_i.

Another example is the flow starting at e_4, which is the sum of 1-cells $e_0 + e_1 + e_5$, which corresponds to our expectation that the future is one of the three cells e_0, e_1 or e_5:

$$\begin{aligned}\Phi_1(e_4) &= \langle v_0 + v_1, v_1\rangle e_0 + \langle v_1 + v_2, v_1\rangle e_1 + \langle v_0 + v_4, v_1\rangle e_3 + \langle v_1 + v_4, v_1\rangle e_4 \\ &+ \langle v_1 + v_5, v_1\rangle e_5 + \langle v_4 + v_5, v_1\rangle e_9 + e_4 = e_0 + e_1 + e_4 + e_5 + e_4 \\ &= e_0 + e_1 + e_5.\end{aligned}$$

We compute the flow at e_5:

$$\begin{aligned}\Phi_1(e_5) &= e_5 + e_6 + e_9 + e_{10} + \langle \partial_2 A_3, e_5\rangle\, \nu_2(A_3) + \langle \partial_2 A_2, e_5\rangle\, \nu_2(A_2) + e_5 \\ &= e_6 + e_9 + e_{10} + e_1 + e_6 = e_1 + e_9 + e_{10}.\end{aligned}$$

Observe that $\Phi_1(e_1) = \Phi_1(e_6) = \Phi_1(e_7) = 0$, which is interpreted as follows: If a flow line leaves a cell through a facet then it cannot run along this facet.

Lastly, we calculate the flow at the 2-cell A_3:

$$\begin{aligned}\Phi_2(A_3) &= \langle \partial_2(A_2), \nu_2(A_3)\rangle\, A_2 + \langle \partial_2(A_3), \nu_2(A_3)\rangle\, A_3 \\ &+ \langle \partial_2(A_4), \nu_2(A_3)\rangle\, A_4 + A_3 = A_4.\end{aligned}$$

Again the flow from A_3 to A_4 is found as expected.

The i-flow Φ_i generates an i-flow line. An i-**flow line** is a sequence (finite or infinite) $\{P_k\}$ such that $\langle \Phi_i(P_k), P_{k+1}\rangle = 1$. In Example 10, the infinite sequence $\ldots, e_4, e_5, e_9, e_4, e_5, e_9, \ldots$ is a flow line. It is the n flow lines that is of significance for control synthesis for combinatorial control systems. Therefore in the sequel we restrict the attention to the n-boundary maps, n-combinatorial vector fields and n-flows. We denote them ∂, ν and Φ, respectively. In particular

$$\Phi(P) = \sum_{Q \in \mathrm{Ad}_n(P)} \langle \partial Q, \nu(P)\rangle\, Q, \quad \text{for } P \in K_n. \tag{9.5}$$

9.6 Lyapunov Function

To the combinatorial vector field and its flow it is possible to associate an execution trace - a ν-path. The concept of ν-paths was first introduced in [Forman, 1998]. A **ν-path** (of length N) is a sequence ,

$$\gamma : P_0, F_0, P_1, F_1, ..., F_{N-1}, P_N, \tag{9.6}$$

such that for all $i \in \{0, 1, ..., N-1\}$

1. $F_i \in K_{n-1}$, and $P_i, P_N \in K_n$;
2. $\langle \nu_i(P_i), F_i \rangle = 1$;
3. $P_i \neq P_{i+1} = \Phi(P_i)$.

We say that ν-path γ starts at P_0 and stops at P_N.

We use the following notation. If there is a ν-path such that $Q, R \in \gamma$ then $Q \sim R$, otherwise $Q \nsim R$.

If $P_0 = P_N$ the ν-path is called closed. Two closed ν-paths, γ, $\tilde{\gamma}$, are equivalent if $\tilde{\gamma}$ can be produced by selecting another starting point of γ.

Chain recurrent sets are sets in which the flow of an element of the set will be cyclic within the set. Thus intuitively this is the case for rest points and non-stationary closed ν-paths.

$P \in K_n$ is an element of a **chain recurrent set** $\mathcal{R}$ if either

1. P is a rest point of V or
2. there is a non stationary closed ν-path γ with $P \in \gamma$

As for dynamical systems, the notion of a Lyapunov function for a combinatorial dynamical system is desirable, and as shown in [Forman, 1998] and [Franks, 1980] it is possible to find a similar Lyapunov function, which has the property, that it is constant on the chain recurrent set, and outside the set it is the negative gradient of the function towards the set.

Let $\mathcal{R}$ be a chain recurrent set. A function $f : K_n \cup K_{n-1} \to \mathbb{R}$ is a **Lyapunov function** if for any pair $(P, F) \in K_n \times K_{n-1}$ with $P \succ F$ one of the following two conditions is satisfied:

1. if $P \notin \mathcal{R}$ then
   $\begin{cases} f(P) < f(F) \text{ if } \langle F, \nu(P) \rangle = 0 \\ f(P) \geq f(F) \text{ if } \langle F, \nu(P) \rangle = 1 \end{cases}$
2. if $P \in \mathcal{R}$ then
   $\begin{cases} f(P) = f(F) \text{ if } F \sim P \\ f(P) < f(F) \text{ if } F \nsim P. \end{cases}$

From the first condition, if a ν-path

$$\gamma : P_0, F_0, P_1 \ldots, F_{N-1}, P_N,$$

with $P_i \in K_n$ and $F_i \in K_{n-1}$ is not a chain recurrent set, then we have the following relation:

$$f(P_0) \geq f(F_0) > f(P_1) \geq \ldots, f(F_{N-1}) > f(P_N),$$

which, as in the case of continuous dynamical systems, means that the Lyapunov function is decreasing along the flow line. The second condition in the definition above asserts that for the chain recurrent set

$$\gamma_{\mathcal{R}} : P_0, F_0, P_1, F_1, \ldots, F_{N-1}, P_N = P_0$$

the following equalities:

$$f(P_0) = f(F_0) = f(P_1) = f(F_1) = \ldots = f(F_{N-1}) = f(P_N)$$

hold.

We will use the combinatorial Lyapunov candidate function as a control specification on the polyhedral complex. An evident choice of a candidate function is the distance to a goal cell P_{goal}. We associate an arbitrary value r to P_{goal}. All its neighboring cells are given a value which is higher than r. Iteratively, still a higher value will be associated to all the neighboring cells to these cells. Let K be the polyhedral complex with a possibly empty unsafe polyhedral complex, $K_u \subset K$, the algorithm for a Lyapunov candidate function is as follows.

Algorithm 2. *Iterative method for assigning to each cell and its facets a Lyapunov function value.*

```
$K_B = \{P_{\text{goal}}\}$, $\text{f}(P_{\text{goal}}) = 0$
While $K_n \setminus (K_u)_n \neq K_B$
  $K_T = \text{Ad}_n(K_B) \setminus (K_B \cup (K_u)_n)$
  $\forall P \in K_T$,
    if $(\text{cl}(P) \cap \text{cl}(K_B))_{n-1} \cap K_u = \emptyset$,
      $\text{f}(P) = \max(\text{f}(K_B)) + 1$
      $\text{f}\left((\text{cl}(P) \cap \text{cl}(K_B))_{n-1}\right) = \left\{\max(\text{f}(K_B)) + \frac{1}{2}\right\}$
      $K_B = K_B \cup P$
```

The resulting Lyapunov candidate function is well defined. This is seen from the following. Let S be a subset of K_n and define $\text{AD}(S) \equiv \text{Ad}(S) \setminus S$. Then $\text{AD}(\text{AD}(S)) \cap \text{AD}(S) = \emptyset$.

Example 11. *An example of Lyapunov function is seen in Fig. 9.7. Here, the goal cell is denoted by G, and the initial cell by I. The grayed out center cross is a forbidden region. By assigning Lyapunov function values starting from the goal cell and outwards it is ensured, that no local minimum occurs.*

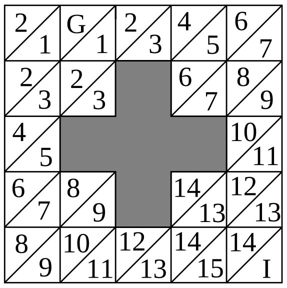


Figure 9.7: Example of Lyapunov value assignment to cells in the Swiss flag example.

Another possibility is to use a potential field alike function as described in [Hwang and Ahuja, 1992; Koren and Borenstein, 1991] with minimum at G and maximum at I, and then for each cell apply the average over the cell as a value of the Lyapunov candidate function at this cell. The disadvantage of potential field functions is that they might have local minima.

9.7 Control Synthesis

Let L_0 be a finite set (of control actions). A **combinatorial control system** is a map

$$\mu : K_n \times L_0 \to C_{n-1}(K)$$

such that for every $l \in L_0$, the restriction of μ to $K_n \times \{l\}$, is an n-combinatorial vector field.

Let $\ll$ be the following partial order relation on $C_{n-1}(K)$. For any two chains $Q = \sum_{P \in K_{n-1}} n_P P$ and $Q' = \sum_{P \in K_n} n'_P P$, we have $Q \ll Q'$ if and only if $n_P \leq n'_P$ for all $P \in K_{n-1}$.

We say that a combinatorial control system system μ is **minimal** if for any two $l_1, l_2 \in L_0$ and any $P \in K_n$ such that $\mu(l_1, P), \mu(l_2, P) \neq 0$, none of the following relations is satisfied: $\mu(l_1, P) \ll \mu(l_2, P)$ and $\mu(l_2, P) \ll \mu(l_1, P)$.

Example 12. *let $P \in K_n$ and A, B, C, D be facets of P. Let $\mu(P, \cdot)$ be given by*

$$\mu(P, l) = \begin{cases} A & \textit{for } l = 1 \\ A + B & \textit{for } l = 2 \\ B + C & \textit{for } l = 3 \\ B + C + D & \textit{for } l = 4. \end{cases}$$

We have

$$A \ll A + B \textit{ and } A + B \ll B + C + D,$$

therefor μ is not a minimal combinatorial control system. It is seen that minimal combinatorial control system rules out redundant controls. If a continuous flow line leaves P through A it also leaves P through $A + B$.

Example 12 indicates that for control synthesis we may assume the combinatorial control systems to be minimal. By having the control objective specified by a Lyapunov function f it is now possible to synthesize a control system by selecting a control action from L_0 that satisfies condition 1 and 2 of the Lyapunov function definition.

Denote the set of all possible combinatorial control systems satisfying the control objective given by the Lyapunov function f by Ω. Let J be an index set, then

$$\Omega \equiv \{\omega_i : K_n \to L_0 |\, i \in J\} \tag{9.7}$$

such that for any $\omega \in \Omega$, and for any pair $(P, F) \in K_n \times K_{n-1}$ with $P \succ F$ one of the following two conditions is satisfied:

1. if $P \neq P_{\text{goal}}$ then
$\begin{cases} f(P) < f(F) \text{ if } \langle F, \mu(P, \omega(P))\rangle = 0 \\ f(P) \geq f(F) \text{ if } \langle F, \mu(P, \omega(P))\rangle = 1 \end{cases}$
2. if $P = P_{\text{goal}}$ then
$\begin{cases} f(P) = f(F) \text{ if } F \sim P \\ f(P) < f(F) \text{ if } F \nsim P, \end{cases}$

where $F \sim P$ means that F and P belong to the same ν-path generated by $\mu \circ (\text{id}, \omega)$ and $\text{id} : K_n \to K_{n-1}$ is the identity map.

If $\Omega \neq \emptyset$, then all ν-path of the closed loop system $\mu \circ (\text{id}, \omega)$ stops at P_{goal}. There might be, however, more than one combinatorial control system in Ω. This flexibility can be used to select the control $\overline{\omega}$, which minimizes a certain cost function. Let $H : K_n \times L_0 \to \mathbb{R}$ be a function,

$$\overline{\omega} = \arg \min_{\omega \in \Omega} \max_{\gamma_\omega} \sum_{P \in \gamma_\omega} H(P, \omega(P)),$$

where γ_ω is a ν-path for the n-combinatorial vector field $\mu \circ (\text{id}, \omega)$ that starts at P_{init} and stops at P_{goal}.

In practice one wants that the decision of the control action is local. For each cell the set of possible control actions is determined to find the next feasible controller.

Let $J : K_n \to \mathbb{R}$ be a function. For an initial cell P_{init} and a goal cell P_{goal} we want to find a combinatorial control such that at each cell P, $\max J(\Phi^\omega(P))$ is minimal, where Φ^ω is the flow for the n-combinatorial vector field $\mu \circ (\text{id}, \omega)$. The following procedure can be used.

Algorithm 3.

```
$P = P_{\text{init}}$
*While* $P \neq P_{\text{goal}}$
  $\overline{\omega} = \arg\min_{\omega \in \Omega} \max J(\Phi^\omega(P))$
  $P = \Phi^\omega(P)$
```

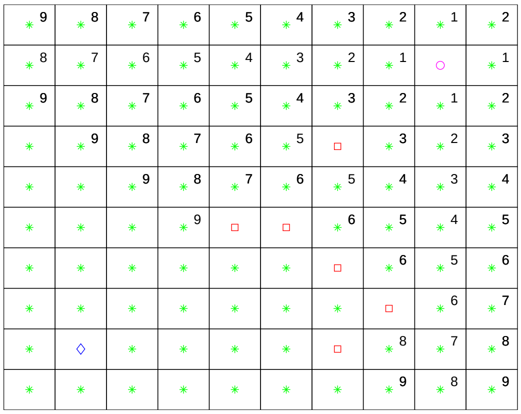


Figure 9.8: Assignment of Lyapunov values to cubes.

9.8 Numerical example

In this section a numerical example is given. It comprises path planning and obstacle avoidance for a small robot modeled as a unicycle, as described in [Oriolo *et al.*, 2002].

State Space. The domain of the example is a 100 by 100 square, which is divided into 100 equally sized cubes. This generates a cubical complex. The space space is depicted in Fig. 9.8, where the diamond indicates the starting location, the circle the goal location, the squares unsafe regions and stars safe regions.

Lyapunov function. To aid the control synthesis a Lyapunov function is used. In this example the Lyapunov function f maps a cube to the smallest number of cubes joining this cell with the goal. For this Algorithm 2 is applied. An intermediate result of the algorithm is shown in Fig. 9.8. For ease of reading only the values of the Lyapunov function at the 2-cells are depicted.

Modeling. The model of a unicycle robot is derived in [Oriolo *et al.*, 2002]. The work shows how the robot's mathematical model, through exact feedback linearization, can be transformed into two independent double integrators

$$\ddot{p} = a, \quad \ddot{\theta} = \alpha, \tag{9.8}$$

where a and α are the linear and angular acceleration, respectively.

For the purpose of this example the kinematic model is described in $\mathbb{R}^2$ by

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \dot{p} \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}. \tag{9.9}$$

To simplify the problem two internal controllers are used. The first controller keeps the robot at a constant speed. This is accomplished by a standard proportional control with a proportional gain of $k_p = 10$ and a reference of $\dot{p}_{ref} = 2$. The second controller regulates the robot's angle, which is carried out by using a proportional-differential controller with $k_p = 20$ and $k_d = 30$ and as input reference the desired angle θ_{ref}. By using these two controllers the system given by (9.8) and (9.9) can be reduced to

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \dot{p}_{ref} \begin{bmatrix} \cos\theta_{ref} \\ \sin\theta_{ref} \end{bmatrix} . \tag{9.10}$$

Since, throughout this example, $\dot{p}_{ref} = 2$ the affine system approximation of (9.10) becomes

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = 2 \begin{bmatrix} \sin\bar{\theta} \\ -\cos\bar{\theta} \end{bmatrix} \theta_{ref} + 2 \begin{bmatrix} \cos\bar{\theta} \\ \sin\bar{\theta} \end{bmatrix} , \tag{9.11}$$

with $\bar{\theta}$ being the linearization point for the given cell.

Controllability. Given the system model (9.11) the admissible and blocking facets are identified. Assume that the flow enters a cell P perpendicular to a facet, i.e. $\bar{\theta} = n\frac{\pi}{2}$. Enumerate the facets of P from F_1 to F_4 starting from the facet through which the cell was entered. It is seen from (9.11) that $\{F_1, F_2, F_3\}$, $\{F_1, F_2, F_4\}$ and $\{F_1, F_3, F_4\}$ are sets of blocking facets, which correspond to admissible facets F_2, F_3 or F_4.

Control synthesis. The algorithm selecting the combinatorial control systems Ω, c.f. (9.7), satisfying the Lyapunov function f is applied. Since there might be more than one control action for a cell a cost function which favors horizontal movement is used in Alg. 3. In Fig. 9.9 the solid line shows the evolving control synthesis.

Numerical simulation. To verify that the developed controller executes the control actions as desired, the original non-linear system is simulated with the combinatorial control law. Fig. 9.10 depicts the resulting trajectory of the closed loop system by the dashed line. The solid line represents the discrete control. It is seen that the synthesized controller makes the original non-linear system behave as specified.

9.9 Conclusion

It has been shown in this paper how a switched control system can be represented by a discrete equivalent - a combinatorial control system. The abstraction has been justified through the derivation of discrete equivalents to general concepts from continuous dynamical systems such as vector fields, flow lines and Lyapunov functions. It has further been shown how, by using the Lyapunov function as a control specification, automatic control synthesis for a combinatorial dynamical system can be accomplished.

Finally, the methodology has been exemplified by application to robot guidance, where, for readability, a simplified version of the unicycle trajectory planning problem was considered.

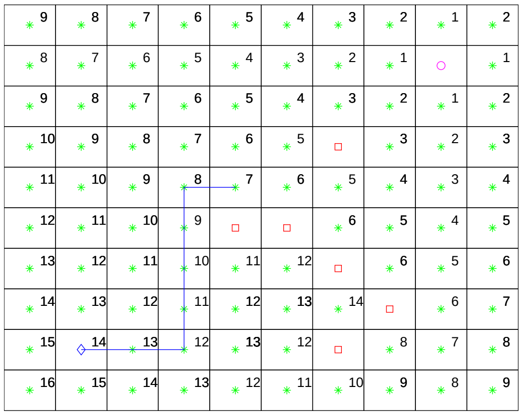


Figure 9.9: Choosing the controllers.

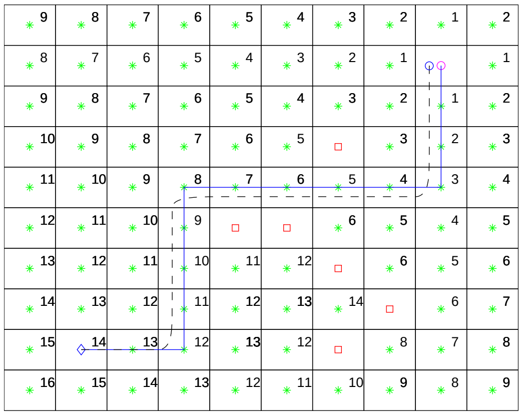


Figure 9.10: Assignment of Lyapunov values to cubes.

Part III

Closure

This part contains a summary and conclusion of the thesis, a discussion of the thesis contributions, and considerations about future work.

Chapters

Chapter 10

Conclusion

In the following the main conclusions for the thesis will be drawn up.

10.1 Part I

The first part of this thesis mainly considers prior art. An overview of previous work on deterministic hybrid systems is given in chapter 2, where hybrid system formulations such as Mixed Logic Dynamics, Linear Complementary Systems, Max-Min-Plus-Scaling and Piecewise Affine hybrid systems are described and related.

Following this, two examples of practical applications of hybrid systems are considered in chapter 3 and 4. In the first chapter control of a pulsed welding machine is consdered, and it is shown how it is possible to reformulate the system as a hybrid system and specify the control objectives as an LTL-formulation. Finally it is shown that it is possible to validate the controller to the LTL-formulation by using a formal verification tool, UPPAAL. In the second paper formation control of satellite is considered, and particularly how control can be performed in a fault tolerant control setting. It is shown how the overall system can be modelled as a piece wise affine hybrid system, and how it is possible to reason about the systems performance in the precense of faults in the system.

Finally chapter 5 gives an overview of the proposed framework structure. It briefly touches upon the advantages of using such a formulation, and considers issues which should be taken into account. Finally an example of how the framework can be used for control is given in the last section of the chapter.

10.2 Part II

In the second part of the thesis the main theoretical part is presented. Chapter 6 gives a comparative analysis of the advantages of using the two proposed frameworks for expressing combinatorial vectorfields and combinatorial flows along with comparing theses to the original work initiated by R. Forman.

The first combinatorial framework is presented in chapter 7 and 8. The first of these chapters introduces the concept of a combinatorial hybrid system, what is understood by a combinatorial manifold, how the controllability is defined on simplices and finally shows how it is possible to do control synthesis for combinatorial hybrid systems. The second chapter dealing with the first framework extends the results of the first chapter, explains how this theory fits into a fault tolerant control setting and finally shows how a combinatorial hybrid systems setting can be used to minimize the overall mission risk through selecting paths throught the combinatorial hybrid system with minimal risk under a finite horizon.

Finally, the second combinatorial hybrid systems framework is presented in chapter 9. Here the definitions of combinatorial vector fields are reformulated in an intuitive way, and the definitions of flow and flow lines are made purely geometric, thus allowing them to work not only on simplices and simplicial complexes, but on polytopes and polyhedral complexes in general. This is expanded by introducing the Lyapunov function for both dimension n and $n - 1$, and deriving a control synthesis algorithm which takes this into account. Finally the framework is applied to a robotic control problem at the end to show how the framework is used in practice.

10.3 Future work

Since the concept of combinatorial hybrid systems is quite new there are many aspects, which are interesting to consider in future research. Three of the main problems will be touched upon in the following three chapters, which represent preliminary research, which has not yet been matured for publishing.

The first concept to consider is that of optimal control for the individual simplices. This subject is interesting, since we previously only have been checking that there exists a controller, and found the set of possible affine controllers for a given system on a simplex. This subject is touched upon in chapter 11.

Another practical issue to consider is when control is performed on a polytope. Then it might not always be possible to find a feasible input set to guarantee exit through a certain facet. It would then be sensible to consider a partitioning of the polytope into a controllable polytope and an un-controllable. This idea is presented in chapter 12.

The final feature which could be considered, is a way to reduce the number of simplices in a given simplicial complex. How it is possible to transform and reduce a simplicial complex with a control combinatorial vector field given on it is considered in chapter 13.

Chapter 11

Optimal Control for Piecewise Affine Systems defined on Simplices with Polyhedral Input Set

This chapter aims to treat optimal control problems for piecewise affine systems defined on simplices. The work here is presented in sketch form only.

11.1 Introduction

Through recent years a number of initial steps have been taken towards truly automatic control of switched dynamical systems, as seen in the works of [Tabuada and Pappas, 2003b; Bemporad *et al.*, 2000] and the references therein. Many of these are based on making a discrete abstraction of the underlying continuous system and then design the controller based on the discrete abstraction. This is however not always a very efficient controller since it does not consider the underlying dynamics of the system.

One such method of discretization which is of particular interest is the one of building a simplicial complex on the state space of interest. How to build controllers on such spaces has previously been addressed in [Habets and van Schuppen, 2004; Habets *et al.*, 2006], and more recently in [Wisniewski and Larsen, 2008] along with algorithms for finding a feasible control strategy through the discretized system on the discrete level in [Larsen and Wisniewski, 2008b].

Since a substantial amount of work has been done on designing automatic controllers for dynamical systems it is desirable to preserve the discretization part of these solutions and then consider how a discrete location is optimally entered and left. This is possible using classical optimal control tools such as the Hamiltonian approach, Dynamic programming and of a more recent nature, LMI's.

The discrete objects considered in the present work are simplices in any dimension. However, expansion of the theory to polytopes in any dimension is straightforward. In the present effort the problems discussed until now are divided into three different classes:

1. System state is not known - exit through given facet.
2. System state is known - exit through given facet.
3. System state is known - exit through given point on facet.

This division is quite natural, which will be obvious later on, since the different cases have quite different solutions.

The first case is the most conservative in the sense, that it does not assume anything about where the system might be, but just that it is somewhere within the simplex and that it should leave the simplex through a certain set of simplices.

The second case is a bit more restrictive, because it assumes, that the starting location of the system is known. In this case it is possible to calculate the trajectory of the system and thus even allowing for switching of the controller within the simplex.

The last case is the most restrictive. It is based on the second case but is extended to exit the simplex through a given point on the exit facet in comparison to the second case where it is just required to leave the simplex through any point on the facet.

In the following the three cases will be discussed. Firstly the problem setting will be clarified in Section 11.2, and the background on optimal control is refreshed in Section 11.3.

The three cases will in turn be treated in Section 11.4, 11.5 and 11.6 and a conclusion will be drawn up in Section 11.7.

11.2 Piecewise Affine Autonomous Systems Defined on Polytopes

In the following section the preliminaries of the considered system will be desribed.

Simplex The basic state space domain, on which systems are considered in this paper is simplices. A simplex in dimension n, i.e. an n-simplex is composed of $n+1$ linearly independent points representing the vertices of the simplex.

A simplex is a basic geometric object, i.e. by showing that the following theory applies to simplices it naturally applies to any object based on simplices. Hence, since any polyhedral set can be decomposed into a finite number of simplices, the theory also apply for any polyhedral set. And since any manifold can be approximated arbitrarily close by a polyhedral set, then it does not pose any limitation of the validity of the presented theory that it is shown in this contribution for simplices.

System definition: On each simplex of the system the dynamics in that simplex is given as an affine system with a polyhedral bounded input set:

$$\dot{x} = Ax + Bu + a \tag{11.1}$$

where, $A \in \mathbb{R}^{n\times n}$ is the systems dynamics, $B \in \mathbb{R}^{n\times m}$ the input dynamics, $a \in \mathbb{R}^m$ a constant offset, $x \in X_i = i^{th}$ − simplex and $u \in U \subset \mathbb{R}^m$(polytope).

11.3 The Optimal Control Problem

The optimal control problem can be formulated as the problem of minimising a given cost function, which generic can be written as:

$$J(g) = \int_{t_0}^{t_1} b(x(s), u(s))ds, \tag{11.2}$$

with $g_\epsilon = (t_0, x_0)$ constrained by the system equation as given by (11.1).

11.3.1 Cost Function for Piecewise Affine System

Given that the objective for the system is to leave the simplex, then an obvious cost function for this would be to define the cost as the distance from the current position to the exit facet.

Distance from a point to the exit facet With reference to figure 11.3.1 then the distance from an interior point of the simplex can be found by taking a vector from the point to an arbitrary point on the exit facet and then projecting it onto the exit facet normal.

$$\text{dist}(x, F_{exit}) = \frac{|n_1 \dot{r}|}{|n_1|}, \tag{11.3}$$

where n_1 is the normalised normal vector to the exit facet and r is a vector pointing from the interior point in question to a point on the facet, which in this case is chosen to be one of the vertices of the exit facet.

Thus in the given case the value function can be written as the following affine function

$$L(x, u, t) = n_1^T v_2 - n_1^T x + c_u^T u, \tag{11.4}$$

where v_2 is a vertex at the exit facet, and $c_u \in \mathbb{R}^{m+}$ is a semi positive definite vector used for scaling the input costs in the value function.

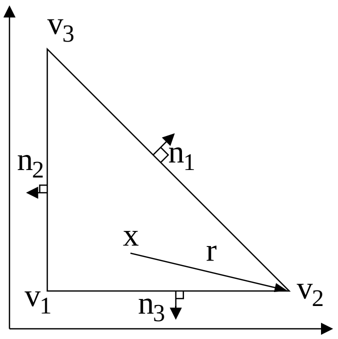


Figure 11.1: Minimum distance - point to facet

Thus the cost function can now be written as:

$$J(x,u) = \int_{t_0}^{t_1} L(x,u,t)dt + K(x_{final}), \tag{11.5}$$

where $K(x_{final})$ is a terminal cost, which, unless otherwise stated, will be assumed to be zero.

11.3.2 Hamiltonian

Looking at the particular case of a finite horizon optimal control, then the necessary conditions in the form of then the Hamiltonian is formulated as:

$$H\left(t,x,u,p(t)\right) = p(t)^T f(x,u) + L(x,u,t) = p(t)^T(Ax+Bu+a) - n_1^T x + n_1^T v_2 + c_u^T u \tag{11.6}$$

Looking at the derivatives of this we find:

$$\frac{\partial H}{\partial p} = f(x,u) = \dot{x} \tag{11.7}$$

$$-\frac{\partial H}{\partial x} = p(t)^T A + n_1^T = \dot{p} \tag{11.8}$$

Infimizing over $u \in U$ gives:

$$\inf_{u\in U} H\left(t,x,u,p(t)\right) = \inf_{u\in U} p(t)^T Ax + p(t)^T Bu + p(t)^T a - n_1^T x + n_1^T v_2 + c_u^T u \tag{11.9}$$

$$= \inf_{u\in U} (p(t)^T B + c_u^T)u + (p(t)^T A - n_1^T)x + n_1^T v_2 \tag{11.10}$$

from which it is seen that in order to infimise over u, the term $p(t)^T B + c_u^T$ should be minimized. However, the map $\mathbb{R}^m \to \mathbb{R}$ is a linear affine map, thus the infimum will be located on one of the vertices of the convex input polytope U, or possibly on a facet of the polytope.

Algorithm 4 (Calculation of the optimal input set)**.**

1. *Form the vertex set of the affine system:* V
2. *Calculate optimal input for each vertex in* V*:*

$\forall v_i \in V : \left\{ u_i^* \in U | \inf_{u \in U} H(t, v_i, u, p) = (p^T(t)B + c_u^T)u + (p(t)^T A - n_1^T)v_i + n_1^T v_2 \right\}$

$\exists u^*(t) = g\left(t, x(t), p(t)\right)$ *Conjecture: g is affine in x*

11.3.3 Dynamic Programming Approach

Again, the system is given as:

$$\dot{x}(t) = Ax(t) + Bu(t) + a$$

and the cost function, which needs to be infimized over is given by:

$$J(t_1) = \int_{t_0}^{t_1} (n_1^T x(s) + n_u^T u(s) + c_1) ds$$

The dynamic programming(DP) equation can be set up like the following: DP value function:

$$V : T \times X \to \mathbb{R},$$

with T being the time interval from t_0 to t_1, and terminal condition: $V(t_1, x) = 0$. As it is desirable to find a minimum of the value function, this is found by seeing when the derivative is zero, which can be written as:

$$0 = D_t V(t,x) + \inf_{u \in U} \left[D_x V(t,x)^T f(x,u) + b(u,x) \right] \tag{11.11}$$

$$= D_t V(t,x) + \inf_{u \in U} \left[D_x V(t,x)^T [Ax + Bu + a] + n_1^T x + n_u^T u + c_1 \right] \tag{11.12}$$

$$= D_t V(t,x) + D_x V(t,x)^T (Ax + a) + n_1^T x + c_1 + \inf_{u \in U} \left[(D_x V(t,x)^T B + n_u^T) u \right] \tag{11.13}$$

Guess
$V(t,x) = s(t)^T x + r(t)$, $V : T \times X \to \mathbb{R}$, and r, s being C^1, which gives the derivatives:

$$D_t V(t,x) = \dot{s}(t)^T x + \dot{r}(t) \tag{11.14}$$

$$D_x V(t,x) = s(t) \tag{11.15}$$

which inserted in the above gives:

$$0 = \dot{s}(t)^T x + \dot{r}(t) + s(t)^T(Ax + a) + n_1^T x + c_1 + \inf_{u\in U}\left[(s(t)^T B + n_u^T)u\right] \quad (11.16)$$

Thus we need to find $s(t)$ and $r(t)$ s.t.

$$0 = \left[\dot{s}(t)^T + s(t)^T A + n_1^T\right] x + \dot{r}(t) + \inf_{u\in U}[(s(t)^T B + n_u^T)u(t)] + s(t)^T a + c_1 \quad (11.17)$$

Thus it is now seen, that in order for the solution to be independent of the current state x, the equation can be split into the following problems:

$$0 = \left[\dot{s}(t)^T + s(t)^T A + n_1^T\right] x \quad (11.18)$$

$$\inf_{u\in U}[(s(t)^T B + n_u^T)u(t)] \quad (11.19)$$

$$0 = \dot{r}(t) + [s(t)^T B + n_u)u(t) + s(t)^T a + c_1 \quad (11.20)$$

It is worth noticing, that there might be more solutions to (11.17) than (11.18)-(11.20).

Derivative calculation

Common for both (11.18) and (11.20) is, that both their values at the termination time, i.e. t_1, is equal zero: $s(t_1) = 0$ and $r(t_1) = 0$.

Solving for $s(t)$ in (11.18) gives:

$$s(t) = -n_1^T A^{-T} + ce^{-A^T t},$$

which then can be used to calculate the time trace of $s(t)$.

Looking at (11.13), then it is seen that the term:

$$\inf_{u\in U}\left[D_x V(t,x)^T(B + n_u^T)u\right]$$

needs to be infimized. Thus it is possible to find $u_{opt} \in U$, and since the problem is affine the solution will lie on one of the vertices of U. It is thus sufficient to check the corners of U. Let the vertex set of U be denoted by U_v

$$\inf_{u\in U}\left[(D_x V(t,x)^T B + n_u^T)u\right]$$
$$\Rightarrow \inf_{u\in U}\left[(s(t)^T B + n_u^T)u\right]$$

Assuming $u_i \in U_v$ is the optimal, then it means, that

$$(s(t)^T B + n_u^T)u_i \leq (s(t)^T B + n_u^T)u_j \,,\ \forall u_j \in U_v \setminus u_i \quad (11.21)$$

11.4 Case 1: Unknown Position - Exit Through Facet

As mentioned in the introduction, then in this case it is not possible to know where the system is within this simplex. This case naturally occurs when only the transition to the simplex is observed, or if the state estimate is very poor. In this case the possible input polytope is restricted to only include the subset, which guarantees that the desired exit facet is crossed, and the remaining facets are not.

Control to a specific facet, F_1, can be done by regarding the normal vector to the facet n_1, and requires that the systems flow at the vertices of the facet will be in the same direction as the normal vector, i.e. out of the simplex. Furthermore it is required, that the flow from the opposite vertex of the face, v_1, is flowing towards the convex hull of the simplex, which is formalized in the following.

Proposition 3. *Given a n-simplex, σ_n, by the vertex set $V = \{v_1, \dots, v_{n+1}\} \in \mathbb{R}^n$ then there exists an input sequence u belonging to the convex input set $U \subset \mathbb{R}^n$, s.t. the linear affine system, $\dot{x} = Ax + Bu + a$ defined on the simplex can be controlled to the facet, given by the normal vector n_1 if, for $i \in \{2, \dots, n+1\}$*

$$\begin{aligned}
&n_1^T(Av_i + Bu + a) > 0 \wedge \\
&n_i^T(Av_j + Bu + a) \leq 0 \forall j = \{1, \dots, n+1\} \setminus \{i\} \wedge \\
&\Sigma_i < n_i, \dot{x}_{v_1} > < 0
\end{aligned} \tag{11.22}$$

has a solution.

This proposition is a reformulation of the results in [Habets and van Schuppen, 2001], where it is shown for convex polytopes, but is reproduced here in the language of this contribution.

From proposition 3 it is possible to generate an allowable input set for each vertex of the simplex. However, since this case deals with the case where nothing is known about the current system state the intersection of the input sets are used as the possible input set, i.e.

$$U' = \bigcap_i U_{v_i}.$$

An example of this will be shown in the following.

Example 13. *In the following, a small example will be used to illustrate how Alg. 4 is applied to control on a simplex:*

Simplex: The positive unit simplex extending from the origin, given by: (0,0),(1,0),(0,1).

Dynamics: $\dot{x} = Ax + Bu + a$, $A = \begin{bmatrix} -1 & -2 \\ -3 & -4 \end{bmatrix}, B = I_{2\times 2}, a = \begin{bmatrix} -5 \\ -6 \end{bmatrix}$

Input polytope: $U = [-10 \leq u_1, u_2 \leq 10]$

Requirement: Flow out of the facet with the normal vector $\sqrt{2}^{-1} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

Firstly calculate the restricted input polytope fulfilling (11.22)*, which gives:*

$$U' = \{u \in U | [1\ 1]u - 17 > 0\},$$

and is depicted in figure 13.

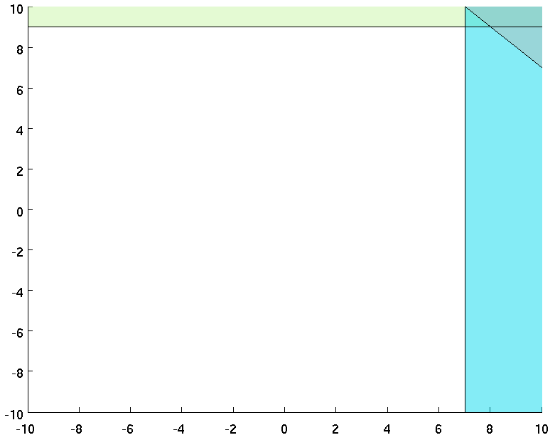


Figure 11.2: Allowable region

Secondly calculate the cost at each vertex point in U', which is shown in figure 11.3:

It is easily seen that one of the points yields a lower value than all the others. Thus this is the optimal selection of a constant input value (not surprisingly, it is at maximum actuation - $(10, 10)$*).*

The other approach to the first case is using dynamic programming instead of the Hamiltonian method. The background of dynamic programming is shown in section 11.3.3. As with the previous example the input polytope is again truncated to only contain the feasible input set for the entire polytope.

To illustrate this the system from example 13 will be used. From earlier we have, that the vertex set of U^* is:

$$U'_v = [(7, 10), (8, 9), (10, 10), (10, 9)] \tag{11.23}$$

Furthermore we have, that the input matrix is: $B = I_{2\times 2}$ and the input penalty is $n_u = (0, 0)^T$.

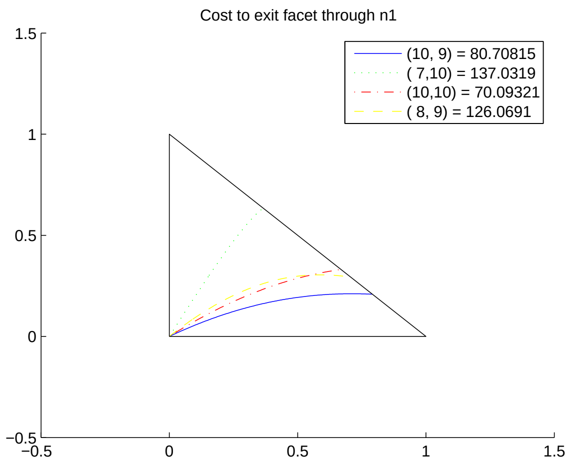


Figure 11.3: Optimal flow for the system

Now, by assuming that u_1 would be the optimal we get:

$$s(t)^T u_1 \leq s(t)^T u_2 \Leftrightarrow 7s_1 + 10s_2 \leq 8s_1 + 9s_2 \Leftrightarrow s_1 \geq s_2 \tag{11.24}$$

$$\leq s(t)^T u_3 \Leftrightarrow 7s_1 + 10s_2 \leq 10s_1 + 10s_2 \Leftrightarrow s_1 \geq 0 \tag{11.25}$$

$$\leq s(t)^T u_4 \Leftrightarrow 7s_1 + 10s_2 \leq 10s_1 + 9s_2 \Leftrightarrow 3s_1 \geq s_2 \tag{11.26}$$

Likewise for the three remaining vertices is found in table 11.1.

u_2 optimal	u_3 optimal	u_4 optimal
$s_1 \leq s_2$	$s_1 \leq 0$	$3s_1 \leq s_2$
$-2s_1 \leq s_2$	$2s_1 \leq -s_2$	$s_1 \leq 0$
$s_1 \geq 0$	$s_2 \leq 0$	$s_2 \geq 0$

Table 11.1: Inequality requirements on $s(t)$ for u_2, u_3 and u_4 being the respective optimal inputs.

The trajectory of the systems costate, $s(t)$, as described in (11.18), can now be calculated for the system, and the time trace of it is shown in figure 11.4.
Now, if the inequality requirements are compared to figure 11.4 it is seen that:

$$r_1 : s_1 > 0 \forall t \Rightarrow u_1 \wedge u_2$$

$$r_2 : s_1 > s_2 \forall t \Rightarrow u_1$$

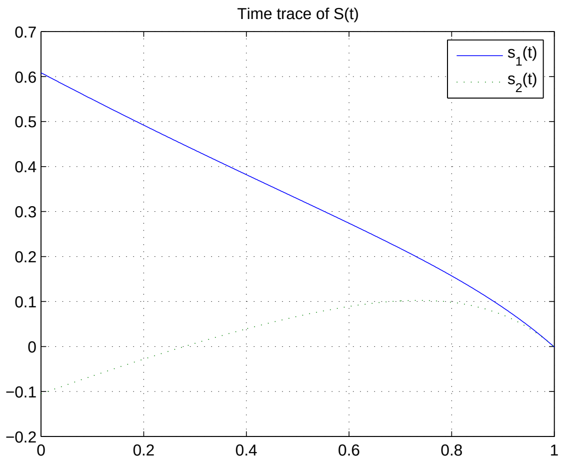


Figure 11.4: Time trace of the solution to $s(t)$

Optimal: $\bigcap r = u_1$

Remark 9. *If the inequalities had been changed u_4 would have been the optimal.*

11.5 Case 2: Known Position - Exit Through Facet

In this section the case where full state information is available will be considered.

As full state information is considered in this case it is possible to switch controllers within the simplex, thus it is not necessary to restrict the input polytope as described under Case 1. To ensure, that the system only crosses the desired facet of the simplex an extra constraint needs to be included in the Hamiltonian. Looking at the Hamiltonian in (11.6), then the augmented version becomes

$$H(t, x, u, p) = p(t)^T f(x, u) + L(x, u, t) + \mu S(x, t), \tag{11.27}$$

where the state constraints are formulated as fulfilling

$$S(t, x) \leq 0, \tag{11.28}$$

and the new co-state μ is given as

$$\mu \begin{cases} > 0, & S = 0 \\ = 0, & S < 0 \end{cases}, \tag{11.29}$$

which gives the following Euler-Lagrange equations

$$\frac{\partial H}{\partial x} = -\dot{p}(t) = \begin{cases} p(t)^T A - n_1^T + \mu \frac{\partial S}{\partial x}, & S = 0 \\ p(t)^T A - n_1^T, & S < 0 \end{cases}. \tag{11.30}$$

In other words: Once the system reaches a boundary of the state-space it will slide along the boundary until it again will be more optimal to be fully inside the simplex.

Lemma 1. *Maximum number of switches for optimal control on an n-simplex in case 2 is $2(n-1)$.*

Proof. Since, as seen from case 1, the optimal input will be in one of the input polytopes vertices during the entire time in the simplex, then this will not lead to any switching. However, when the system reaches the boundary of the system, then it will switch to a grazing control. For each time the system reaches a boundary the system is restricted to one dimension less, until reaching a dimension-1 facet, i.e. hit (n-1) boundaries, in which case it must follow that one. Similarly, to exit at the facet the system may leave the grazing giving an additional maximal (n-1) switches. □

Example 14. *An example of this is shown in the following.*

Let the simplex be defined as in the previous example. The system dynamic is $\dot{x} = Ax + Bu + a$, $A = \begin{bmatrix} 0.1 & -0.5 \\ 0.5 & 0.1 \end{bmatrix}$, $B = I_{2\times 2}, a = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ *with the input polytope being:* $u_1, u_2 \in U = [-0.2; 0.2]$

Objective as in the previous example: Flow out of the facet with the normal vector $\sqrt{2}^{-1} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.

The system is stated at $x_{init} = (0, 0.2)$. *As it is seen from figure 11.5, then the system evolves unrestricted for the first* 0.15 *s after which it hits the boundary of the simplex. At this instance the control law is changed to grazing along the facet, as seen in red. This is followed until* 0.28 *s, where it again becomes feasible to follow the original optimal trajectory. The unconstrained trajectory is plotted on top of the constrained in dotted style.*

Remark 10. *As seen from figure 11.5, then reaching a boundary of the simplex leads to a grazing behavior of the system. This phenomena means, that during the period of grazing the trajectory is suboptimal, compared to the unconstrained system evolution. There are however a number of possibilities to alleviate this phenomena. One way would be to allow for a certain degree of overlap between the simplices, ie. the trajectory would be allowed to enter a neighbouring simplex to a certain extend before switching. Something which might or might not be allowable depending on the system. Another possibility would be to perform a complete switch to the neighbouring simplex, given, that it is possible to control it back to the original simplex again. This would lead to a form of cooperation between neighbouring simplices, which will be dealt with in the following.*

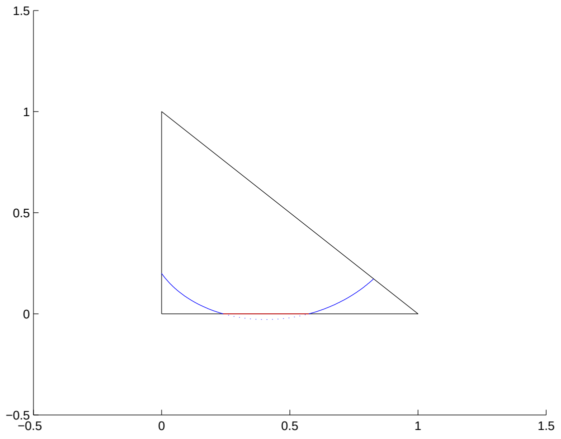


Figure 11.5: An example of a scenario when the second case is used.

11.6 Case 3: Known Position - Exit Through Point on Facet

The last case which will be discussed in this context is that of exiting the simplex through a given point on the exit facet. This has the advantage, that the trajectory of the system can be optimized over a number of simplices, as opposed to just a single simplex, which was the case in the two preceding cases.

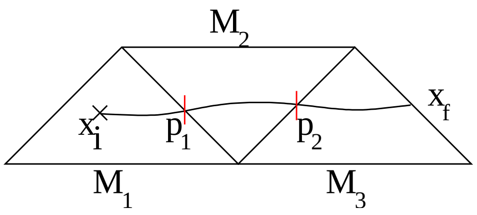


Figure 11.6: Illustration of the philosophy of case 3

An illustration of the idea behind the third case can be seen in figure 11.6. Assuming that the system contains of the three simplices M_1-M_3, and the initial and final state of the system is given by x_i and x_f respectively, then, given a cost function, there are two parameters, which can be used for doing an overall optimization on the entire system, illustrated here by the two crossing points, p_1 and p_2.

Recalling the definition of a cost function in (11.5) it is now clear, that the terminal cost term is the cost for entering the next simplex at a given position. Thus leading to a backwards recursive/iterative solution method.

In order exit a facet through a certain point an additional set of switches are needed. Recalling, that Lemma 1 required $2(n-1)$ switches to leave a certain facet, then in order to leave a facet through a certain $n-m$-facet would require an additional m switches.

Lemma 2. *Maximum number of switches for optimal control on an n-simplex with $m \leq n$ terminal constraints is $2(n-1)+m$.*

11.7 Conclusion

This chapter has touched upon development of optimal control specification on affine systems given on simplices with polyhedral input sets and affine cost functions. Such systems are quite common, e.g. in the process industry. The first case has shown how is it possible to select a suitable input for a process if the only information available is that the simplex has been entered, i.e. a facet has been crossed. The second case takes into account, that a state estimate is available, thus more agressive bang-bang control can be utillized, with possible switches of controllers within the simplex. The final case consideres how to do an overall optimization of control through a given set of simplices within a finite horizon.

Chapter 12

Autonomous Hover Flight for a Quad Rotor Helicopter using a Piecewise Affine Hybrid Systems Formulation

This paper deals with the design of an autonomous hover flight controller for an XPro helicopter. Since the system is highly nonlinear, the system is modeled as a set of piecewise affine systems defined on polytopes tied together using a hybrid automaton formulation. Many practical problems with the method is highlighted, and solutions for some of them are presented. Finally the presented concepts are applied to the actual system, and it is shown that it is feasible to design an autonomous hover controller using the presented method.

12.1 Introduction

The lack of automatic methods for control design of large non-linear systems often leads to quite time consuming and error prone human controller designs. One way of tackling this problem is to regard the nonlinear systems as a set of piecewise affine systems, a work which was initiated by Sontag [Sontag, 1981; 1982]. These piecewise affine systems (PAS) were later considered cooperatively as the dynamics in the discrete location of a hybrid automaton (PAHS) as described in [Henzinger, 1996b; Sontag, 1996]. Now, by introducing a control system on the PAHS, as introduced in [Habets and van Schuppen, 2001], it is possible to open or block certain facets, thereby guiding the system through a given subset of the PAS before reaching the target PAS.

The control problem of opening and blocking facets of PAHS defined on polytopes was first addressed in [Habets and van Schuppen, 2004] where the polytope in question was split into simplices, and the control problem was considered on each simplex separately. The simplicial control approach was further studied in [Habets *et al.*, 2006]. Even though this described approach is feasible, then it scales poorly, which is easily seen by considering an n-cube as the polytope in question. Then it requires as a minimum $n^2 - n$ simplices, but in praxis it is often divided using a Delaunay triangulation, which gives an upper bound of $n^{2^{\lceil n/2 \rceil}}$ simplices. Thus it is apparent, that it is desirable to derive the control law from the original polytope.

12.1.1 The Dragonflyer X-Pro Quad Rotor Helicopter

The system, which is considered in this paper is a quad rotor helicopter. Unlike a normal helicopter with a main rotor, which gives the lift, and a tail rotor to stabilize the helicopter around its yaw axis, then the quad rotor as four identical rotors, which combined gives the lift. The distribution of lift between the individual rotors thus gives the possibility of doing translatory as well as rotational maneuvers. Although this type of helicopter is inherently much more difficult to control than an ordinary helicopter, it has the advantage, that it is robust towards one rotor failure, which is a desirable feature when operating such helicopters either in a hostile environment, or near humans.

The helicopter in question has been retrofitted with a number of additional sensors compared to the original 3-axis gyros which it is standardly delivered with. These sensors have been added in order to get a better estimate of the helicopters attitude and position. However, there has been no changes performed on the actuators used in the system.

12.1.2 Delimitation of the control task

In this paper, the control of the helicopter will be limited to a hover configuration. This delimitation has been done due to two main reasons, the first one being, that it is desirable to keep the model relative simple, so that the focus is kept on the methodology used in addressing the problem, and secondly because an autonomous hover controller for a quad rotor helicopter, a precursor mission for an autonomous flight controller, has yet to be proven in field tests.

Firstly in this paper the model describing the systems dynamics and kinematics in hover will be presented in section 12.2. Following this in section 12.3 it will be described how the system can be regarded as a set of PAS with an overlying automaton describing how the set of PAS is traversed. Furthermore it is described how the different transition in the automaton can be either blocked or opened by use of control at the PAS level and a number of algorithms for doing this is presented. The section ends with discussing how the dimension of the system at hand can be reduced to a tangible number by only regarding essential states and decoupling certain states. Following this an example of how the actual control was calculated on one of the PAS in the system is presented in section 12.4. The paper is finished off by section 12.5 where some concluding remarks on the method is given.

12.2 Systems Dynamics

This section will provide an overview of how the non-linear helicopter model is constructed and how the model parts interact. Furthermore simplifications in the model and why these are acceptable will be discussed.

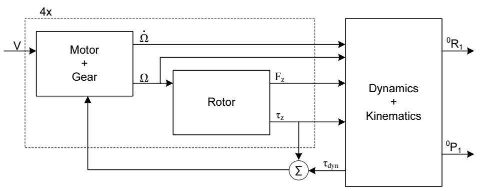


Figure 12.1: Modelling overview of the X-Pro divided into three subsystems, motor-gear, rotor and body.

As can be observed from Figure 12.1, the controllable inputs to the X-Pro are the four motor voltages V. The outputs from the model are the X-Pro attitude ${}^0_1R$ and position ${}^0_1P$, of the helicopter, as seen from the inertial frame, in this case the ground. There are three subsystems: motor-gear, body and rotor. The interfaces between these will be described in the following.

The angular velocity, Ω, produced by the motor and reduced by the gear is feed to both the rotor and body block. The angular acceleration $\dot{\Omega}$ is connected to the body block in order to calculate the body moments which eventually leads to attitude. It should be noted that the angle of attack and linear velocity of the body, which specifies angle of relative wind and dissymmetric lift respectively, has been omitted as rotor inputs. This is a natural and acceptable simplification as the main objective is to operate in a hover state. The rotor output is the force F_z and the torque τ_z. The generated rotor torque, τ_z, is lead not only to the body block, but also back to the motor along with the torque generated by the body dynamics. These two torque contributions are summarized to comprise the total load torque seen by the motor.

12.2.1 Gear-Motor Model

In the following the gear-motor model for the rotors will be stated. As this type of model is quite standard the model will be stated here without any derivation, however this can be found in [Franklin *et al.*, 2002].

Table 12.1: Empirical model parameters

(a) Motor parameters

Parameter	Description	Value	Unit
L_a	Armature inductance	0.2811	mH
R_a	Armature resistance	0.2094	Ω
K	Torque constant	0.0046	$\frac{N \cdot m}{A}$
J_{motor}	Inertia of motor	2.62	$\frac{\mu\dot{\omega}}{\tau}$
J_{gear}	Inertia of motor	1.34	$\frac{\mu\dot{\omega}}{\tau}$
b_{comb}	Combined viscous friction	0.994	$\frac{\mu N \cdot m}{rad/sec}$
τ_{comb}	Combined rolling friction	2.9	$mN \cdot m$

(b) Rotor Parameters

Parameter	Value	Unit
a_{F_z}	-208	$\frac{\mu N \cdot s^2}{rad^2}$
b_{F_z}	-9.10	$\frac{mN \cdot s}{rad}$
a_{τ_z}	8.79	$\frac{\mu Nm \cdot s^2}{rad^2}$
b_{τ_z}	1.0	$\frac{mNm \cdot s}{rad}$
c_{τ_z}	41.4	mNm

The following resulting equations for the gear-motor assembly is found, where i_a is the motor current, Ω_{motor} is the motor angular velocity:

$$i_a(s) = \frac{V_a(s) - K \cdot \Omega_{motor}(s)}{L_a \cdot s + R_a}, \; \Omega_{motor}(s) = \frac{K \cdot i_a(s) - \tau_f(s) - \tau_l(s)}{J_{motor} \cdot s} \tag{12.1}$$

The resulting equation for angular velocity after the gearing, Ω_{gear}, can then be found to be

$$\Omega_{gear}(s) = \frac{K \cdot i_a(s) - \tau_{f_{tot}}(s) - \tau_l(s)}{\eta \cdot (J_{motor} + \frac{J_{gear}}{\eta^2}) \cdot s}, \tag{12.2}$$

with τ_l being the load torque, and $\tau_{f_{tot}}$ being the total frictional torque defined as

$$\tau_{f_{tot}}(s) = \tau_{comb} + b_{comb}\Omega_{motor}(s) \tag{12.3}$$

with the empirically parameters as stated in Table 1(a).

12.2.2 Rotor Model

A sufficient model for the rotor blade dynamics in hover can be found in [Prouty, 1989] with $\Omega = \Omega_{gear}$ describing the rotor angular velocity, which gives the force acting on the z-axis:

$${}^{r}F_z = a_{F_z}\,\Omega^2 + b_{F_z}\,\Omega, \tag{12.4}$$

along with the torque acting around the z-axis:

$$^{r}\tau_z \quad = \quad a_{\tau_z}\,\Omega^2 + b_{\tau_z}\,\Omega + c_{\tau_z} \tag{12.5}$$

The parameters for Equation (12.4) and (12.5) has been found empirically and are stated in Table 1(b).

12.2.3 Helicopter Body Model

Since the X-Pro consists of several, individual moving parts affecting its movement, the vehicle is considered composed of five components: The four rotors, each producing a lift on the helicopter and the helicopter base.

Moments: Moments are calculated based on the angular velocity of the X-Pro, and on the external influences of rotors and motors. Equation (12.6) describes the torque relation between the motors, rotor momentum, rotor forces and the angular acceleration, $\dot{\omega}$, of the X-Pro.

$$^1\dot{\vec{\omega}}_1 = {^1I_{tot}}^{-1} \left\{ \begin{array}{l} \sum_{j=1}^{4} \left[{^1\vec{n}_{rotor,j}} + {^1\vec{P}_{2,j}} \times {^1\vec{f}_{rotor,j}} \right] - {^1\vec{\omega}_1} \times \left({^1I_{tot}}\, {^1\vec{\omega}_1} \right) \ldots \\ -\, {^1}\left({^{C_2}\tilde{I}_2} \right) {^1\dot{\vec{\Omega}}_{2,tot}} - {^1\vec{\omega}_1} \times \left({^1}\left({^{C_2}\tilde{I}_2} \right) {^1\vec{\Omega}_{2,tot}} \right) \end{array} \right\}, \tag{12.6}$$

with $^1P_{2,j}$ is a vector describing the rotor position relative to the base. $^1I_{tot}$ being the total helicopter inertia, and $^{C_2}\tilde{I}_2$ is the reactive inertia of each motor/gear/rotor assembly.

The angular acceleration is expressed in the X-Pro's own coordinate system.

Attitude: Calculates the orientation of the X-Pro, based on its angular velocity relative to the universal frame is done according to [Craig, 2005].

$$\dot{R} = S(\vec{\Omega})R \tag{12.7}$$

Based on these equations, the attitude of the X-Pro is derived to be:

$$^0_1\dot{R} = S\left({^0\vec{\omega}_1} \right) {^0_1R} \tag{12.8}$$

Forces: By summing the external forces acting on the X-Pro the linear acceleration in CM can be calculated:

$$^1\dot{\vec{v}}_1 = \frac{\sum_{j=1}^{4} \left[{^1\vec{f}_{rotor,j}} \right]}{m_{tot}} + {^1_0R}\,{^0\vec{g}} \Leftrightarrow {^0\dot{\vec{v}}_1} = {^0_1R} \frac{\sum_{j=1}^{4} \left[{^1\vec{f}_{rotor,j}} \right]}{m_{tot}} + {^0\vec{g}} \tag{12.9}$$

The linear acceleration is expressed in the coordinates of the X-Pro's own frame $\{1\}$ in the left side of (12.9), which coincides with what would be measured, if linear accelerometers were to be mounted in the CM of the X-Pro. The linear accelration seen from the universal frame is stated on the right side of (12.9).

Position: The position of the X-Pro relative to the universal frame, based on the attitude and the linear velocity can be calculate as follows. The linear velocity is expressed in coordinates of the X-Pro's own frame. Because of this, the position, ${}^0P_1$, of the X-Pro can be expressed as:

$$ {}^0\dot{P}_1 = {}^0\vec{v}_1 \tag{12.10}$$

12.3 Piecewise Affine Hybrid Systems Formulation

This section constitutes the main part of this paper. The first part, (12.3.1) will sum up previous theory on PAHS needed to proceed. After this the main algorithm will be presented in section 12.3.2. The following sections are devoted to specifics of this algorithm in the context of the quad rotor helicopter at hand.

12.3.1 Theories of PAHS

A Piecewise-Affine Hybrid System consists of a discrete automaton with a continuous-time affine system on a polyhedral set at each mode, and a switching mechanism between both discrete and continuous dynamics[Henzinger, 1996b].

The PAHS can through hybrid system formalism be characterized by the tuple $\mathcal{H}$ presented in (12.11), [Habets *et al.*, 2006, p.939].

$$\mathcal{H} = (Q, E, f, U, \{(X_q, \mathcal{A}_q) \mid q \in Q\}, \{(G_q(e), \mathcal{R}_q(e)) \mid (q, e) \in dom(f)\}) \tag{12.11}$$

where Q is the set of discrete locations, E the set of discrete events and f the discrete transition functions $f :\subset Q \times E \rightarrow Q$. These three elements constitute the discrete automaton whereas the continuous affine systems are composed of $\mathcal{A}_q = (A_q, B_q, a)$ defined on the polytope state set X_q. The input to the affine systems are in the set U, also defined on a polytope. The previously mentioned hybrid interactions between the automaton and the affine systems are described via the guard sets $G_q(e)$ and the affine reset maps $\mathcal{R}_q(e) : G_q(e) \rightarrow X_{f(q,e)}$.

Given a discrete location $q \in Q$, the continuous affine systems are defined by

$$\dot{x}_q(t) = A_q x_q(t) + B_q u(t) + a_q, \tag{12.12}$$

where $x_q \in X_q$ and $u \in U$. With (12.11) and (12.12) defined the evolution of the PAHS can be described. Whenever the systems state x_q leaves the polytope set X_q an event $e \in E$ will occur corresponding to the guard set $G_q(e)$, provoking a transition $f(q, e)$ eventually leading to a new discrete location. With this evolution in mind the control problem evaluates to driving a state trajectory to a specific target facet $F_j \subset X_q$ of the polytope P_d.

Proposition 4, [Habets and van Schuppen, 2004, p. 24], deals with the calculation of control signals to open or blocked facets. In the proposition the open facet is F_1, n_1 is the unit normal vector pointing out of the polytope of the facet i, for a precise description of open and blocked facets see [Habets and van Schuppen, 2004].

Proposition 4. *Let (A, B, a) be an affine system defined on a polyhedron X. Let V_1 be the set of vertices of a facet F_1 of X. The facet F_1 is open (i.e. all trajectories of the close loop systems leaves X only through F_1) if the following two conditions are satisfied*

$$\begin{aligned}
&(1) \quad \forall j \in V_1 : \\
&\qquad (a) \quad n_1^T(Av_j + Bu_j + a) > 0 \\
&\qquad (b) \quad \forall i \in W_j \setminus \{1\} : n_i^T(Av_j + Bu_j + a) \leq 0 \\
&(2) \quad \forall j \in \{1, ..., M\} \setminus V_1 : \\
&\qquad (a) \quad \forall i \in W_j : n_i^T(Av_j + Bu_j + a) \leq 0 \\
&\qquad (b) \quad \sum_{i \in W_j} n_i^T(Av_j + Bu_j + a) < 0
\end{aligned}$$

Condition (1) of Proposition 4 is utilized when calculating control signals to all the vertices contained in facet F_1, whilst condition (2) of Proposition 4 is used when calculating control signals for all vertices not contained in F_1. The affine control law $u = Fx + g$ is derived by means of (14) in [Habets and van Schuppen, 2004, p. 30] which is

$$\begin{bmatrix} v_1^T & 1 \\ \vdots & \vdots \\ v_{N+1}^T & 1 \end{bmatrix} \begin{bmatrix} F^T \\ \hline g^T \end{bmatrix} = \begin{bmatrix} u^T \\ \vdots \\ u_{N+1}^T \end{bmatrix} \tag{12.13}$$

It is evident from (12.13) that a unique solution can be found for F and g, if the system is defined on a simplexthereby leading to a unique solution for F and g.

If the system was defined on a convex polytope with more than $d + 1$ vertices, the vertex matrix would be $m \times n$, thereby not leading to a unique solution. However, by applying the More-Penrose pseudo inverse to the vertex matrix a least squares solution to the problem can be found. This is interesting from an engineering point of view as energy efficient and actuator wearing often is of importance.

12.3.2 Combinatoric Controllability

The algorithm proposed in [Habets and van Schuppen, 2004] states that even though the control is made on a polytope, the control law can only be formed by triangulating that polytope and doing local control on each simplex. The paper [Habets *et al.*, 2006] works altogether on simplices. Here however, it is desirable to consider any convex polytope.

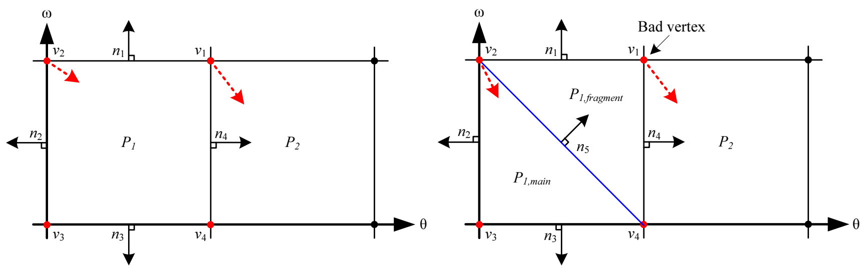
This is done on the basis of the ideas presented in the aforementioned papers. One general problem is, that it might be impossible to find a control law, which will work for the entire polytope, but unlike [Habets and van Schuppen, 2004], where the polytope is divided into a simplicial complex, the algorithm presented herein aims at doing a more optimal division of the polytope.

A short motivating example will be described in the following.

Example 15. *Consider the simple second order motor model given by:*

$$\begin{bmatrix} \dot{\theta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \omega \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \qquad (12.14)$$

and let it be defined on a set of unit cubes as shown in figure 12.2(a). It is desirable to move the system towards the origin, i.e. v_3, which means blocking facet F_1 and F_4. When calculating the needed input for blocking F_4 from v_1 it is found that $u_1 = \emptyset$. This can also intuitively be understood as a system with inertia would require an infinite input to change its state flow instantaneous. The idea now is to first look at the vertex farthest



(a) The original system. (b) After bad vertex removal.

Figure 12.2: The example system. The vertex v_3 is at the origo, and the control objective is to lead the system towards the origo, thus the two facets with normal vectors n_1 and n_4 is desired to keep blocked. With the division in (a) this is not possible. However, by removing v_1 the remaining structure fulfills the requirement as seen in (b).

from the control goal in order to keep the main part of the polytope closet to the control goal, which in this case is v_1, and then cut away this vertex. By this the cube is split into two polytopes, a main polytope containing the original vertex set without the removed one, and a fragment polytope which contains the removed vertex along with the vertices in the newly formed facet. Closing the facet is now performed again using the new main polytope, and in this case it is now seen, that it is possible to block the newly formed facet, F_5, as shown in Figure 12.2(b).

To generalize the ideas presented in the example the system at hand should first be split up into a number of convex polytopes, the number of polytopes is dependent on how accurately the PAHS should resemble the underlying original system. Each polytope can now be treated locally, ie. separately from the entire system, thus leading to a number of small separately solvable problems. For each polytope the facets and normal vectors to these are now established. The facets which are desired to have blocked are now found on the basis of the overall control goal and with that the required input polytope can be calculated according to Proposition 4. Now, as in the example, vertices, for which the input polytope is empty, are identified as bad vertices and the polytope can subsequently for each bad vertex be divided into two, one being the convex hull of the bad vertex and its adjacent vertices, which in the following will be called the fragment of the polytope and the main polytope which is the convex hull of the remaining vertices of the polytope. This procedure is called *trimming*. When all bad vertices have been removed it is possible to find a control law for the main polytope. Since it was not possible to find a control law to the original objective for the fragments of the polytope an alternative control law is found, which will bring the the system into a neighboring polytope, from which the system can reach the goal. The following algorithm states the *Combinatoric Controllabiltiy method* in a compact form.

Algorithm 5.

```
1. State space division into polytopes
2. Identify facets and normal vectors
3. Define facets to block
4. Form control space U for each vertex
5. Bad vertex identification and polytope trimming.
    (a) Identify bad vertices, if none, break.
    (b) Remove one bad vertex, results in fragment and new main structure.
    (c) Form new common facet.
    (d) Revise facets to block in both fragment and main structure.
    (e) Go to (a).
6. Control law generation on main and fragment structures.
    (a) Choose control signal from set of points in conv(U).
    (b) Find control law $u = Fx + g$ on each polytope.
7. Move to next polytope if exist go to (2), else all control laws are found, break.
```

The outcome of the above stated method when run on all polytopes are the control laws for all structures in all polytopes. Left is to connect the polytopes through guard sets and reset maps to comprise the full PAHS system.

12.3.3 Delimitation of Control Objective

The system at hand is described as a 24-dimensional system, however, since it is desirable to present a tangible example in this paper, it is desirable to reduce the number of states considerably. This is easiest observed by the following general results: The number of cubes needed to split a d-dimensional space in n sections on each axis gives a total of n^d cubes, which for the problem at hand with only one division at each axis would yield $2^{24} \approx 17 \cdot 10^6$ cubes. Thus it becomes obvious, that the number of states needs to be reduced before proceeding with applying the methodology.

The system has integral states, and these can be removed without any concern of loss of dynamic characteristics. This gives a system of 20 states, and further the rotor speed controllers can be removed, which results in a dimension of 16. If all these states are included the number of cubes, with the aforementioned division, is $2^{16} = 65536$. Albeit this number is feasible for computation, it is still considered too large in this context, hence the system will be divided into smaller parts.

The dependencies in the linear model of the X-Pro, can be viewed as the following: $\Omega_r \mapsto {}^1\omega_1 \mapsto {}^0_1q \mapsto {}^0v_1 \mapsto {}^0P_1$. These dependencies can further be divided into two, namely $\Omega_r \mapsto {}^1\omega_1 \mapsto {}^0_1q$ and ${}^0v_1 \mapsto {}^0P_1$. This division is only valid if there is an overlying supervisor which chooses the correct attitude control and then updates the position controller based on the attitude control, in order to keep the X-Pro within the control objective. With this in mind, the system is now reduced to a 10 dimensional system. Finally, the dynamics of the motor-gear-rotor system is considered being more than a decade faster than the dynamics of the X-Pro body. Since ${}^1\omega_1$ uses the rotor speed in its calculations, the rotor speed is chosen as input to the attitude model. With this, the total system dimension is reduced to 6.

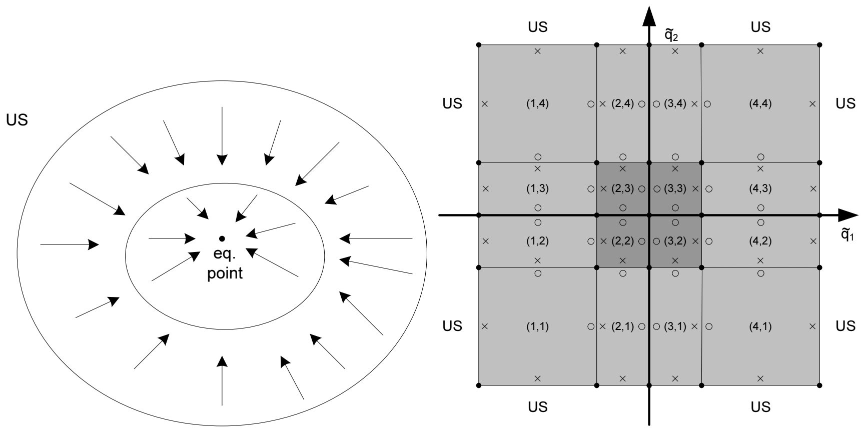
Axis Division

Following a reduction in the number of states the axis division is to be considered. The target point, i.e. the origin quite naturally formes the first point of division. From the initial requirements set for the helicopter its angular speeds should never exceed $\pi\frac{rad}{s}$, which quite naturally transforms into the boundaries of the ω_1, ω_2 and ω_3 axes. One additional division is made on these axes at half the velocity as well, i.e. $\frac{\pi}{2}\frac{rad}{s}$. The q_1 and q_2 axes, which in hover forms the two horizontal axes are likewise limited by the requirements to have a maximum inclination of 15 degrees, which gives 0.24. As to have a better performance near to the hover configuration an additional division at 0.03, corresponding to 1.5 degrees is introduced. Finally the q_3 state, which is the yaw orientation is divided into two pieces, likewise at 0.24, however, since the heading of the helicopter is not very important in hover control further subdivision is not performed. This leads to a total of 2048 6-cubes, which is a more practical number.

12.3.4 Facets to Block

In order to guide the system towards the control goal certain facets of the polytopes are blocked. From the initial example it should be clear, that due to system constraints, it is not guaranteed that a given facet can be blocked.

The general idea of assigning which facets to be blocked or opened is shown in Figure 12.3(a), that is to guide the system toward the control goal by blocking facets facing away from the control goal and keeping facets facing towards the goal open. A practical



(a) The basic idea. (b) Implementations wise.

Figure 12.3: (a)The main idea. The circles denote a set that the trajectory can not get out of ones inside. US denotes the Unsafe Set. (b) The practical implementation, facets whose normal vector is pointing away from equlibrium point are blocked. The crosses denotes blocked and circles open facets.

example of this is shown in Figure 12.3(b), where open facets are denoted by circles and blocked facets by crosses. Facets to be blocked are found by looking at the dot product of the vector going from the control goal to the polytopes barycenter and the normal vector to the facet. If this product is positive then the facet should be blocked.

12.3.5 Constructing the Control Space Polytope

Evaluating Proposition 4 for one vertex v_j towards one facet will provide one half-space, and evaluating all n facet that have vertex v_j in it, with blocked and open facets respectively, will provide n half-spaces in the control space. It is important to notice that the initial control space polytope U_j is already a polytope, due to physical limitations in the motors. In a simple 2-dimensional case the control polytope could be looking as depicted in Figure 12.4. When forming the control space polytopes there is the possibility, that

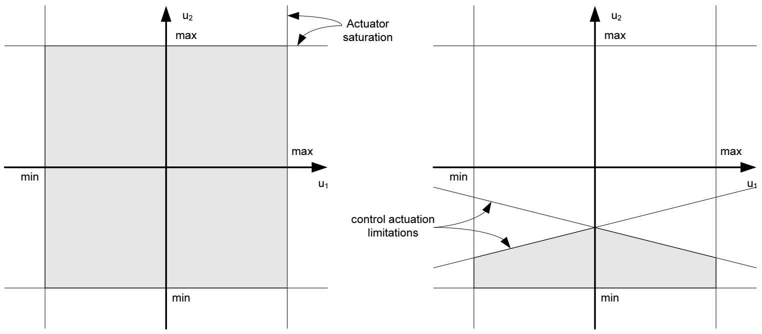


Figure 12.4: From the left: The initial bounded polytope U in the control space. Extra constraints add more halfspaces to the description of U, which limits the volume of the resulting polytope.

the resulting polytope becomes empty, thus no valid control input fulfills the requirement. In this case the vertex needs to be considered separately, which will be discussed in the following.

12.3.6 Bad Vertex Identification and Polytope Trimming

In Example 15 the idea of a bad vertex is introduced, which in short is a vertex that had no possible control signals that fulfilled the control goal. Considering the U_j polytope, then when $conv\{U_j\}$ is empty there is also no control solution. There are two reasons that can cause the $conv\{U_j\}$ to be empty. One being if the constraints in relation to the control problem, called H_c, then $H_{c1} \cap H_{c2} ... \cap H_{cN} = \emptyset$. In this case the vertex is stated as a proper bad vertex since this problem is related to that facet that is wanted blocked. The other possibility is when the constraints related to limitations of the actuators, called H_s have the characteristics of $H_s \cap H_c = \emptyset$ when $H_{c1} \cap H_{c2} ... \cap H_{cN} \neq \emptyset$. Vertices belonging to the last type is an indication of an unfeasible state space partition, and a repartition might render it feasible.

Given that some of the vertices during the control space construction turned out to be bad vertices, then a trimming of the polytope is needed in order to find a feasible control law. However, the main structure should always span $\mathbb{R}^d$. If not, then it means that there is no feasible control law. Evaluating the U polytope for all vertices with respect to feasibility will provide a set of bad vertex candidates. The chosen method of handling one bad vertex at a time give rise to the a notion of vertex classification which will decide which of the bad vertices should be handled first. This classification is done as following:

Algorithm 6.

1. *Identify the vertex nearest the control goal and classify this as zero*

2. *All class one vertices are found by following all edges from the class zero vertex*
3. *From all these class one vertices one can arrive a the set of class two vertices by following uncovered edges.*
4. *The classification continues and is finished when the class set contains only one vertex, this will be in class d, when the cube is in* $\mathbb{R}^d$.

Now, the trimming is performed for the bad vertex with the highest classification first, thus working from the outside and in towards the control goal. Take as an example the polytope given by the vertex set V_m in Figure 12.5. Assume that vertex v_8 is the proper bad vertex with the highest classification, then a trim should be performed. Thus the new main structure is $V_m = V_m \setminus \{v_8\}$, and the fragment polytope is given by $V_{f_1} = \{v_8, v_7, v_6, v_4\}$. It is clear that by this procedure a new common facet between the main structure and the fragment is formed. Since a new facet has been formed this also needs to

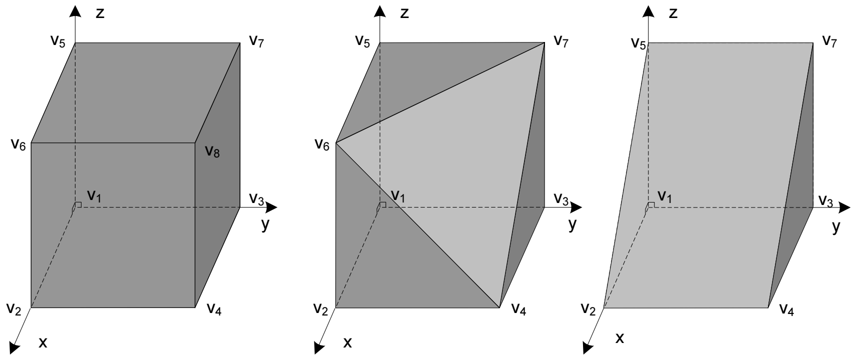


Figure 12.5: The resulting main structure after two vertices are trimmed.

have control requirements associated with it. Due to the classification method proposed in Proposition 6 the main structure will always be the closest to the control goal, thus any new facet formed seen from the main structure should be blocked, and the complementary facet seen from the fragment is kept open. The remaining facets of the fragment keeps its original properties from before the splitting. The only facets that need special attention are the facets in the fragment that can not be blocked, the solution is to open what can not be blocked, still making a control that is the best possible, and let further control be up to neighboring polytopes. Recall the unsafe area, which was the area outside the allowable state space. A fragmentation of a cube that lies up against an unsafe area will itself be defined as a unsafe area if the facet to this area can not be blocked. In this manner the unsafe area is automatically modified to a feasible solution, as is the case for the individual polytopes. This procedure is stated more compact in Algorithm 7.

Algorithm 7.

1. *Feasibility study - find all bad vertex candidates.* **If** *no bad vertex is found the polytope is fully trimmed ,* **break**.
2. *Choose bad vertex with highest classification.*
3. *Trim main with respect to chosen bad vertex.*
    (a) *Find supporting half-space that fulfils trimming properties.*
    (b) *Add common facet to both main structure and fragment.*
    (c) *Newly formed facet is wanted blocked seen from the main structure.*
    (d) *Facets that can not be blocked in the fragment structure is opened.*
4. *Repeat the procedure,* **go to** *(1).*

12.4 Example

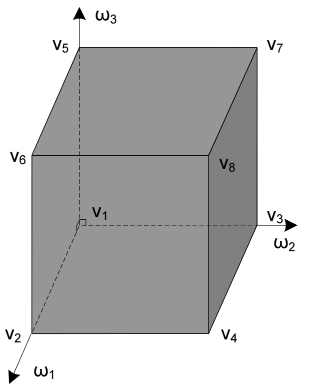
To give a clear and comprehendable example a subpart of the system is considered. Hence a three dimensional cube is considered in order to present a working control law, thereby maintaining the *Combinatoric Controllability method* sequence.

The considered state space system is constituted by the three angular velocity states $(\omega_x, \omega_y, \omega_z)$. The system is barycentric linearized, i.e. in $\left(\frac{3\pi}{4}, \frac{3\pi}{4}, \frac{3\pi}{4}\right)$ and has the following state space representation:

$$\begin{bmatrix} \dot{\omega}_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} 0 & -2.056 & -2.056 \\ 2.056 & 0 & 2.056 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} + \begin{bmatrix} 0.015 & 0.013 & 0.015 & -0.042 \\ 0.013 & 0.015 & -0.042 & 0.015 \\ 0.004 & -0.004 & 0.004 & -0.004 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} + \begin{bmatrix} -4.844 \\ 4.844 \\ 0 \end{bmatrix} \tag{12.15}$$

The affine system is defined on the 3-cube shown in Figure 12.6, with the vertices and normals as defined adjacent to it. The control objective is to drive the trajectory towards the origin of the state space, hence the open facets are (F_1,F_2,F_3) and the blocked facets (F_4,F_5,F_6). Due to space constraints only the calculation of the input polytope for v_8 will be shown in detail. As v_8 is the only vertex contained in the blocked set the inequalities are derived according to Proposition (4)(2). From a practical perspective it will prove sufficient to calculate for Proposition (4)(2a) with the harder $<$ instead of $\leq$. This assessment is made from a performance point of view, with the tradeoff in the sense that the trajectory is not allowed to evolve directly on the boundary of the facets (F_4,F_5,F_6). As the goal of the control is to direct the trajectory towards the facets this assessment is considered acceptable. The input polytope looks like the following for v_8:

$$\begin{aligned} \forall i \in \{4,5,6\} : n_i^T & \begin{bmatrix} 0.015 & 0.013 & 0.015 & -0.042 \\ 0.013 & 0.015 & -0.042 & 0.015 \\ 0.004 & -0.004 & 0.004 & -0.004 \end{bmatrix} \begin{bmatrix} u_{8_F} \\ u_{8_L} \\ u_{8_B} \\ u_{8_R} \end{bmatrix} < \\ & - n_i^T \left(\begin{bmatrix} 0 & -2.056 & -2.056 \\ 2.056 & 0 & 2.056 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \pi \\ \pi \\ \pi \end{bmatrix} + \begin{bmatrix} -4.844 \\ 4.844 \\ 0 \end{bmatrix} \right) \end{aligned}$$



Vertex	Value	Normal	Value
v_1	$(\frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2})^T$	n_1	$(0, 0, -1)^T$
v_2	$(\pi, \frac{\pi}{2}, \frac{\pi}{2})^T$	n_2	$(-1, 0, 0)^T$
v_3	$(\frac{\pi}{2}, \pi, \frac{\pi}{2})^T$	n_3	$(0, -1, 0)^T$
v_4	$(\pi, \pi, \frac{\pi}{2})^T$	n_4	$(0, 0, 1)^T$
v_5	$(\frac{\pi}{2}, \frac{\pi}{2}, \pi)^T$	n_5	$(0, 1, 0)^T$
v_6	$(\pi, \frac{\pi}{2}, \pi)^T$	n_6	$(1, 0, 0)^T$
v_7	$(\frac{\pi}{2}, \pi, \pi)^T$		
v_8	$(\pi, \pi, \pi)^T$		

Figure 12.6: The polytope (3-cube) which the linear system is defined on along with the vertices and normal vectors to the 3-cube. The facets enumeration of the polytope relate directly to the subscript of the normal vectors.

With the equation stated above the U_8 control polytope is defined. Within this polytope, control signals for the four motors must be chosen. The smallest control signal set still residing in U_8 is $[-435 \;\; 435 \;\; 435 \;\; 0]$. With the control signals for vertices $v_1, \ldots, v_7$ calculated in the same way, the following can be found:

$$\begin{bmatrix} \frac{\pi}{2} & \frac{\pi}{2} & \frac{\pi}{2} & 1 \\ \pi & \frac{\pi}{2} & \frac{\pi}{2} & 1 \\ \frac{\pi}{2} & \pi & \frac{\pi}{2} & 1 \\ \pi & \pi & \frac{\pi}{2} & 1 \\ \frac{\pi}{2} & \frac{\pi}{2} & \pi & 1 \\ \pi & \frac{\pi}{2} & \pi & 1 \\ \frac{\pi}{2} & \pi & \pi & 1 \\ \pi & \pi & \pi & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ g \end{bmatrix} = \begin{bmatrix} -277 & 277 & 277 & 0 \\ -859 & 859 & 859 & 0 \\ -1 & 0 & 0 & 0 \\ -356 & 356 & 0 & -356 \\ -356 & 356 & 0 & -356 \\ -1 & 0 & 0 & 0 \\ -1105 & 1105 & 0 & -1105 \\ -435 & 435 & 435 & 0 \end{bmatrix} \tag{12.16}$$

Taking the pseudo inverse of the vertex matrix provides the affine control law $u = Fx+g$:

$$u = F\omega + g = \begin{bmatrix} 13.875 & -64.230 & -64.230 \\ -13.875 & 64.230 & 64.230 \\ 161.907 & -111.552 & -111.552 \\ 175.782 & -175.782 & -175.782 \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} + \begin{bmatrix} -153.825 \\ 153.575 \\ 340.637 \\ 187.062 \end{bmatrix}, \tag{12.17}$$

which in the following will be verified by simulation on the 3-cube.

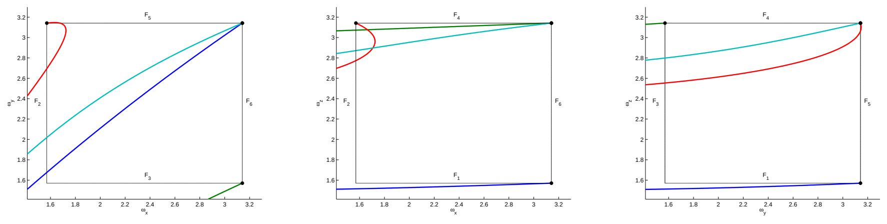
12.4.1 Simulation of control law on 3-cube

The results of the simulation of the closed loop system stated in 12.18.

$$\dot{x} = (A + BF)x + (Bg + a) \tag{12.18}$$

are shown in Figure 12.7, which is constituted by three two dimensional subplots corresponding to the $\omega_x\omega_y$, $\omega_x\omega_z$ and $\omega_y\omega_z$ axes. The system is started in the vertices v_4,v_6,v_7,v_8 and the trajectories are shown. For all the plots applies that the trajectories must leave either the left or bottom facet corresponding to the facets in the open set F_1, F_2 and F_3.

As can be seen from the graphs all trajectories are driven correctly towards the facets except two, which are contained in subfigures 12.7(a) and 12.7(c). The trajectories that arise from points $(\frac{\pi}{2}, \pi)$ and (π, π) in Figure 12.7(a) and 12.7(c) respectively, slightly exits the cube before they enter again converging towards the correct facets. It turns out that these points constitute v_7 and that the control law fails to block facet F_5.



(a) ω_x and ω_y axes. (b) ω_x and ω_z axes. (c) ω_y and ω_z axes.

Figure 12.7: Verification of the control law applied to 3-cube system.

12.5 Discussion

The objective of this paper was to show how the theories presented in [Habets and van Schuppen, 2004] and [Habets *et al.*, 2006] apply to a practical setup. The system at hand is a highly non-linear helicopter, which needs to be represented by several piecewise linear approximations to resemble its original dynamics. A task which is most intuitively done by regarding the linearized systems as being defined on n-cubes aligned with the state-space axes. This method however poses a number of challenges for which sufficient algorithms for solving has been presented. This includes algorithms useful for splitting the polytope in question into smaller polytopes in order to find a subpolytope for which the desirable control is possible along with methods for doing the division into smaller polytopes in a feasible way. Most noticeable of these is the concept of the *Combinatoric Controllability* method, which manipulates the original cube into a polytope for which the desired control is possible.

Chapter 13

Simplicial moves

Since the number of cells in a given complex easily becomes rather high, this last chapter will seek to introduce a few techniques capable of handling this issue.

13.1 Introduction

As it has become apparent through the preceeding chapters, then one can easily end up with many neighbouring simplices expressing approximately similar behaviour. This chapter will seek to give a breaf introduction into techniques capable of simplifying the simplicial complex by reducing the number of cells in it through merging similar cells to one another.

13.2 Simplicial Moves

Simplicial moves have been discussed extensively in the litterature [Lickorish, 1999b; Ludwig and Reitzner, 2006; Mijatović, 2003], and the references therein. There are only a finite number of moves available, however, rather strong results have been shown using them.

What has been considered so far is systems defined on polyhedral sets. In chapter 7 and 8 a division of the state space into a simplicial complex was considered, and in chapter 9 a division into a polyhedral complex was considered, in which each polyhedron can be regarded as a finite number of simplices with similar dynamic merged together. It is this merging, which simplicial moves tries to formalize.

Definition 39 (Bistellar move [Lickorish, 1999b])**.** *Suppose that A is an r-simplex in an abstract simplicial complex K and that* $\operatorname{lk}(A, K) = \delta B$ *for some $(n-r)$-simplex $B \notin K$.*

The bistellar move $\kappa(A, B)$ consists of changing K by removing $A \star \delta B$ and inserting $\delta A \star B$.

With $A \star B$ meaning the join of A and B.

For $n = 2$, there are three types of bistellar moves, which are depicted in Fig. 13.1. Intuitively they can be thought of as splitting a simplex into $n + 1$ simplices, changing a common facet between two simplices and merging $n + 1$ simplices into one, from left to right respectively.

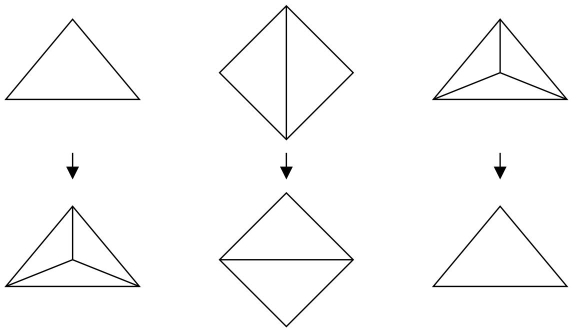


Figure 13.1: The three possible moves in two dimensions.

Having this it is clear that it is possible to manipulate a given simplicial complex to some extend. And it was shown by J. Alexander and M. H. A. Newman, that if you have two triangulations, A and B of an n-polyhedron, then it is possible to transform A to B using a finite amount of bistellar moves. To indicate this relation $A \sim B$ means, that A is equivalent to B by bistellar moves.

However, this theorem only holds true for the geomety. For control combinatorial vector fields the system dynamics also needs to be taken into account.

13.3 Simplicial Moves for Control Combinatorial Vector Fields

Recalling, that a control combinatorial vector field is caracterized by the ability to either block the system from exiting through certain facets of the simplices, or guaranteeing, that the system leaves the simplex in finite time by guaranteeing that there are no fixed points in its interior or to block all facets, which equals guaranteeing a stable fixed point in its interior, see section 5.2.2.

In order for a simplicial move to be valid for a CCVF it is required, that:

Definition 40. *Let σ_1 and σ_2 be two subcomplexes. Then $\sigma_1 \approx \sigma_2$ iff.*

1. $\sigma_1 \sim \sigma_2$

2. $\underbrace{\Phi \circ \ldots \circ \Phi}_{i} \partial\sigma_1 = \underbrace{\Phi \circ \ldots \circ \Phi}_{j} \partial\sigma_2$, $i, j \in \mathbb{N}^+$ *and where "=" is to be evaluated facetwise.*

This is a quite restrictive definition, however, if fulfilled, then it also means, that σ_1 and σ_2 are truly equivalent.

13.4 Conclusion

This final chapter has shown how it is possible to define eqivalences between different partitions of the state space. Although the definition might seem restrictive, then it is necessesary to guarantee a true equivalence between the different partitions of the systems dynamics.

Bibliography

[Alur and Dill, 1994] Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994.

[Alur and Madhusudan, 2004] Rajeev Alur and P. Madhusudan. Decision problems for timed automata: A survey. In *Formal Methods for the Design of Real-Time Systems*, volume 3185, pages 1–24. Springer Berlin / Heidelberg, 2004.

[Asarin *et al.*, 2000a] Eugene Asarin, Oliver Bournez, Thao Dang, and Oded Maler. *Approximate Reachability Analysis of Piecewise-Linear Dynamical Systems*, volume 1790/2000 of *Lecture Notes in Computer Science*, pages 20–31. Springer, 2000.

[Asarin *et al.*, 2000b] Eugene Asarin, Oliver Bournez, Thao Dang, Oded Maler, and Amir Pnueli. Effective synthesis of switching controllers for linear systems. *Proceedings of the IEEE*, 88(7):1011–1025, July 2000.

[Baker *et al.*, 2007] John Baker, Pete Bender, Pierre Binetruy, Joan Centrella, Teviet Creighton, Jeff Crowder, Curt Cutler, Karsten Danzman, Steve Drasco, Lee S. Finn, Craig Hogan, Cole Miller, Milos Miloslavljevic, Gijs Nelemans, Sterl Phinney, Tom Prince, Bonny Schumaker, Bernard Schutz, Michele Vallisneri, Marta Volonteri, and Karen Willacy. Lisa: Probing the universe with gravitational waves. Technical report, LISA Mission Science OFfice, 2007.

[Balluchi *et al.*, 2006] Andrea Balluchi, Alberto Casagrande, Pieter Collins, Alberto Ferrari, Tiziano Villa, and Alberto L. Sangiovanni-Vincentelli. Ariadne: a framework for reachability analysis of hybrid automata. In *Proceedings of the 17th International Symposium on the Mathematical Theory of Networks and Systems, Kyoto, Japan, July 2006.*, 2006.

[Barraquand *et al.*, 1992] Jérôme Barraquand, Bruno Langlois, and Jean-Claude Latombe. Numerical potential field techniques for robot path planning. *IEEE Tran. on systems, Man and Cybernetics*, 22(2):224–241, 1992.

[Behrmann *et al.*, 2001] Gerd Behrmann, Alexandre David, Kim G. Larsen, Oliver Möller, Paul Pettersson, and Wang Yi. UPPAAL - present and future. In *Proc. of 40*th *IEEE Conference on Decision and Control*. IEEE Computer Society Press, 2001.

[Bemporad and Morari, 1999a] Alberto Bemporad and Manfred Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35:407–427, 1999.

[Bemporad and Morari, 1999b] Alberto Bemporad and Manfred Morari. Verification of hybrid systems via mathematical programming. In *Hybrid Systems: Computation and Control*, volume 1569 of *Lecture Notes in Computer Science*, pages 31–45. Springer, 1999.

[Bemporad *et al.*, 2000] Alberto Bemporad, Giancarl Ferrari-Trecae, and Manfred Morari. Observability and controllability of piecewise affine and hybrid systems. *IEEE Transactions on Automatic Control*, 45(10):1864–1876, October 2000.

[Bendtsen *et al.*, 2005] Jan Dimon Bendtsen, Jakob Stoustrup, and Klaus Trangbk. Bumpless transfer between observer-based gain scheduled controllers. *International Journal of Control*, 78(7):491–504, 2005.

[Berger *et al.*, 2003] Lois Berger, Phillip Ferguson, Jonathan P. How, and Stephanie Thomas. Distributed control of formation flying spacecraft. In *AIAA Guidance, Navigation and Control Conference*, 2003.

[Blanke *et al.*, 2003] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki. *Diagnosis and Fault-tolerant Control*. Springer-Verlag, 2003.

[Bøgh, 1997] S.A. Bøgh. *Fault tolerant Control Systems*. PhD thesis, Aalborg University, Denmark, 1997.

[Boskovic and Mehra, 2003] Jovan D. Boskovic and Raman K. Mehra. An adaptive reconfigurable formation flight control design. In *Proceedings of the American Control Conference Denver, Colorado June 4-6*, 2003.

[Branicky, 1998] M. S. Branicky. Multiple lyapunov functions and other analysis tools for switched and hybrid system. *IEEE Transactions on Automatic Control*, 43(4):475–482, April 1998.

[Bredon, 1997] G. E. Bredon. *Topology and Geometry*. Springer-Verlag, 1997.

[Chen and Patton, 1999] Jie Chen and R. J. Patton. *Robust Model-Based Fault Diagnosis For Dynamic Systems*. Kluwer Academic Publishers, 1999.

[Chutinan and Krogh, 2003] A. Chutinan and B. H. Krogh. Computational techniques for hybrid system verification. *IEEE Trans. on Automatic Control*, 48(1):64–75, 2003.

[Collins and van Schuppen, 2004] Pieter Collins and Jan H. van Schuppen. Observability of piecewise-affine hybrid systems. In *Hybrid Systems: Computation and Control*, volume 2993, pages 265–279. Springer Berlin / Heidelberg, 2004.

[Craig, 2005] John J. Craig. *Introduction to Robotics Mechanics and Control, 3th. ed.* Prentice Hall, 2005.

[Decarlo *et al.*, 2000] R.A. Decarlo, M.S. Branicky, S. Pettersson, and B Lennartson. Perspectives and results on the stability and stabilizability of hybrid systems. *Proceedings of the IEEE*, 88(7):1069–1082, July 2000.

[Declerck and Staroswiecki, 1991] P. Declerck and M. Staroswiecki. Characterization of the canonical components of a structural graph for fault detection in large scale industrial plants. In *Proceedings of ECC*, pages 298–303, 1991.

[Desai *et al.*, 1999] Jaydev P. Desai, Vijay Kumar, and James P. Ostrowski. Control of changes in formation for a team of mobile robots. In *Proceedings of the 1999 IEEE Internation Conference on Robotics and Automation*, 1999.

[Elfving *et al.*, 2003] A. Elfving, L. Stagnaro, and A. Winton. SMART-1: key technologies and autonomy implementations. *Acta Astronautica*, 52:475–486, Apr 2003.

[Falcone *et al.*, 2007] Paolo Falcone, Francesco Borrelli, Jahan Asgari, Hongtei Eric Tseng, and Davor Hrovat. Predictive active steering control for autonomous vehicle systems. *IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY*, 15(3):566–580, 2007.

[Feng, 2002] Gang Feng. Stability analysis of piecewise discrete-time linear systems. *IEEE Transactions on Automatic Control*, 47(7):1108–1112, July 2002.

[Forman, 1998] Robin Forman. Combinatorial vector fields and dynamical systems. *Mathematische Zeitschrift*, 228:629–681, 1998.

[Franklin *et al.*, 2002] Gene F. Franklin, Abbas Emami-Naeini, and J. David Powell. *Feedback Control of Dynamic Systems, 4th. ed.* Prentice Hall, 2002.

[Franks, 1980] John M. Franks. *Homology and Dynamical Systems*. American Mathematical Society, 1980.

[Frehse, 2005] Goran Frehse. Phaver: Algorithmic verification of hybrid systems past hytech. In *HSCC*, pages 258–273, 2005.

[Garcia-Sanz *et al.*, 2007] Mario Garcia-Sanz, Irene Eguinoa, and Samir Bennani. Full-matrix inverse-based mimo qft control design for spacecraft flying in formation. In *Proceedings of the 17th IFAC Symposium on Automatic Control in Aerospace*, 2007.

[Goodman, 2004] Jacob E. Goodman, editor. *Handbook of discrete and computational geometry*. Chapman & Hall/CRC, 2 edition, 2004.

[Grünbaum, 2003] Branko Grünbaum. *Convex Polytops*. Springer, 2003.

[Habets and van Schuppen, 2001] L. C. G. J. M. Habets and J. H. van Schuppen. A controllability result for piecewise-linear hybrid systems. In *Proc. European Control Conference (ECC 2001)*, 2001.

[Habets and van Schuppen, 2004] L. C. G. J. M. Habets and J. H. van Schuppen. A control problem for affine dynamical systems on a full-dimensional polytope. *Automatica*, 40:21–35, 2004.

[Habets *et al.*, 2006] L. C. G. J. M. Habets, P. J. Collins, and J. H. van Schuppen. Reachability and control synthesis for piecewise-affine hybrid systems on simplices. *IEEE Transactions on Automatic Control*, 51:938–948, 2006.

[Heemels *et al.*, 2000] W. P. M. H. Heemels, J. M. Schumacher, and S. Weiland. Linear complementarity systems. *SIAM J. Appl. Math.*, 60(4):1234–1269 (electronic), 2000.

[Heemels *et al.*, 2001] W. P. M. H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37:1085–1091, 2001.

[Henzinger, 1996a] Thomas A. Henzinger. The theory of hybrid automata. *Proceedings of the 11th Annual IEEE Symp. on Logic in Computer Science*, pages 278–292, 1996.

[Henzinger, 1996b] Thomas A. Henzinger. The theory of hybrid automata. In *Proceedings of the 11th Anual Symposium on Logic in Computer Science*, 1996.

[Hudson, 1969] J. F. P. Hudson. *Piecewise Linear Topology*. W.A. Benjamin, Inc., 1969.

[Huth and Ryan, 2004] Michael Huth and Mark Ryan. *Logic in Computer Science: Modelling and Reasoning about Systems*. Cambridge Univesity Press, 2 edition, 2004.

[Hwang and Ahuja, 1992] Y.K. Hwang and N. Ahuja. A potential field approach to path planning. *Robotics and Automation, IEEE Transactions on*, 8(1):23–32, Feb 1992.

[International, 1998] Quality Associates International, editor. *FMEA - Quick Refrence Guide*. Quality Associates International, 1998. `http://www.quality-one.com/services/fmea.cfm`.

[Izadi-Zamanabadi and Larsen, 2007] Roozbeh Izadi-Zamanabadi and Jesper A. Larsen. A fault tolerant control supervisory system development procedure for small satellites - the AAUSAT-II case. In *Proc. of the 17th IFAC Symposium on Automatic Control in Aerospace*, Toulouse, France, June 2007.

[Izadi-Zamanabadi and Staroswiecki, 2000] R. Izadi-Zamanabadi and M. Staroswiecki. A structural analysis method formulation for fault-tolerant control system design. In *39th IEEE Conference on Decision and Control*, pages 4901–4902, 2000.

[Izadi-Zamanabadi, 2002] R. Izadi-Zamanabadi. Structural analysis approach to fault diagnosis with application to fixed-wing aircraft motion. In *American Control Conference*, 2002.

[Jensen and Wisniewski, 2001] Hans-Christian B. Jensen and Rafael Wisniewski. Quaternion feedback control for rigid-body spacecraft. In *Guidance, Navigation and Control Conference*, 2001.

[Koenig and Likhachev, 2002] S. Koenig and M. Likhachev. Improved fast replanning for robot navigation in unknown terrain. *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, 1:968–975 vol.1, 2002.

[Koren and Borenstein, 1991] Y. Koren and J. Borenstein. Potential field methods and their inherent limitations for mobile robot navigation. *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pages 1398–1404 vol.2, Apr 1991.

[Kouvaritakis and Cannon, 2001] Basil Kouvaritakis and Mark Cannon, editors. *Non-Linear Predictive Control: Theory and Practice*. IET, 2001.

[Lafferriere *et al.*, 2005] G. Lafferriere, A.Williams, J. Caughman, and J.J.P. Veerman. Decentralized control of vehicle formations. *Systems & Control Letters*, 54:899–910, 2005.

[Larsen and Wisniewski, 2008a] Jesper A. Larsen and Rafael Wisniewski. Combinatorial hybrid systems. In *2008 IEEE International Symposium on Computer-Aided Control System Design*, San Antonio, Texas, 2008.

[Larsen and Wisniewski, 2008b] Jesper A. Larsen and Rafael Wisniewski. Combinatorial hybrid systems. In *2008 IEEE INTERNATIONAL SYMPOSIUM ON COMPUTER-AIDED CONTROL SYSTEM DESIGN*, San Antonio, Texas, USA, September 2008. IEEE.

[Larsen *et al.*, 1997] Kim G. Larsen, Paul Pettersson, and Wang Yi. Uppaal in a nutshell. *International Journal on Software Tools for Technology Transfer (STTT)*, 1(1-2):134–152, December 1997.

[Larsen *et al.*, 2004] Kim G. Larsen, Paul Pettersson, and Wang Yi. Uppaal in a nutshell. *International Journal on Software Tools for Technology Transfer (STTT)*, 1:134–152, 2004.

[Larsen *et al.*, 2007] Jesper A. Larsen, Rafael Wisniewski, and Roozbeh Izadi-Zamanabadi. *Hybrid Control and Verification of a Pulsed Welding Process*, volume 4416/2007 of *Lecture Notes in Computer Science*, pages 357–370. Springer, 2007.

[Lee, 2002] John M. Lee. *Introduction to Smooth Manifolds*. Springer, 2002.

[Léger, 2007] A. Léger. Darwin, science-across disiplines, 2007. A Proposal for the Cosmic Vison 2015-2025 ESA Plan.

[Leth and Wisniewski, 2009] John J. Leth and Rafael Wisniewski. On formalism and stability of switched systems. In *Submitted to HSCC'09*, San Francisco, California, 2009.

[Li and Wisniewski, 2006] Zheng Li and Rafael Wisniewski. Analysis of synchronization for coupled hybrid systems. In *Proceedings of the 2006 Chinese Control Conference*, pages 7–11, 2006.

[Lickorish, 1999a] W. B. R. Lickorish. Simplicial moves on complexes and manifolds. *Geometry & Topology Monographs*, 2:299–320, 1999.

[Lickorish, 1999b] W. B. R. Lickorish. Simplicial moves on complexes and manifolds. In *Proceedings of the Kirbyfest (Berkeley, CA, 1998)*, volume 2 of *Geom. Topol. Monogr.*, pages 299–320 (electronic). Geom. Topol. Publ., Coventry, 1999.

[Ludwig and Reitzner, 2006] Monika Ludwig and Matthias Reitzner. Elementary moves on triangulations. *Discrete Comput. Geom.*, 35(4):527–536, 2006.

[Lygeros and et. al., 1999] John Lygeros and et. al. Hybrid systems; modeling, analysis and control–eecs 291e lecture notes and class projects. Technical Report UCB/ERL M99/34, Department of Electrical Engineering and Computer Science, UC Berkeley, 1999.

[May, 1992] J. P. May. *Simplicial Objects in Algebraic Topology*. The University of Chicago Press, 1992.

[McInnes, 2000] Colin R. McInnes. Autonomous path planning for on-orbit servicing vehicle. *Journal of the British Interplanetary Society*, 53:26–38, 2000.

[Meskin and Khorasani, 2006] Nader Meskin and K. Khorasani. Fault detection and isolation of actuator faults in spacecraft formation flight. In *Proceedings of the 45th IEEE Conference on Decision & Control*, 2006.

[Mijatović, 2003] Aleksandar Mijatović. Simplifying triangulations of S^3. *Pacific J. Math.*, 208(2):291–324, 2003.

[Milner, 1989] Robin Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[Morari *et al.*, 1999] M. Morari, A. Bemporad, and D. Mignone. A Framework for Control, State Estimation, Fault Detection, and Verification of Hybrid Systems. *Scientific Computing in Chemical Engineering II*, 2:46–61, August 1999.

[Mueller and Thomas, 2005] Joseph B. Mueller and Stephanie J. Thomas. Decentralized formation flying control in a multiple-team hierarchy. *Annals of the New York Academy of Sciences*, 1065:112–138, 2005.

[Munkres, 1973] James R. Munkres. *Elementary Differential Topology*. Princeton University Press, revised edition, 1973.

[Munkres, 1984] James R. Munkres. *Elements of algebraic topology*. Addison-Wesley Publishing Company, Menlo Park, CA, 1984.

[Oriolo *et al.*, 2002] G Oriolo, A. De Luca, and M Vendittelli. Wmr control via dynamic feedback linearization: design, implementation, and experimental validation. *IEEE Transactions on Control Systems Technology*, 10(6):835–852, 2002.

[Pedersen *et al.*, 2002] G. K. M. Pedersen, A. S. Langballe, and R. Wisniewski. Synthesizing multi-objective h_2/h_∞ dynamic controller using evolutionary algorithms. In *Proceedings of the 15th IFAC World Congress*, 2002.

[Pell *et al.*, 2004] Barney Pell, Douglas E. Bernard, Steve A. Chien, Erann Gat, Nicola Muscettola, P. Pandurang Nayak, Michael D. Wagner, and Brian C. Williams. An autonomous spacecraft agent prototype. *Autonomous Robots*, 5(1):29–52, Nov 2004.

[Prajna *et al.*, 2004] Stephen Prajna, Pablo A. Parrilo, and Anders Rantzer. Nonlinear control synthesis by convex optimization. *IEEE Trans. Automat. Control*, 49(2):310–314, 2004.

[Prouty, 1989] Raymond W. Prouty. *Helicopter Performance, Stability, and Control*. PWS Publishers, 1989.

[Quattrup *et al.*, 2001] Michael Melholt Quattrup, Jakob Krogh-Srensen, and Rafael Wisniewski. Passivity based nonlinear attitude control of the Rømer satellite. In *IFAC Symposium on Aerospace*, pages 85–90, 2001.

[Quottrup *et al.*, 2004] Michael Melholt Quottrup, Thomas Bak, and Roozbeh Izadi-Zamanabadi. Multi-robot planning: A timed automata approach. In *Proc. of the IEEE 2004 International Conference on Robotics and Automation*, New Orleans, USA, May 2004. IEEE.

[Rugh and Shamma, 2000] W. Rugh and J. S. Shamma. Research on gain scheduling? *Automatica*, 36:1401–1425, 2000.

[Scharf *et al.*, 2003] Daniel P. Scharf, Fred Y. Hadaegh, and Scott R. Ploen. A survey of spacecraft formation flying guidance and control (part i): Guidance. In *Proceedings of the American Control Conlerence Denver, Colorado June 4-6.2003*, 2003.

[Scharf *et al.*, 2004] Daniel P. Scharf, Fred Y. Hadaegh, and Scott R. Ploen. A survey of spacecraft formation flying guidance and control (part ii): Control. In *Proceeding of the 2004 American Control Conference Boston, Massachusetts June 30 -July 2,2004*, 2004.

[Schutter and van den Boom, 2001] B. De Schutter and T.J.J van den Boom. Model predictive control for max-min-plus-scaling systems. In *Proceedings of American Control Conference*, 2001.

[Smith and Hadaegh, 2005] Roy S. Smith and Fred Y. Hadaegh. Control of deep-space formation-flying sacecraft; relative sensing and switched information. *Journal of Guidance, Control, and Dynamics*, 28:106–114, 2005.

[Sontag, 1981] Eduardo D. Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Transactions on Automatic Control*, AC-26(2):346–358, April 1981.

[Sontag, 1982] Eduardo D. Sontag. Remarks on piecewise-linear algebra. *Pacific Journal of Mathematics*, 98(1):183–201, 1982.

[Sontag, 1996] Eduardo D. Sontag. *Interconnected automata and linear systems: a theoretical framework in discrete-time*, volume 1066, pages 436–448. Springer-Verlag, 1996.

[Storer, 2004] J. Storer. *The Haynes Manual on Welding*. Haynes Group, 2004.

[Tabuada and Pappas, 2003a] Paulo Tabuada and George J. Pappas. Finite bisimulations of controllable linear systems. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, 2003.

[Tabuada and Pappas, 2003b] Paulo Tabuada and George J. Pappas. *Model Checking LTL over Controllable Linear Systems is Decidable*, volume 2623, pages 498–513. Springer-Verlag Berlin, 2003.

[Thanapalan *et al.*, 2006] K.K.T. Thanapalan, S.M. Veres, E. Rogers, and S.B. Gabriel. Fault tolerant controller design to ensure operational safety in satellite formation flying. In *Proceedings of the 45th IEEE Conference on Decision & Control*, 2006.

[Thomsen, 2004] Jesper Sandberg Thomsen. *Advanced Control Methods for Optimization of Arc Welding*. PhD thesis, Aalborg University, 2004.

[Tsiotras and Doumtchenko, 2000] P. Tsiotras and V. Doumtchenko. Control of spacecraft subject to actuator failures: State-of-the-art and open problems. In *The Richard H. Battin Astrodynamics Conference*, 2000.

[Wisniewski and Kulczycki, 2005] Rafael Wisniewski and P. Kulczycki. Slew maneuver control for spacecraft equipped with star camera and reaction wheels. *Control Engineering Practice*, 13:349–356, 2005.

[Wisniewski and Larsen, 2008] Rafael Wisniewski and Jesper A. Larsen. Combinatorial vector fields for piecewise affine control systems. In *Proceedings of the 17th IFAC World Congress*, Seoul, South Korea, 2008.

[Wisniewski, 2006] Rafael Wisniewski. Towards modelling of hybrid systems. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 911–916, 2006.

[Zaccarian and Teel, 2002] L. Zaccarian and A.R. Teel. A common framework for anti-windup, bumpless transfer and reliable designs. *Automatica*, 38(10):1735–1744, 2002.

[Zomorodian, 2005] Alfa J. Zomorodian. *Topology for Computing*. Cambridge University Press, 2005.